Design and Implementation of the PHaSE Core: Establishing Hardware Roots of Trust for Safety-Critical Embedded Devices

Manoel Augusto de Souza Serafim

Institute of Science and Technology – Federal University of São Paulo (UNIFESP)
São José dos Campos, São Paulo, Brazil

manoel.serafim@unifesp.br

Abstract. *The assessment of cybersecurity standards for safety-critical embedded systems has gained momentum across the aerospace, medical, defense, and automotive industries. A key challenge in complying with these standards is establishing robust Hardware Roots of Trust that account for evolving threats and malicious hardware changes. This research outlines the development of the Programmable Hardware Siloed Engine (PHaSE) core, integrated within an FPGA to act as an improvement over Trusted Platform Modules (TPMs). The design will support secure boot, ensure confidentiality, offer tamper resistance, and establish security enclaves. To assess its functionality, resource utilization during secure boot is analyzed whilst benchmarking it against commercial TPMs.*

1. Introduction

Safety-critical embedded systems necessitate modular yet robust Hardware Roots of Trust (HRoT) to assure the integrity of cryptographic algorithms. For example, HRoT maintains the incorruptibility of public keys used in secure boot [Lin and Wang 2012] and hides private keys stored in non-volatile memory[Sevinç et al. 2007]. These elements enable safety-critical systems to establish chains of trust that ensure data confidentiality, protect intellectual property, and defend against threat actors. However, incorporating components that provide HRoT lead to expensive and centralized targets that rely on intrinsic trustworthiness[De Oliveira Nunes et al. 2021].

1.1. Existing Solutions

HRoT elements are typically provided by Application-Specific Integrated Circuits (ASICs) designed to securely manage keys, protect against tampering, resist to and side-channel attacks and perform cryptographic operations. One way to insert these integrated circuits is through Trusted Platform Modules (TPMs), external circuits that communicate with the Main Processing Unit (MPU) typically via SPI or Low Pin Count buses. Another approach to implementing a HRoT is through Hardware Security Module(HSMs), which are tightly coupled cores that share an address space with the MPU. TPMs and HSMs are governed by a set of guidelines proposed by the Trusted Computing Group, which define their functionality, interfaces, and implementation. Although revisions of these standards are introduced over time, they often cannot be implemented due to the immutable nature of the existing solutions. Conventional implementations also fall short in implementing high-performance lightweight cryptographic cores. While these devices can offload some

tasks from the device's MPU, their performance limitations mean that, under time constraints, system developers often use them solely for key management. Consequently, the MPU must handle cryptographic functions in a side-channel vulnerable environment. Another downside of these solutions is their lack of modularity for safety-critical applications, as their implementation invariably necessitates hardware revisions that entail high certification costs against standards like DO-254.

1.2. Proposal

In this paper, I present the design and implementation of the Programmable Hardware Siloed Engine (PHaSE), an open core that offers a transparent, high-performance, modular and patchable set of RoT. Specifically, the modules therein act as hardware accelerators for cryptographic operations, manage keys, generate true random bits, and configure robust platform configuration state registers. This system interfaces over SPI, aiming to provide a similar interface to that of existing TPMs. The implementation adheres to the specifications of the Trusted Computing Group's TPM 2.0 standard, revision 1.83. Here, I present initial experiments demonstrating advantages of the platform compared to TPMs. Additionally, I discuss the implications of this design, how it differs from other HRoT implementations and deploy it in a Lattice LFE5U-25F FPGA.

The following section presents a background on the specification of TPMs and the use of HRoTs for safety-critical embedded systems security. Next, I describe the hardware architecture and rationale behind the PHaSE core design. I then introduce secure booting as a test problem, followed by a description of the operation of the main interfacing component. Finally, I benchmark it against one commercial TPM, and conclude with insights and suggestions for future work.

2. Roots of Trust in the Trusted Platform Module

Trust is based on the expectation of behavior. These expectations can be established through the inheritance of trust, following the principle of the chain of trust[Perez et al. 2006], but ultimately, a point is reached where an assumption of trustworthiness is made. This means that trusting is choosing to be vulnerable to the action of a foundational trusted party in which misbehavior can not be detected. A root of trust is this foundation upon which all assumptions are made.

Establishing trust is crucial, but it also introduces risks. To mitigate these risks, trusted entities in secure systems are delimited by trust boundaries. Within these boundaries, elements trust each other based on the premise of tamper resistance and intrinsic integrity. The TPM exemplifies such a system component, defined within a trust boundary and operating independently from the host system that trusts it. The boundaries in a TPM consist of computational infrastructure that does not compromise the objectives of the trusted platform[Hoeller and Toegl 2018], combined with the three basic roots of trust:

- RoT Storage (RoTS): These are memory elements that provide shielded locations, inaccessible to entities outside the trust boundary. It is here that private keys are stored.
- RoT Measurement (RoTM): This is a Processing Unit that sends measurements, composed of relevant information regarding integrity and also its identity to the

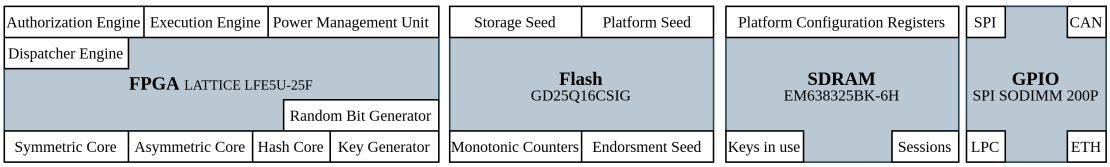

Figure 1. Hardware Architecture of PHaSE

RoTS. It is controlled by instructions, named Core RoTM, that are executed when each new chain of trust is established.

- RoT Reporting (RoTR): These are signed digests of the content from audit logs, key properties and the evidence of state of the Platform Configuration Registers(PCRs).

3. Trusted Platform Modules in Safety-Critical Systems

Building the computing infrastructure that is able to protect itself from any hardware or software attack and fault is a high priority in safety-critical embedded systems. Implementing a TPM allows the system to detect whether this computing base was compromised or not[Arthur et al. 2015]. By the use of chains of trust, the platform establishes confidentiality and authenticated integrity. TPMs provide: hashing functions, used to verify uniqueness and one-way functions; asymmetric encryption and decryption, used in key exchanges; asymmetric signing and signature verification, used for binary integrity checks; symmetric encryption and decryption, used for efficient encryption; symmetric signing and signature verification, used for integrity and authenticity using a shared key.

To address these needs, TPMs can be implemented in different ways. One approach is to integrate it into a single-chip component that includes a processor, flash memory, RAM, and ROM. This design promotes modularity by consolidating all necessary infrastructure onto a single die. However, it also consolidates a centralized target with an open access point to all TPM functionalities. Another method integrates roots of trust directly within the silicon die of the MPU, effectively creating an HSM core that shares an address space with the roots of trust. This strategy prioritizes performance and security but sacrifices modularity and backward compatibility with legacy LRUs. A third approach is in-host RoT execution with tightly controlled memory boundaries [Pereira et al. 2018].

4. Architecture

PHaSE is a high-performance module with a set of roots of trust implemented in programmable hardware that enables modularity, transparency and reliability in secure safety-critical embedded systems. The hardware architecture, depicted in Figure 1, consists of an FPGA, a NOR Serial Flash memory, and an SDRAM chip, all interfaced with the host via SPI and DDR2 SODIMM configuration.

In the FPGA, siloed cryptographic cores provide symmetric (AES256), asymmetric (ECC) and hashing (SHA-3) algorithms that are supported by a random bit generator and implemented through equivalent block ciphers. The use of these algorithms will be sanctioned by engines that will verify authorization based on the resources requested, execute measurements of host and core integrity and dispatch the instructions for each core

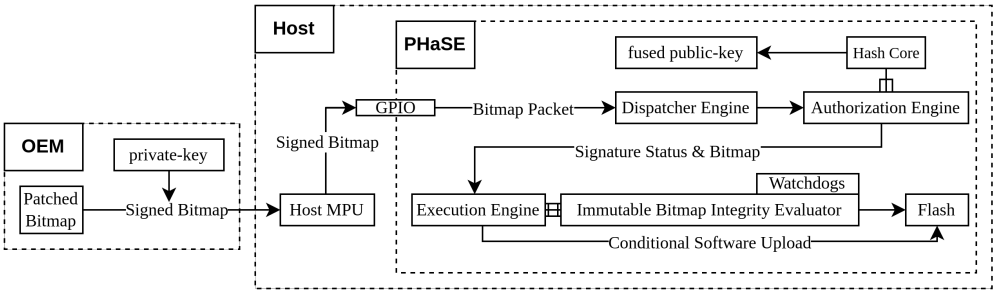

Figure 2. Bitmap Patching

whilst maintaining audit logs. In the flash, the bitmap used by the FPGA will be stored. To prevent unauthorized modifications, the bitmap contents are signed and the public key used to verify the bitmap integrity will be stored in immutable shorts within the fabric of the FPGA. This public key is generated using a private key that is only available to the OEM. After the initial load, the FPGA will only permit the upload of bitmaps that are correctly signed with this specific key. By providing this functionality, PHaSE supports vulnerability patching, as shown in Figure 2. Additionally, this flash memory is used to create shielded NVM locations, isolating certain resources from the host. These resources are only accessible to cryptographic algorithms, provided they pass authorization controls and integrity checks.

The SDRAM includes protected and unprotected regions for storing transient data, Platform Configuration Registers, active keys, and session data. Access is managed by a verifiable interface controller within the FPGA's dispatcher engine, ensuring secure and efficient address space sharing with the host while maintaining strict memory boundaries.

5. Host Secure Booting with Encrypted Images

To ensure that instructions loaded into host memory are trusted by the Original Equipment Manufacturer (OEM) and have not been compromised, it's essential to cryptographically measure and validate all code before execution. This practice is crucial in safety-critical embedded applications, preventing breaches of code integrity from single event upset or malicious activities, thus ensuring both safety and security [Ray 2019].This use case for the PHaSE was chosen because it exemplifies the device's overall operation, utilizing all of its cores and engines.

Here's how it could be applied in this scenario: When the system powers up, the PHaSE core takes control of the host MPU's execution flow. It begins by establishing the RoTM and setting up the PCR using measurements from the signed binary image of the firmware stored in the host MPU's flash memory.Within the execution engine, the authorized values of the Program Counter (PC) are verified. Provided that access is granted, the firmware image's signature is checked against a public key stored in the FPGA fabric. If the signature is verified as being from a trusted authority, the PHaSE core reads the contents of the encrypted flash memory. Using a set of hierarchical symmetric keys stored in shielded locations, the PHaSE core decrypts the firmware and loads the unencrypted version into the host flash. Once these measures are in place, control is transferred to the host MPU. The host MPU then verifies its image integrity against the value stored in the PHaSE SDRAM. With all checks completed, the boot process continues, repeating

these measurements for the bootloader binary. After the bootloader finishes executing, all subsequent operational code is checked and decrypted inside the PHaSE core.

6. Benchmarking Secure Boot

In this section I benchmark performance and security properties of the PHaSE core against the SLB 9670 by evaluating all functions used in the secure boot process. This analysis initially focuses on the hashing algorithm, and in latter itterations of this paper will be expanded to include the signature validation algorithm, and the patchability in case of a an out-of-bounds write vulnerability being found (A real vulnerability found in the SLB9670 will be used as a sample– CVE-2023-1017). Both the FPGA and the SLB 9670 will interface directly with an Atmel AT32U4 MCU via SPI outside of an Operating System, ensuring that no kernel processes interfere with measurements.

The PHaSE core utilizes the SHA-3 hashing algorithm, validated against FIPS 202. In contrast, the SLB 9670 relies on SHA-2, which is standardized in FIPS 180-4. The SHA-3 algorithm, based on the Keccak-f function, offers parallelization within digest state computations, making it well-suited for hardware implementations[Bertoni et al. 2013]. Consequently, SHA-3 achieves faster processing times on FPGA platforms. On the other hand, SHA-2 in the SLB 9670 lacks significant hardware gains due to intrinsic digest state dependencies in the algorithm. Both the PHaSE core and the SLB 9670 support the TPM Interface Specification layer. To interface with them, we can use the TPM_CC_Hash command (crafted as a packed over SPI) with the code 0x0000017D. By hooking the return of the SPI transfer and receive functions, we can precisely measure the response times of both solutions:

Table 1. Comparison of PHaSE Core and SLB 9670

Feature	**PHaSE Core**	**SLB 9670**
Construction	Sponge	Merkle-Damgard
Algorithm Considerations	Parallelizable in FPGA	No significant gains
Input	2MB Random Binary Image	
Hashing Time (ms)	48	944
Clock Speed	43 MHz Generated by PLL	43MHz
Number of Iterations	1000	

6.1. Preliminary Results

The PHaSE core outperforms the Infineon SLB 9670VQ2.0 in hashing performance. Additionally, PHaSE offers the advantage of continuous updates, ensuring compatibility with future hashing standards, which enhances its robustness and scalability. However, it is important to note that PHaSE has higher energy consumption compared to TPMs optimized for low standby power[Thomas et al. 2009].

7. Future Work and Final Remarks

This paper outlines the development of PHaSE, a programmable hardware root of trust for safety-critical systems. Future work will focus on optimizing power consumption, enhancing compatibility with various platforms, providing more benchmarks and conducting stress tests to ensure robustness. The goal is to establish PHaSE as a versatile solution for secure embedded systems, offering improved modularity and updatability.

References

Arthur, W., Challener, D., and Goldman, K. (2015). *A Practical Guide to TPM 2.0: Using the New Trusted Platform Module in the New Age of Security*. Apress Media, LLC, New York City.

Bertoni, G., Daemen, J., Peeters, M., and Van Assche, G. (2013). Keccak. In Johansson, T. and Nguyen, P. Q., editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 313–314, Berlin, Heidelberg. Springer Berlin Heidelberg.

De Oliveira Nunes, I., Ding, X., and Tsudik, G. (2021). On the root of trust identification problem. IPSN '21, page 315–327, New York, NY, USA. Association for Computing Machinery.

Hoeller, A. and Toegl, R. (2018). Trusted platform modules in cyber-physical systems: On the interference between security and dependability. In *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 136–144, London, UK.

Lin, K.-J. and Wang, C.-Y. (2012). Using tpm to improve boot security at bios layer. In *2012 IEEE International Conference on Consumer Electronics (ICCE)*, pages 376–377, Las Vegas, NV.

Pereira, L., Ortiz, L., Rossi, D., Rosa, M., Fonseca, K., Prado, C., Rust, L., Britto, A., and Riella, R. (2018). Using intel sgx to enforce auditing of running software in insecure environments. In *2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 243–246.

Perez, R., Sailer, R., and Van Doorn, L. (2006). VTPM: virtualizing the trusted platform module. In *Proceedings of the 15th Conf. on USENIX Security Symposium*, pages 305–320, Vancouver, Canada.

Ray, S. (2019). Safety, security, and reliability: The automotive robustness problem and an architectural solution. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–4.

Sevinç, P. E., Strasser, M., and Basin, D. (2007). Securing the distribution and storage of secrets with trusted platform modules. In *Proceedings of the 1st IFIP TC6 /WG8.8 /WG11.2 International Conference on Information Security Theory and Practices: Smart Cards, Mobile and Ubiquitous Computing Systems*, WISTP'07, page 53–66. Springer-Verlag.

Thomas, D. B., Howes, L., and Luk, W. (2009). A comparison of cpus, gpus, fpgas, and massively parallel processor arrays for random number generation. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '09, page 63–72, New York, NY, USA. Association for Computing Machinery.