Security overview

Security overview

Stable: 06.05.2020 - 05:24 / Revision: 06.05.2020 - 05:24

Contents

1 Article Purpose

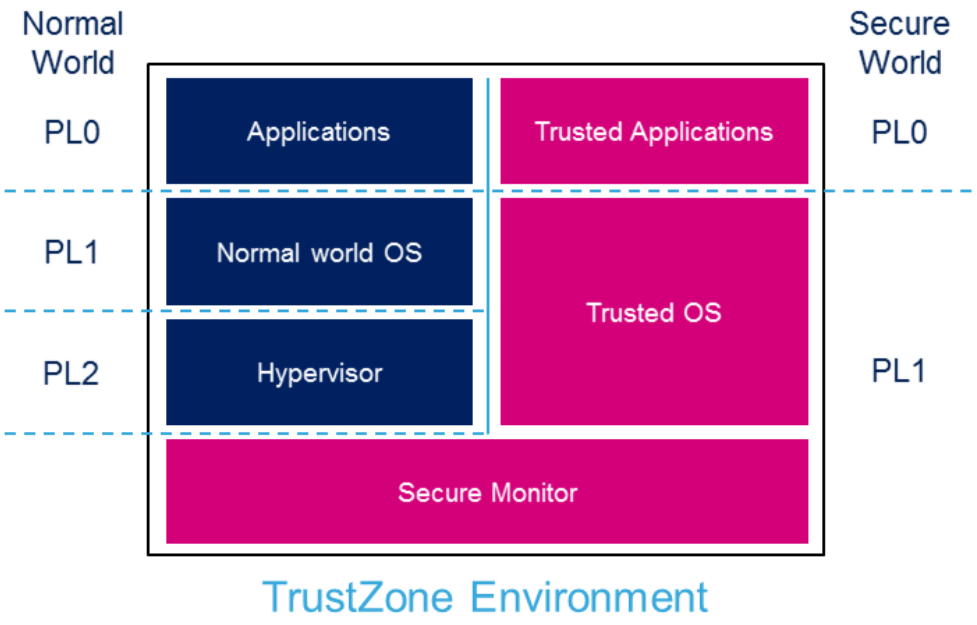
The purposes of this article is to explain how to secure an STM32 MPU-based platform thanks to several hardware mechanisms, and to briefly introduce the software components responsible for the secure configuration.

2 Introduction

The STM32 MPU is based on the Arm® Cortex-A® core, which is based on the Arm® TrustZone[1] architecture that enables context isolation: the **normal world** holds the applications whereas the **secure world** isolates all the trusted applications and core secure services so that they can safely manipulate platform secret data. The MPU includes Firewall mechanisms that allow the secure world to forbid read/write accesses from the normal world to given peripherals.

Armv7 defines PL0, PL1 and PL2 privilege levels:

- PL0 is the lowest software execution level (unprivileged calls allowed for applications).
- PL1 is the execution level for the OS.
- PL1 (secure) is also the privilege level for secure monitor execution, to switch from the secure to the normal world.
- PL2 is dedicated to the hypervisor (only non-secure).



The **normal world** is used to run rich OSs such as Linux Kernel and its applications framework.
The **secure world** runs a secure monitor with minimal services (i.e TF-A) or a TEE as secure OS (i.e OP-TEE OS).

The secure boot is a key feature of this multiple execution context environment. It allows the boot chain to be authenticated by the ROM code as well as the authentication of the components that are launched in the secure and normal worlds.

The TrustZone environment is a complete system solution that is not limited to the Cortex context. It provides a bus and peripheral infrastructure to the MPU in order to ensure that the secure world relies on a completely secured pipe when it controls a secure peripheral. The assignment of the peripherals to a given world is done thanks to a Firewall mechanism, which is set up during the secure world initialization.

Dedicated secure and normal contexts also impact the debugging facilities: depending on the targeted user, the debug can be opened to both worlds (e.g. for a secure aware developer), to normal world only (for a Linux® developer) or completely closed (for the end user). This is achieved by configuring the Debug control.

Some internal or external peripherals can be used by the secure world to support cryptographic operations. Refer to security peripherals for an introduction.

3 Secure boot

The secure boot is essential to ensure the integrity and security of the platform at runtime.

The STM32 MPU trusted boot chain was design to guarantee such a secure boot sequence.
It performs the following tasks:

- Configuration of the platform firewall, which is the foundation for a safe execution of the platform

- Configuration of the platform debug capabilities
- Verification of the integrity (thanks to a hash algorithm) and authentication (using asymmetric cryptography algorithms) of the started software components, including the trusted execution environment

TF-A is the recommended open source bootloader. Its implementation supports the trusted boot and peripheral access control with firewall.

3.1 STM32MP1

STM32MP1 secure boot implementation is described in the STM32MP15 secure boot article.

4 Firewall

MPU firewalls comprise access filters for MPU peripherals and subsystems memory mapped interfaces, internal RAMs /ROMs and external memory (DDR). Depending on the assignment, an MPU interface can be dedicated or shared between several hardware execution context(s).

4.1 STM32MP1

- ETZPC:
  - assigns access rights to MPU peripherals from Cortex-A7 contexts (secure or normal) and Cortex-M4 context.
  - assigns access rights to internal ROM/RAM from Cortex-A7 and Cortex-M4.
- TZC: assigns access rights to DDR regions.
- RCC: can restrict the access of some of its registers to the secure execution context.
- PWR: can restrict the access of some of its registers to the secure execution context.
- BSEC: The OTP memory can be fused to restrict the access to some of its content to the secure execution context.
- RTC: This MPU interface can restrict some of its interface registers to the secure execution context.
- GPIO: GPIO bank Z can be configured to restrict some GPIO configuration to the secure execution context.
- TAMP: can restrict the access of some of its registers to the secure execution context.
- EXTI: can restrict the access of some of its registers to the secure execution context.
- GIC: can restrict the access of some of its registers to the secure execution context.
- MDMA: can configure MDMA interrupt execution context.

5 Secure debug

The STM32 MPU offers the possibility to manage the platform debug configuration.
It is indeed possible to enable/disable independently secure and non secure debug accesses.

5.1 STM32MP1

Debug accesses are controlled through BSEC peripheral.
By default, the STM32 MPU is started by the ROM code with both secure and non-secure debug enabled. When the trusted boot is enabled, the ROM code disables debug accesses and relies on the FSBL to configure them.

6 Trusted execution environment

Thanks to Arm® TrustZone, the secure code is executed in an isolated context to guarantee code and data integrity: this is the TEE. It runs in parallel with the rich OS and provides secure services.

6.1 TF-A

The TF-A configuration supports the installation of a minimal runtime secure service provider and peripheral access control with firewall.

6.2 OP-TEE OS

The OP-TEE is recommended as an open source TEE solution. The package provides additional secure services to the platform since it can host core secure services and run trusted applications.

7 Security peripherals

7.1 Cryptographic hardware acceleration

The STM32 MPU embeds multiple peripherals for cryptographic acceleration:

- CRYP
- HASH

7.2 Trusted platform module (TPM)

The STM32 MPU can be associated to an external trusted platform module (TPM).
It provides secret data storage capabilities as well as cryptographic capabilities allowing to use them.

8 References

- https://www.arm.com/why-arm/technologies/trustzone-for-cortex-a

Microprocessor Unit

Operating System

Read Only Memory

Doubledata rate (memory domain)

Random Access Memory (Early computer memories generally hadserial access. Memories where any given address can be accessed when desired were then called "random access" to distinguish them from the memories where contents can only be accessed in a fixed order. The term is used today forvolatile random-accesssemiconductor memories.)

One Time Programmed

General-Purpose Input/Output (A realization of open ended transmission between devices on an embedded level. These pins available on a processor can be programmed to be used to either accept input or provide output to external devices depending on user desires and applications requirements.)

First Stage Boot Loader

Trusted Execution Environment

Trusted Firmware for Arm Cortex-A

Open Portable Trusted Execution Environment

Trusted Platform Module