iUICC POC Group Primary Platform requirements Approved release

Version 1.0

17 May 2017

This is a White Paper of the GSMA

Security Classification: Non-confidential

Access to and distribution of this document is restricted to the persons permitted by the security classification. This document is confidential to the Association and is subject to copyright protection. This document is to be used only for the purposes for which it has been supplied and information contained in it must not be disclosed or in any other way made available, in whole or in part, to persons other than those permitted under the security classification without the prior written approval of the Association.

Copyright Notice

Copyright © 2017 GSM Association

Disclaimer

The GSM Association ("Association") makes no representation, warranty or undertaking (express or implied) with respect to and does not accept any responsibility for, and hereby disclaims liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

Antitrust Notice

The information contain herein is in full compliance with the GSM Association's antitrust compliance policy.

Table of Contents

1 Introduction

1.1 Overview

In order to meet market expectations asking for deeper integration of functionalities, the GSMA has set up a PoC to perform evaluations of technical solutions, and thus to ease the migration of UICC technology within Systems on Chips (SoC). The GSMA document defines a Tamper Resistant Element able to support multiple firmwares able to address different use cases such as:

- eUICC firmware
- UICC firmware (2G/3G/4G/5G)
- Payment (VAS)
- Secure Bootstrap (VAS)
- DRM (VAS)
- Passport (VAS)
- Home Automation (VAS)
- Physical Access control (VAS)
- Logical Access control (VAS)
- Mobile authentication and identity (VAS)

The iUICC use case is an application of a generic secure platform able to support any use case requiring the benefits of security (e.g. confidentiality, authenticity, integrity and non-repudiation) and privacy (e.g. anonymity).

Due to the intrinsic performance of an integrated Tamper Resistant Element within a SoC, the fast-swapping of use cases allows virtual association of any applications running in a generic or trusted core of the SoC with secure companion applications inside the TRE.

1.2 Scope

This document describes:

- Use cases for an integrated TRE within a System on Chip
- The functional requirements of the Primary Platform
- The requirements and associated rationales for the security of the Primary Platform
- The requirements and associated rationales for the Virtual Primary Platform
- The testing requirements of the Primary Platform
- The requirements for hosting the TRE on the system-on-Chip
- A set of appendixes to facilitate the overall understanding of the iUICC PoC project, such that the reader may appreciate the rationale of the requirements and associated solutions.

1.3 Definitions and Abbreviations

Abbreviation or Term	Description
APDU	a smart card Application Protocol Data Unit
Backward Compliance	the capability of the Primary Platform to support capability of the past releases

Abbreviation or Term	Description
CLF	ContactLess Front-end (aka NFC Controller)
BIST	Built-In Self Tests
Device refresh	Operations related to return to factory settings
DMA	Direct Memory Access
ePOP	Embedded Package on Package
ETSI	European Telecommunications Standards Institute
eUICC	TRC, embedded in a device, hosting a firmware obeying to the GSMA RSP requirements
Firmware	Use case dependent data that may contain: OS and application data. Some data can be interpretable by a virtual machine or semantically equivalent principles. The firmware may be segmented.
firmware loading	The operations performed by the Primary Platform for loading a whole firmware into a blank or erased TRE
firmware update	Operations perform by the Primary Platform or the firmware itself for upgrading some parts of the firmware.
Forward Compliance	Capability of the Primary Platform to support future firmware within the limits of its resources
GSM	Global System for Mobile Communications
GSMA	GSM Association
HCI	Host Controller Interface (defined in ETSI TS 102 622)
HCP	Host Controller Protocol (defined in ETSI TS 102 622)
IDS	Image Delivery Server
Image	An instance of a generic data format encapsulating firmware, cryptographic material, and directives, which together can be processed by PBL
iNVM	internal Non-Volatile Memory
ISO	International Standard Organization (in reference to ISO 7816 interface)
iUICC	Integrated UICC: TRE within a System On Chip hosting a firmware obeying to ETSI UICC standards or GSMA RSP specifications
JHAS	JIL Hardware-related Attacks Subgroup
JIL	Joint Interpretation Library
M2M	Machine to Machine
MMU	Memory Management Unit
MNO	Mobile Network Operator
MTU	Maximum Transmission Unit
MVNO	Mobile Virtual Network Operator
NFC	Near Field Communication (also called contactless)
NVM	Non-Volatile Memory

Abbreviation or Term	Description
OEM	Original Equipment Manufacturer
PBL	Primary Boot Loader - a generic/open firmware loader allowing the remote provisioning of an agnostic firmware into a Primary Platform. The provisioning can concern a part of an existing firmware (firmware upgrade) or the whole firmware (firmware loading)
PBL agent	A trusted program running in the device and managing the communication with the IDS and the transfer of the image in the TRE
PME	Physical Memory
PN	Part Number of the TRE
PoC	Proof of Concept implementation
PRE	Physical Register
Primary Platform	A specified set of hardware and software functions within a TRE. This set is agnostic and independent of the firmware and the use cases. The Primary Platform embeds PBL. See [4].
Profile	A set of data and applets related to a GSMA RSP subscription
RFU	Reserved for Future Use
rNVM	Remote Non-Volatile Memory located outside the TRE perimeter
RNG	Random Number Generator
RO	Read Only
RSP	Remote SIM Provisioning
RW	Read Write
SIP	System in Package
SE	Secure Element
smart card	A removable TRC having a specified form factor
SoC	System On Chip: a baseband, an application processor, …
SSD	Security Scheme Descriptor: container for the keys encrypting the firmware
SR	Security Requirement
Subscription Management	The principles and protocols for managing a subscription within an eUICC
stream cipher	A stream cipher is a symmetric key cipher wherein plaintext digits are combined with a pseudorandom cipher digit stream (keystream). The stream can be a flow of bits or blocks
TCO	Total Cost of Ownership
ToE	Target of Evaluation
TPM	Trusted Platform Module
TRC	Tamper Resistant Chip: a standalone Integrated Circuit hosting a TRE
TRE	Tamper Resistant Element: a silicon enclave within a TRC or a SoC that supports the security and tamper resistance requirements for a Primary Platform

Abbreviation or Term	Description
UICC	A removable TRC hosting a firmware that obeys the ETSI UICC standards
VAN	Vulnerability analysis level. Tamper Resistance Grade as defined in [2]
VAS	Value Added Service
VME	Virtual MEmory: a mapping of a Physical Memory
VPP	Virtual Primary Platform
VPP Application	Application running on top of VPP
VRE	Virtual REgistry of peripherals: a mapping of the physical registers (for managed peripheral access)

1.4 References

Ref	External Reference	Description
[1]	ETSI TS 102 622	Smart Cards; UICC – Contactless Front-end (CLF) interface; Host Controller Interface (HCI) (Release 7.3 or higher)
[2]	Joint Interpretation Library	Application of Attack Potential to Smartcards, v2.9, Jan 2013
[3]	Strategy Analytics	Flash memory: eMMC vs UFS – Technology Comparison and Vendor Profiles Sept 2015
[4]	GSMA iUICC PoC group	Use cases for the iUICC
[5]	BSI-CC-PP-0084-2014	Security IC Platform BSI Protection Profile 2014 with Augmentation Packages.
[6]	Remote BIST	Elena Dubrova, Mats Näslund, Gunnar Carlsson, John Fornehed, Ben Smeets. *Two Countermeasures Against Hardware Trojans Exploiting Non-Zero Aliasing Probability of BIST.* DOI 10.1007/s11265-016-1127-4. Springer. J Sign Process Syst. 2016

2 Concept Overview

In order to formalize the terminology used within the subsequent descriptions, we consider the following high level concept.

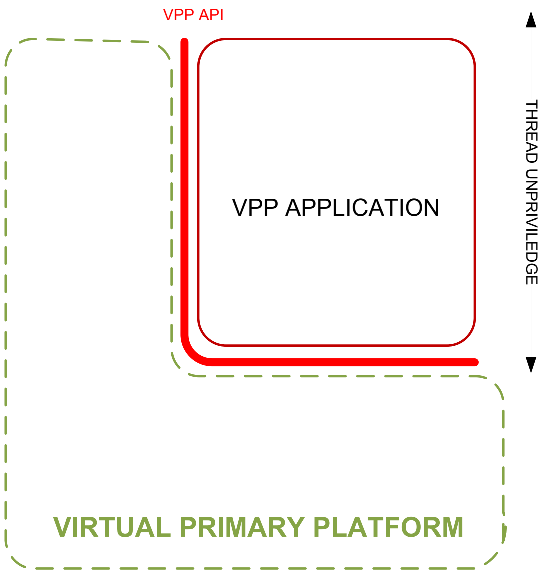


Figure 1: Global Concept

Error! Reference source not found. illustrates 3 parts:

- An **API**: interfacing the Virtual Primary Platform in order to expose a generic Primary Platform to the VPP Application,
- The **Virtual Primary Platform**: encapsulating a specific hardware Primary Platform and offering the lowest level of virtualization of the non-critical resources via an API, such that the VPP is independent of the use cases, and
- The **VPP Application**: an application of the Virtual Primary Platform; a VPP Application can be offered either by the Primary Platform provider as system VPP Application or by a third party. The VPP Application is use case dependent.

The objective of the concept is to meet a tradeoff:

- To make the VPP Application as independent as possible of the Primary Platform
- To provide at least the same level of flexibility for supporting all use cases as the legacy smart card platforms
- To ease the certification of the Primary Platform
- To ease the certification of the composite result (i.e. Primary Platform and third party firmware)
- To reduce the engineering efforts for adapting the VPP Application to the specificity of the Primary Platform.
- For the resulting Primary Platform, to provide a level of confidence equivalent or greater than the legacy Primary Platform (see appendixes).

3 Use Cases

The GSMA iUICC PoC group has drafted a use case document [4] which is used as a reference for this chapter. Some use cases have been extended and are given as examples. The use cases assume the TRE is included within a SoC targeting a device for the consumer market (e.g. smart phones, tablets, smart watches, connected cars…) and do not cover low footprint IoT devices.

3.1 eUICC Firmware 1

Vanilla Tel has deployed a generic subscription management offer based on GSMA RSP version X Phase Y infrastructure, and wants now to upgrade its subscription management system to GSMA RSP version X Phase Y+1, requiring changes in the eUICC OS.

Vanilla Tel wants to manage a homogenous installed base of eUICC and therefore upgrades the installed base by loading (through the PBL) a new eUICC image containing profile(s) of its customer.

Vanilla Tel offers to its customer the possibility to subscribe to a temporary service for getting access to Banana Tel (a local operator during a journey). Banana Tel will install its profile into the Vanilla Tel eUICC VPP Application using GSMA RSP version X Phase Y+1.

3.2 eUICC Firmware 2

Vanilla Tel is willing to support the GSMA RSP subscription ecosystem release X phase Y and sustains existing UICC-based custom services.

Vanilla Tel will load via PBL, a specific eUICC image containing custom features and profile(s) of its customer.

The Vanilla Tel customer subscribes to a temporary service for getting access to Banana Tel, a local operator during a journey. Banana Tel will install its profile into the Vanilla Tel eUICC VPP Application using the GSMA RSP mechanism.

3.3 eUICC Firmware 3

Vanilla Tel supports the GSMA RSP subscription ecosystem. Vanilla Tel hosts a MVNO called Banana Tel, which has its own set of UICC-based services.

Thanks to the PBL 2 images, one for Vanilla Tel and one for Banana Tel, are loaded onto the UICC.

The Vanilla Tel and Banana Tel operate on different eUICC VPP Applications sharing sequentially the same Virtual Primary Platform (as only one VPP Application is executed at a given point of time) while ensuring the continuity of service of their customers.

3.4 UICC Firmware 4

Vanilla Tel is a conglomerate of operators in multiple countries and wants to sustain the continuity of custom UICC-based services, which are differentiated by country. Using the PBL technology, Vanilla Tel will load custom UICC images that are functionally equivalent to its legacy removable custom UICCs and compatible with its legacy infrastructure.

3.5 Payment Firmware (VAS)

Vanilla Tel, Banana Tel and Zarkistan Bank Ltd want to deploy independent payment solutions using different technologies (QR-Code, EMV NFC, and Banana Pay). Each of the solutions asks for different certification schemes and business agreements. Each VAS can share the Virtual Primary Platform sequentially, independently, and without any interference by any other VAS. According to a contextual change led by the detection of the required payment scheme and the user's directive, the current VPP Application is backed-up off from the Virtual Primary Platform and the selected VPP Application is restored into that same Virtual Primary Platform.

3.6 Secure Bootstrap (VAS)

Genesis is an OEM which is providing a powerful tablet to Vanilla Tel. At the power ON of the tablet, the VPP Application related to secure bootstrap of the Sock powering the tablet is loaded into the Virtual Primary Platform. This TPM-like VPP Application provides a root of trust for allowing the secure hierarchical bootstrapping of the critical software components of the Sock. Once the secure bootstrapping is completed, then the Virtual Primary Platform can serve any other use cases (such as loading the Vanilla Tel VPP Application).

3.7 Content Protection (VAS)

Vanilla Tel targets deployment of a music steaming service using its IMS (IP Multimedia Subsystem) infrastructure. The music streams are encrypted for a given device. Vanilla Tel loads via PBL an image with a content protection (DRM) algorithm into the Tamper Resistant Element and then the application is activated when the user wants to enjoy the music service.

3.8 Driver Licence (VAS)

The government of Zarkistan aims to deploy virtual driver licenses on mobile phones with a trusted User Interface of the device. The Zarkistan government deploys a Virtual Driver License application which benefits from the biometric sensors and the TRE base ecosystem to interface to the TRE (Tamper Resistant Element). The Virtual Driver License mobile app triggers the loading, via the PBL agent, of the Driver License VPP Application image into the TRE. The Driver License VPP Application can be activated according to a contextual stimulus (e.g. car engine start, police control …) by using a short range wireless interface.

3.9 Home Automation (VAS)

Clementine Security Ltd promotes in combination with Vanilla Tel, an offer for a turnkey home automation solution, relying on a LTE-powered gateway embedded on iUICC. Clementine Security deploys a Virtual Home application on the gateway which benefits from the strong security provided by the TRE included in the gateway. An image developed by Clementine Security ensures that the PBL loads the security function into the TRE, and runs in parallel with the UICC VPP Application of Vanilla Tel. Clementine Security customers can securely access their home automation system using state-of-the art security principles.

3.10 IOT Use Case (VAS)

RisingStar Tel targets deployment of a set of sensors on rails using cellular technology (LTE-MTC). The sensors will report periodically the temperature of the rails to a data collection server.

The operator aims to provide the authenticity of the device to the railways company and therefore loads the secure firmware in the TRE. The sensor is fully blanked when installed and remotely provisioned with the firmware, including the network access credentials and dedicated functions related to the railways company.

3.11 Virtual Car Key (VAS)

SuperCar Ltd targets deployment of a car entry system using the smart phone of its customers, using Vanilla Tel cellular network. The SuperCar customer loads the virtual car key application from the app store. This keyless application bootstraps the loading of the Car Access cryptographic algorithm image into the Tamper Resistant Element. SuperCar Ltd can deploy custom and highly diversified keyless entry systems without compromising the overall security of the system.

3.12 Logical Access Control (VAS)

Vanilla Tel aims at deploying a virtual private network (VPN) for its employees. The Vanilla Tel employees load the VPN app from the app store. The keyless application bootstraps the loading of the VPN secure engine image (which requires higher processing power than what a smart card can provide) into the TRE. Therefore, Vanilla Tel can deploy this custom and highly secure VPN to its employees, while maintaining independence of other secure services.

4 Primary Platform Requirements

4.1 Primary Platform Economical Requirements

4.1.1 Initial Objective

The Tamper Resistant Element aims at being integrated within the same die of a System on Chip (SoC) for targeting a better integration and for offering an optimal and cost effective solution in increasing the performances without downgrading the security as compared to legacy UICC solutions.

Ref	Requirement	Rationale	SR[1]
PLT_1	The Primary Platform shall be independent of the functionality conveyed by the VPP Application	The Primary Platform is generic and independent of the hosted VPP Application. The VPP can be shared between multiple VPP Applications supporting various use cases. The TCO of the TRE will be reduced proportionally with the number of supported use cases.	

4.2 Primary Platform Functional Requirements

4.2.1 Core Platform Requirements

Ref	Requirement	Rationale	SR[2]
PF_1	The primary platform shall provide a solution for supporting Forward Compliance, regarding to the size of firmwares.	The Forward Compliance is the capability to support future needs within the limit of its external resources.	
PF_2	The Primary Platform shall provide a PBL VPP Application.	The image loader (i.e. Primary Boot Loader) shall be provided with the Primary Platform (as it cannot be loaded) irrespective of the image	

[1] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

[2] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

Ref	Requirement	Rationale	SR[2]
		loader technology (see definition of PBL in Section 2). This requirement is a pre-requisite of the iUICC GSMA PoC	
PF_3	The Primary Platform shall support the privacy by design rules.	The Primary Platform and its abstraction the Virtual Primary Platform may exist among multiple independent firmware issuers. Then, for example, any sharing of large constants (e.g. public key, certificates, serial numbers…) could lead to data correlation and privacy protection infringement (see European mandate M/530).	
PF_4	The Primary Platform shall manage multiple VPP Applications addressing different use cases from different issuers.	We benefit from the fast swapping between VPP Applications for virtual emulation of multiple virtual secure elements by sharing a common TRE: ▪ Lower Total Cost of Ownership for the VPP Application issuers ▪ No security compromise ▪ High performances By independently supporting multiple VPP Applications, we can support any combination of VPP Applications, hosting a single or multiple use cases.	
PF_5	The Primary Platform shall provide a means for running no more than a single VPP Application at a given time.	Some resources are not shareable. Thus, the VPP enforces that only a single VPP Application can run at a given time, leading to sequential execution of the VPP Applications. The communication between VPP Applications is not defined and is dependent upon the VPP Application. Secure mailbox or shared files outside the VPP Application can be accessible via the communication interfaces of the VPP. The definition of such means for exchanging application data between VPP Application is outside the scope of the Primary Platform (which is use case independent)	
PF_6	The Primary Platform shall provide a means for avoiding dependency of the VPP Application with respect to the execution memory addressing.	Flat addressing is required in order to ease the porting of legacy software, to ease the design of new VPP Applications, and to improve performance. This requirement enforces the use of modern memory management (e.g. MMU versus overlay) and interoperability between Virtual Primary Platform implementations. (Editor's note: 99% of the application/OS which have been deployed by the smart card vendors irrespective of the use cases are using Primary Platform based on a flat memory addressing (no overlay). In order to guarantee the continuity of services, smart card vendors cannot be practically expected to rewrite their OS and Java Card VM for supporting a non-flat addressing.)	
PF_7	The Primary Platform shall provide a means for	The power cut of the Primary Platform can occur any time. The management of the data transfer	

Ref	Requirement	Rationale	SR[2]
	preventing the corruption of content against power on/off cycles.	from the secure RAM to an rNVM is under the responsibility of the Primary Platform. The VPP API has to offer a means for performing synchronization between the SRAM and the rNVM	
PF_8	The Primary Platform shall provide a means for managing the access of the data to/from the remote NVM in order to prevent dependencies on the remote NVM implementation.	The VPP Application accesses only to a virtual memory. The VPP provides a means for interfacing to the rNVM according to the architecture selected by the SoC maker. The VPP Application operation shall be independent of the Primary Platform and its surrounding technology for managing the rNVM.	
PF_9	The Primary Platform shall provide means for protecting its data to/from remote NVM (Non Volatile Memory) outside the Tamper Resistant Element (TRE). These means shall be equivalent to those which are applied to the internal memory of the TRE, along the life cycle of the device hosting the SoC. Supported protections shall include confidentiality, anti-replay/roll-back, integrity, Perfect Forward Secrecy.	Operation of the VPP Application while running and while idle shall have protections which are effectively equivalent to those of a running/idle application on a dedicated TRE.	
PF_10	The Primary Platform should provide a means to manage DoS attacks against its data to/from the remote NVM, and to do so in a manner that is effectively equivalent with non-integrated TRE behavior.	The Primary Platform cannot control what takes place outside the TRE perimeter, but it shall detect corruptions of data which are stored outside that perimeter and take actions which are consistent with those actions that are taken when a TRE detects attacks on its internal memories. For example, this means can provide DoS protection against a non-secure processor also residing on the SoC.	
PF_11	The Primary Platform shall provide a means to protect the keys which are used by the crypto engine or the stream cipher.	When the VPP Application needs direct access to the crypto processor, then it shall be managed according to the state of the art. The VPP needs to access private cryptography for decrypting/encrypting/authenticating the data to/from the rNVM. It shall be managed according to the state of the art.	
PF_12	The VPP shall provide a means to notify the VPP Application when an exception occurs	Operation of the VPP Application shall receive notifications which are effectively equivalent to those of an operating application on a dedicated TRE. As with a dedicated TRE, exceptions may be internally generated (e.g., access rights infringement) or externally generated (e.g., hardware attack detection).	
PF_13	The VPP shall provide a means for system VPP application to access additional privileges to access system specific capability.	The Primary Platform maker may design VPP Application extended rights for its needs. PBL is such a system VPP Application for which an extended API is available. Nevertheless, the VPP shall handle the VPP Application as a	

Ref	Requirement	Rationale	SR[2]
		system VPP Application in order to grant additional rights	
PF_14	The VPP shall provide a means to notify the VPP Application when a shutdown request occurs.	For example, the VPP shall support delivery of requests to the VPP Application to properly complete an ongoing transaction. However, in case of a “dirty” removal (e.g. battery removal) this requirement will not apply. The interrupted application will be notified upon next power cycle.	
PF_15	The VPP shall provide a means for the VPP Application to notify the VPP when the VPP Application is ready to be shutdown.		
PF_16	The Primary Platform shall provide a µkernel.	Legacy firmware may be designed for using a µkernel. In order to ease the porting of these legacy applications to VPP Applications, the VPP shall offer a µkernel support, because it cannot be emulated in thread mode. Legacy firmware that does not use a µkernel has the possibility to encapsulate the resulting porting in a single thread.	
PF_17	The VPP shall offer collaborative scheduling for VPP Application threads.	The thread scheduling related to the threads of the VPP Application shall be controlled by the VPP Application in order to prevent unexpected breaks of processes, e.g., for cryptography operations.	

4.2.2 Access Rights Requirements

Ref	Requirement	Rationale	SR[3]
MK_1	The Primary Platform shall provide means, in kernel mode, to prohibit the access to the content of the memory used in thread mode.	Protection against fault attacks in kernel mode. Full insulation between the kernel and thread mode for the VPP threads and the VPP Application threads. See annexes and Figure 19.	
MK_2	The Primary Platform shall provide means for protecting the access to the VME (Virtual Memory) and to VRE (Virtual Register of peripherals) in thread mode.	All data exchanges shall be protected. Protection against fault attacks and uncontrolled data leaks	
MK_3	The Primary Platform shall provide a means to protect against access violations from hardware automated	DMA: Direct Memory Access	

[3] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

Ref	Requirement	Rationale	SR[3]
	memory transfer (e.g. a DMA).		
MK_4	The Primary Platform shall provide a means to generate an exception if a virtual memory area is shared with two threads without mutual consent for both of them.	Hierarchy of rights enforcement	
MK_5	The Primary Platform shall provide a means to generate an exception if multiple threads grant read/write access to the same shared memory area.	A shared memory between multiple threads shall have a single thread granting the RW access.	
MK_6	The Primary Platform shall provide an efficient means for making the content in a physical RAM page unusable by other parties before reallocation of the page to a virtual memory.	Data leak protection between threads	
MK_7	The µkernel shall provide an API for triggering the backup of a virtual memory area into the rNVM.	Synchronization support for Java Card transaction (commit/rollback). This feature is needed due to the nature of the TRE memory (RAM) and power off/on cycle issues.	
MK_8	The µkernel shall provide a means to enforce hierarchy of access rights when a thread forks.	Hierarchy of rights enforcement. Following requirements explicitly add more specific requirements for this.	
MK_9	The µkernel shall provide a means to ensure VPP can only create a single VPP Application thread.	VPP bootstraps a single application thread, and then it is up to this first VPP Application thread to fork other threads.	
MK_10	The µkernel shall provide a means to generate an exception if a VPP Application thread attempts to fork a VPP thread.	A VPP thread can fork a single VPP Application thread but a VPP Application cannot fork any VPP thread.	
MK_11	The µkernel shall provide a means to prevent a VPP thread from gaining access rights to the crypto processor or the RNG peripherals.	The VPP provides generic facilities to the VPP Application; the VPP Application is liable for its credentials. Thus, the handling of these credentials cannot be delegated to the VPP. The µkernel is liable of the MMU settings.	

Ref	Requirement	Rationale	SR[3]
MK_12	The Primary Platform shall provide a means to guarantee to the VPP Application that the cryptographic values are hidden from the VPP. Cryptographic values include input/output data for both the crypto processor and the RNG operations.	The VPP provides generic facilities to the VPP Applications; the VPP is liable of the management of all peripherals except the crypto processor and the RNG.	
MK_13	The µkernel shall be minimal and shall be designed to enforce the implementation of peripheral drivers in thread mode.	Protection against fault attacks in kernel mode. We enforce here the pure definition of a µkernel.	
MK_14	The µkernel shall have a level of security certification (i.e. Common Criteria) higher or equal to the highest level of certification required by the Primary Platform.	Usually a µkernel has the highest level of certification and allows designing less critical software components with a lower level of certification. This is the main difference between : • A monolithic kernel including all drivers and forcing the Primary Platform maker to apply the same level of certification for all software components of its Primary Platform to support. A µkernel approach where the software components may have different level of certification leading to less engineering efforts.	
MK_15	The µkernel designer shall provide a means to guarantee the µkernel support only the required functionalities.	Certification process Software Trojan horse guarantee/audit	
MK_16	The Primary Platform shall provide a means to dispatch the crypto processor interfaces to multiple VPP Application threads.	Enforcement /segmentation of the roles (security)	
MK_17	The µkernel shall provide a means to a thread to downgrade its rights.	Accesses to VME and VRE are controlled, and may be further self-restricted by a thread	

4.2.3 Primary Platform Certification Requirements

Ref	Requirement	Rationale	SR[4]
PS_1	The certification of the Primary Platform shall claim in	State of the art tamper resistant legacy products	

[4] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

Ref	Requirement	Rationale	SR[4]
	its Security Target the conformance with Protection Profile BSI-CC-PP-0084-2014 [5] including Loader package 2.		
PS_2	The Security Target of a Primary Platform instance shall cover all the security requirements of this document.	The iUICC Primary Platform requirements is a superset of the PP-0084	
PS_3	The certification minimum assurance level is EAL4 augmented with AVA_VAN.5 and ALC_DVS.2	This is the minimum level to claim BSI-CC-PP-0084-2014	
PS_4	AVA_VAN.5 tests shall be performed in accordance with the JIL Application of Attack potential to Smartcards documentation [2]		

4.2.4 Virtual Primary Platform functional requirements for communications

Ref	Requirement	Rationale	SR[5]
VPPF_1	The VPP shall provide to the VPP Application means for conveying HCP packets and for getting the MTU of the link layer that is conveying the HCP packets.	Continuity of NFC services in the legacy UICC. Moreover The TS102.622 offers a migration path for supporting legacy APDUs.	
VPPF_2	The VPP shall provide to the VPP Application a means for getting the size of incoming packets.		
VPPF_3	The data link layer conveying an HCP packet shall be able to operate without information on the context related to its upper layer (HCI).	The HCP sub-layer should not perform any parsing of the HCP packet and in any case, it shall not deduce a behavior according to parsing information.	
VPPF_4	The HCP sub-layer shall be balanced.	There is neither master nor slave.	
VPPF_5	The VPP shall provide to the VPP Application a means to		

[5] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

Ref	Requirement	Rationale	SR[5]
	interface a FIFO of incoming HCP packets.		
VPPF_6	The VPP shall provide to the VPP Application a means to interface a FIFO of outgoing HCP packets.		
VPPF_7	The VPP shall provide to the VPP Application a means for getting the number of HCP packets in the FIFO of incoming packets.		
VPPF_8	The VPP shall provide to the VPP Application a notification when the FIFO for incoming packet is no longer empty.		
VPPF_9	The VPP shall provide to the VPP Application a means for getting the number of remaining HCP packets in the FIFO for outgoing packets before its saturation.	Data flow management.	

4.2.5 Primary Platform Remote Auditing Requirements

Ref	Requirement	Rationale	SR[6]
PT_1	The Primary Platform shall provide a means for prohibiting the access to long term secrets in test mode.		
PT_2	The Primary Platform shall provide a means for switching from operational to test mode after the issuance of the SOC in-field.		
PT_3	The Primary Platform shall support a Hardware Built-In	The use of Built-In Self Test[7] (BIST) techniques with external challenge does not target the	

[6] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

[7] BIST techniques are classified in a number of ways, but two common classifications of BIST are the Logic BIST (LBIST) and the Memory BIST (MBIST). LBIST, which is designed for testing random logic, typically employs a pseudo-random pattern generator (PRPG) to generate input patterns that are applied to the device's internal scan chain, and a multiple input signature register (MISR) for obtaining the response of the device to these test input patterns. An incorrect MISR output indicates a defect in the device or a fraud (hardware Trojan Horses).

Ref	Requirement	Rationale	SR[6]
	Self Test mode with challenge.	manufacturing testing but does target a remote checking in the field about the authenticity of the chip design against hardware Trojan horses and untraced hardware design changes.	
PT_4	The Primary Platform shall provide a means for a VPP Application to confidentially inject a challenge for the hardware Built-In Self Test.	In order to check the authenticity of a design in the field, this last one can be unpredictably stimulated with a referenced challenge leading to a predictable signature.	
PT_5	The Primary Platform shall provide a means to retrieve the Built-In Self Test result from a VPP Application		
PT_6	The Primary Platform maker shall provide a means for a design audit.	Certified labs shall be able to evaluate the compliance of the Primary Platform design with the security design rules. An audit against Hardware Trojan horse shall be possible.	
PT_7	The Primary Platform shall provide a means to a VPP Application to challenge the Primary Platform against basic hardware and software modifications by comparing them to those that are submitted to certification labs.	This is a generic and global requirement to cover the needs for a dynamic remote audit of the Primary Platform against basic hardware and software changes without incremental certifications (see [6] for more details)	

4.2.6 System on Chip Requirements

Ref	Requirement	Rationale	SR[8]
SOC_1	The SoC may embed its NVM memory controller.		
SOC_2	The TRE may have a direct and priority access to the remote NVM (rNVM) controller.		
SOC_3	The rNVM controller may manage a private logical partition with rNVM.		
SOC_4	The size of the logical private partition may be dynamically managed by rNVM controller.		

[8] Note, where there is a bullet in the Security Requirement column this indicates the requirement is a security requirement.

Ref	Requirement	Rationale	SR[8]
SOC_5	The SOC design may provide a means to prevent the access to the private logical partition to any cores of the SoC except the TRE.	Replay attack protection and DoS protection	
SOC_6	The rNVM may be integrated in the same package with the SoC (f.i. SIP or ePOP).	Prevent physical attacks.	
SOC_7	The SoC may provide a means to ensure the highest priority for TRE memory transactions about the rNVM controller queuing.	Performance and DoS protection	
SOC_8	The rNVM controller may be included in the ToE.	Replay attack and DoS protection	
SOC_9	The SoC maker may provide a means to integrate an obfuscated TRE.	Hardware Trojan horse protection	
SOC_10	The rNVM or the iNVM shall expose an endurance greater than 500 K erasing cycles.		

5 Informative Appendices

5.1 The Legacy Primary Platform

The legacy Primary Platform provides essential hardware functions such that the legacy vendors of firmware have the full knowledge and control of the hardware resources of the Primary Platform. Only the low level driver related to the local Non Volatile Memory is provided for ensuring 3 functions:

- Write in NVM
- Erase the NVM
- Repair the NVM

Thus, the firmware provided by the legacy vendors is fully liable for all management of the hardware functions, the countermeasures against data/secrets leaks, and the communication to/from the TRE. Moreover, the Primary Platform software is very limited and has few or no access to the communication means that could lead to data/secrets leaks.

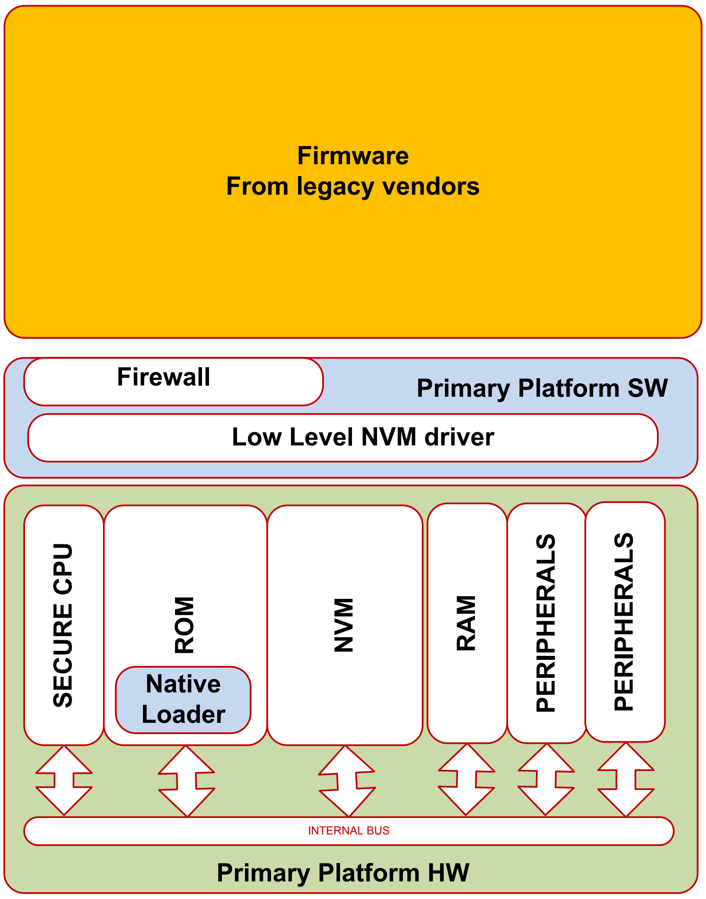


Figure 2: Legacy Primary Platform

The first objective of the requirements related to the VPP is to enforce the confidence of the VPP Application makers in the Primary Platform beyond the certification process.

The VPP concept delegates to the Primary Platform all use case independent facilities. Thus, the liability is split between the VPP and the VPP Application. This split approach requires a clear specification and control of the limits and risks in order to assign the liability (certification). Moreover the behavior and the interfaces of the VPP shall be guaranteed in order to ease the porting of the VPP Application across the VPPs from multiple TRE makers.

5.2 The Primary Platform

The Primary Platform is a combination of hardware and software functions which are specific and shall support a set of requirements. The Virtual Primary Platform is a Primary Platform as viewed from the VPP Application.

At the moment there are 2 different points of view, as represented by the figures below.

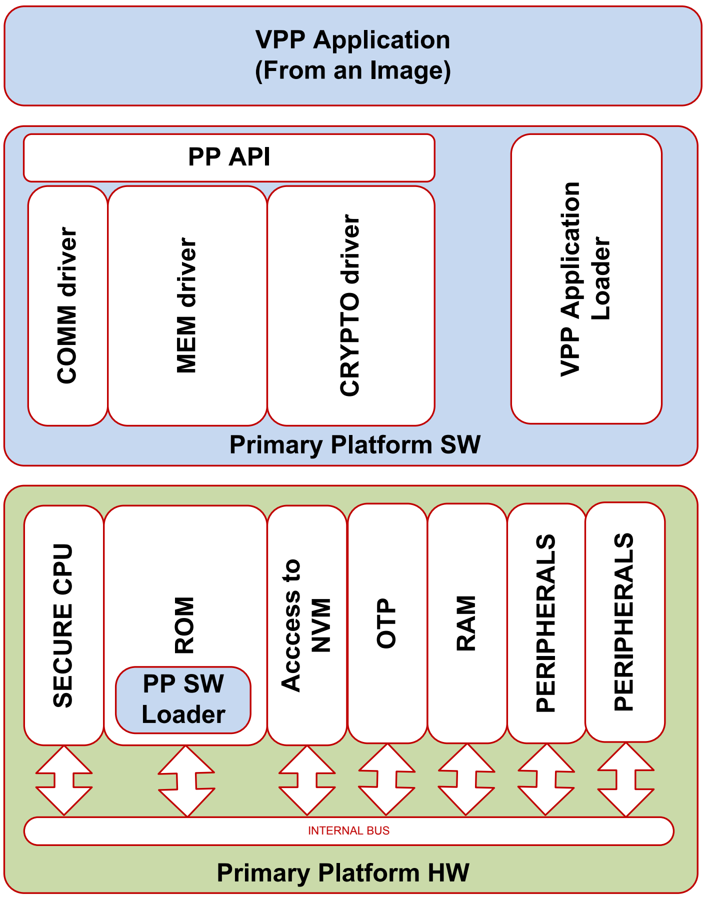


Figure 3: Architecture: Details about VPP and VPP Application – simple layered view

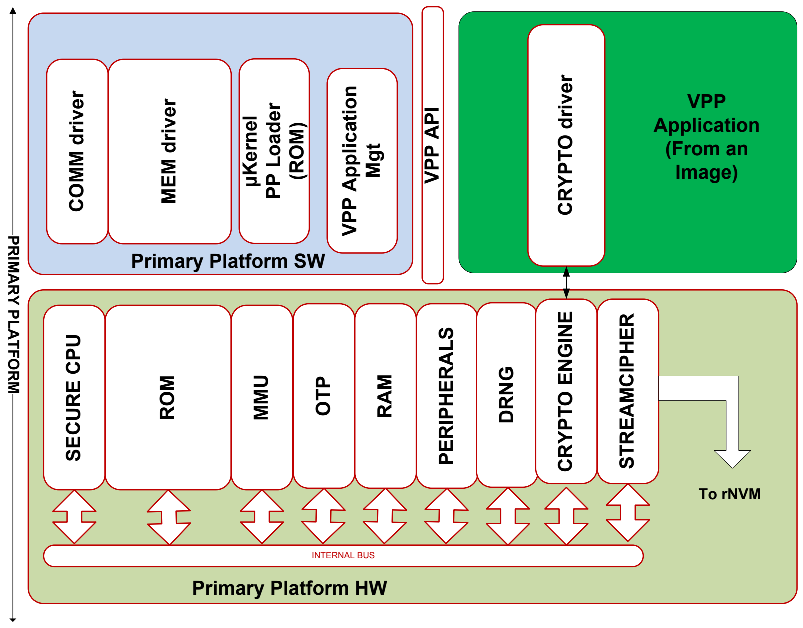


Figure 4: Architecture: Details about VPP and VPP Application – showing isolation of crypto engine

5.2.1 Rationale for moving the control of the crypto-processor/RNG within the VPP Application

Only a VPP Application can access to the Crypto Engine for the following reasons:

- The crypto-engine hardware is not shareable
- The cryptographic protocols and counter-measures are use-cases-dependent. Thus, the crypto engine driver cannot be in the VPP, which are use-cases-independent
- The VPP Application is liable for the credentials (e.g., keys). Thus, it cannot be delegated to the VPP without moving the liability.

The Primary Platform HW and SW certification is independent of the use case supported by the VPP Application.

5.3 The Primary Boot Loader (PBL)

The UICC Primary Boot Loader is itself a VPP Application of the Virtual Primary Platform with an additional specific API, able to perform the loading of an image following a standard format encapsulating a firmware (according to a specific security scheme) into the secure memory of a Tamper Resistant element.

Once the firmware is loaded, all subsequent operations are dependent upon the VPP, including the swapping of VPP Applications (i.e., backup then restore) or their deletion.

The image can contain the whole VPP Application (VPP Application loading) or a partial VPP Application (VPP Application update).

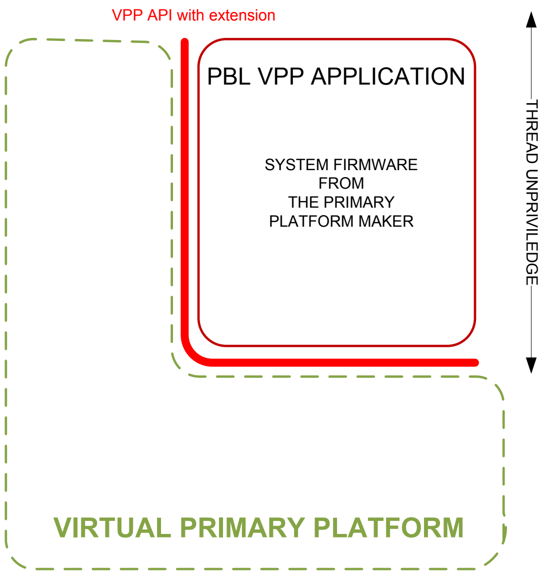


Figure 5: PBL as a VPP Application

5.3.1 PBL Image

The image contains a VPP Application which is extracted by the PBL VPP Application.

- The PBL image is a container encapsulating the binary data (i.e. VPP Application) and cryptographic material according to defined format and security scheme.
- The PBL image is generated by an image maker that provisions an Image Delivery Server. The IDS binds an image to a given TRE and then transfers it to the PBL agent running in the device. The PBL agent requests the TRE to load the PBL VPP Application into memory, and then transfers the image to the PBL VPP Application. The PBL VPP Application extracts and decrypts the encapsulated VPP Application, re-encrypts the VPP Application with private VPP keys, and stores the re-encrypted VPP Application into a remote NVM. Thus, that rNVM contains a collection of encrypted VPP Applications ready for execution.

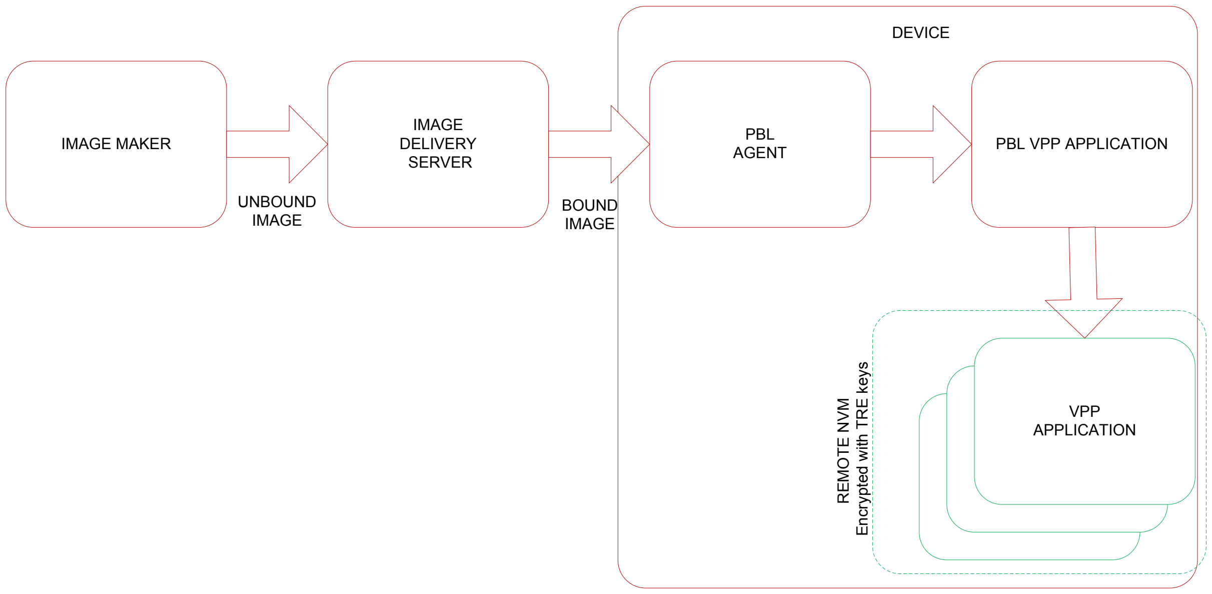


Figure 6: PBL Flow

- The PBL function, like any other functionality needing to access the crypto engine and the DRNG (Digital Random Generator), shall be hosted within a VPP Application.
- The PBL VPP Application shall be provided by the Primary Platform maker because it cannot be loaded from the IDS server (i.e., chicken and egg paradox).
- The PBL VPP Application cannot be deleted.
- For its own operations, the VPP shall use/emulate independent cryptographic means than those offered to the VPP Applications (such as for protecting data to/from the remote NVM).
- The PBL VPP Application is not in charge of management of the VPP Applications in RAM (such as backup/restore/deletion).

PBL is a native functionality of the Primary Platform for loading an image (which follows a standard format encapsulating a firmware according to a specific security scheme) to the secure memory of the TRE.

The image can contain the whole firmware (firmware loading) or a partial firmware (firmware update/upgrade).

5.4 PBL Binding Principle

Figure 7 illustrates the PBL image processing from the image maker generating the eUICC firmware image for the receiving Primary Platform.

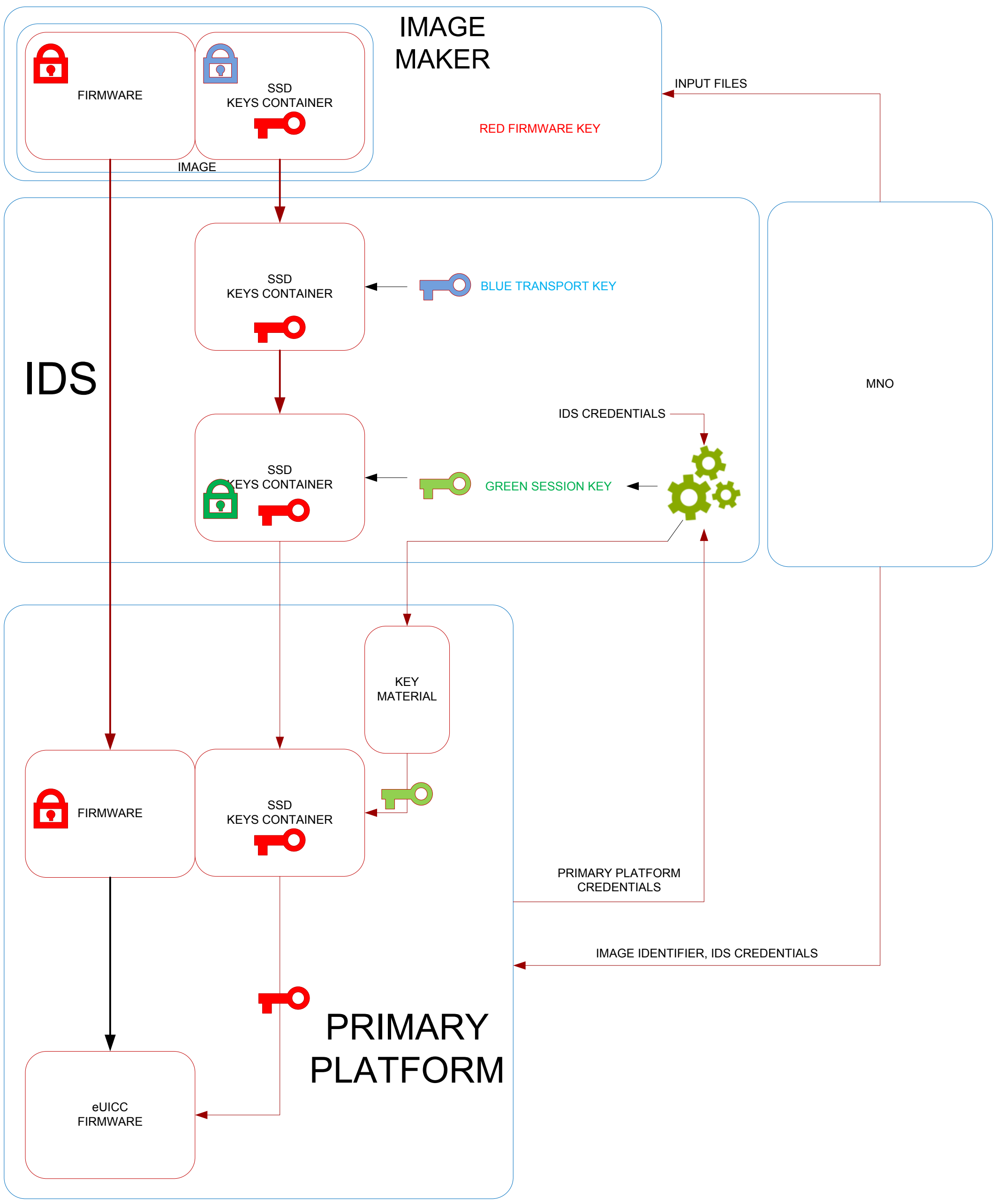


Figure 7: PBL image processing

The procedure is the following:

- Subsequent to a user request, the MNO pushes the image identifier and IDS credentials to the Primary Platform via the device and the PBL agent.
- The Primary Platform answers with its credentials to the IDS.

- The IDS gets the image from the image maker. Using a transport key (blue) that is shared with the image maker, the IDS decrypts the SSD key container to obtain the firmware key (red).
- The IDS computes a session key (green) using key material from both its credentials and also the Primary Platform credentials.
- The IDS encrypts the SSD (key container) with the session key (green), appends the key material with the resulting image and forwards all to the Primary Platform via the device and its PBL agent.
- The Primary Platform deduces the session key (green) from the key material and its own credentials, decrypts the SSD (key container), extracts the firmware key (red), and decrypts the TRE firmware.

5.5 Functional data flow of the PBL agent

Figure 8 illustrates the functional data flow between the TRE and the PBL agent.

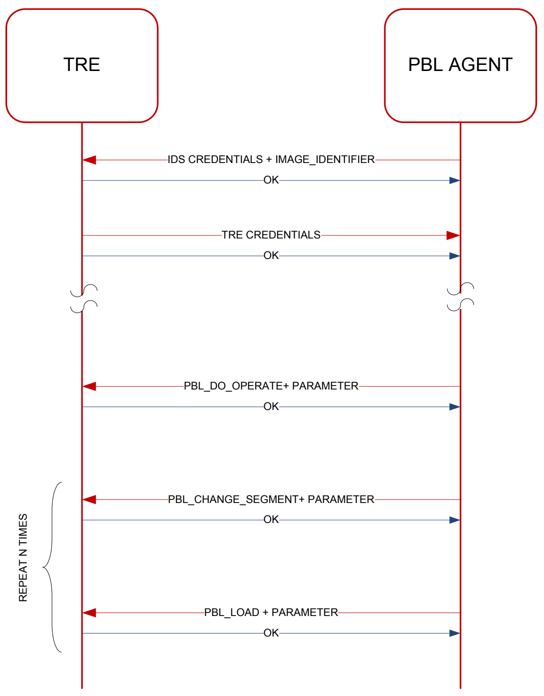

TRE
PBL AGENT
IDS CREDENTIALS + IMAGE_IDENTIFIER
OK
TRE CREDENTIALS
OK
PBL_DO_OPERATE+ PARAMETER
OK
PBL_CHANGE_SEGMENT+ PARAMETER
OK
REPEAT N TIMES
PBL_LOAD + PARAMETER
OK

Figure 8: PBL agent to TRE data flow

The PBL interface has 5 commands and 5 steps:

1. The PBL agent forwards the IDS credentials and the image identifier previously collected by any means (NFC NDEF, QR-CODE, Push, etc.)
2. The TRE sends back its credential, which will be forwarded to the IDS.
3. The PBL agent gets the image from the IDS, extracts the parameters and sends the PBL_DO_OPERATE commands and a parameter. The TRE deduces the session key from the key material within the parameter.
4. The PBL agent sends the PBL_CHANGE_SEGMENT and a parameter. The parameter contains the information for loading a segment of the firmware.

5. The PBL agent sends the PBL_LOAD and a parameter. The parameter contains the segment of the firmware to load. The parameter length matches the segment length to load.

Steps 4 and 5 are repeated until all firmware segments are successfully loaded. The number of segments is provisioned within the image and its manifest. The PBL agent has no need for parsing the image. The PBL agent is driven by the image manifest and agnostic about the security scheme or the internal image format.

An image may have no segment then only the PBL_DO_OPERATE command is sent to the TRE for the remote administration of the TRE.

5.5.1 Firmware Loading

There are no cryptographic operations to perform within the PBL agent and likewise no confidential keys to manage.

The steps 1 and 2 and the steps 3 to 5 can be performed at different life cycles of the device. Prior to the PBL loading a firmware, the PBL checks whether the TRE already hosts any firmware, and if so, the existing one is deleted or backed-up.

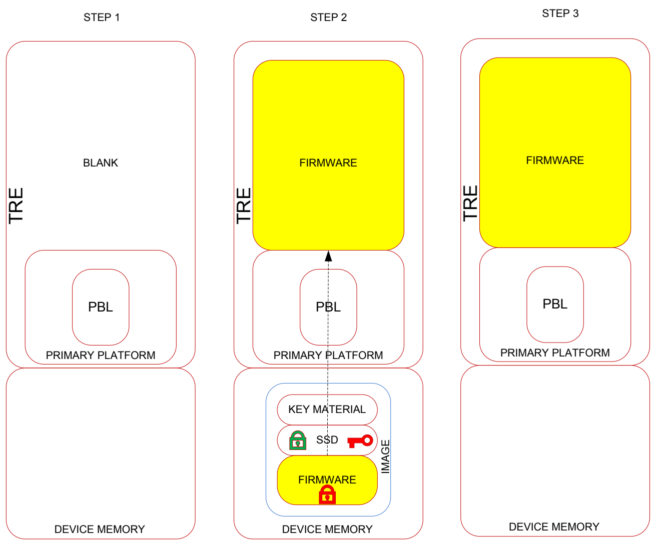


Figure 9: Loading

3 steps:

1. The TRE is blank or erased.
2. The image is downloaded into the device memory. The PBL agent running in the device feeds the PBL function that extracts the firmware from the image and installs it in the TRE.
3. The image in the device memory is no longer valid and should be deleted.

5.5.2 Firmware Update

Prior to the PBL updating a firmware, the TRE hosts the firmware to update in its memory. Then the PBL rewrites some parts of the said firmware.

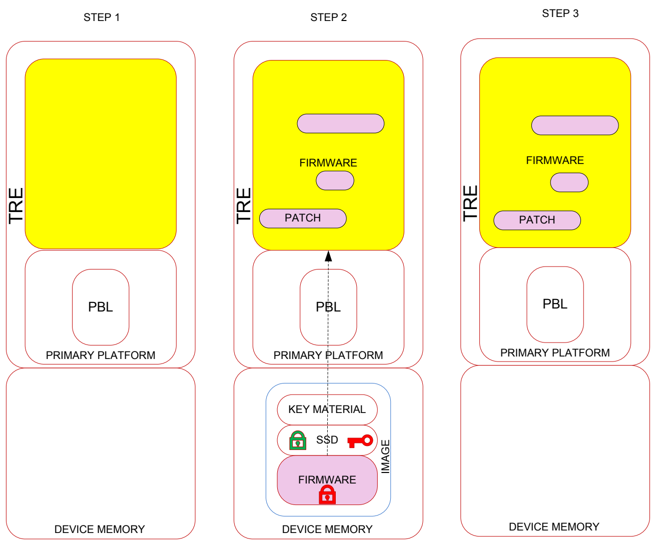


Figure 10: Firmware update

3 steps:

1. The TRE hosts a firmware (yellow)
2. The image is downloaded into the device memory. The PBL agent running in the device feeds the PBL function that extracts the firmware patches (violet) from the image and installs the patches into the current firmware in the TRE.
3. The image in the device memory is no longer valid and should be deleted.

Once the firmware is loaded/updated, all subsequent operations are dependent on the Primary Platform. The subsequent operations are mainly local and related to:

- The backup of the firmware from the TRE to the device. Subsequent to the backup, the firmware in the TRE is erased.
- The restore of the firmware from the device to the TRE. Prior the restore of the firmware into the TRE, the current firmware is backed-up or deleted.
- The deletion of the firmware in the TRE.

5.6 Backup/Restore/Deletion Procedures

5.6.1 Backup

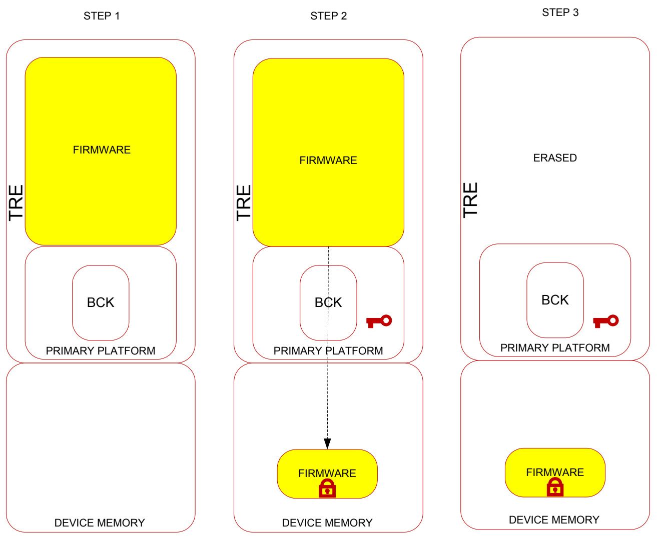


Figure 11: Backup Procedure

3 steps:

1. The TRE hosts a firmware (yellow).
2. Subsequent to a request from the device, the Primary Platform backs up the encrypted firmware into the device memory using an ephemeral key (violet).
3. The firmware in the TRE is erased.

The backup function in the Primary Platform is outside the scope of the PBL.

5.6.2 Restore

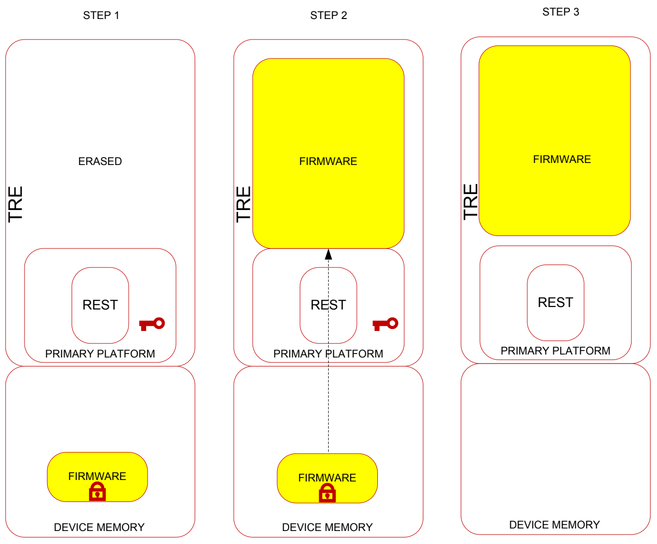


Figure 12: Restore Procedure

3 steps:

1. The TRE does not host any firmware (erased).
2. Subsequent to a request from the device, the Primary Platform restore the encrypted firmware into the TRE using an ephemeral key (violet).
3. The TRE hosts a firmware (yellow). The TRE destroys the ephemeral key (violet) and the encrypted firmware should be deleted from device memory.

The restore function in the Primary Platform is out of the scope of the PBL VPP Application.

5.6.3 Delete

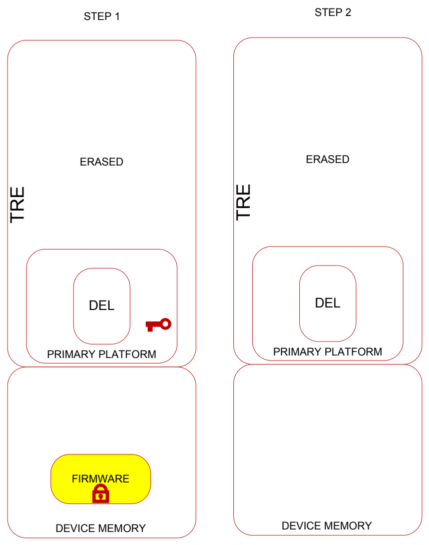


Figure 13: Deletion procedure

2 steps:

1. The encrypted firmware is in the device memory.
2. Subsequent to a request from the device, the Primary Platform destroys the ephemeral key and all internal data related to the firmware to delete, and the encrypted firmware is deleted from the device memory.

The delete function in the Primary Platform is in the scope of the PBL (remote deletion) and in the scope of the Primary Platform (local deletion).

5.7 System-on-Chip Architecture Proposals

This section describes some notional System on Chip architecture concepts that could be envisioned. Inclusion of the concepts herein does not imply that they necessarily meet the requirements in this document nor that they are acceptable from a security standpoint. Any proposed SoC concept would need to be validated against the requirements in this document before it could be considered to be a viable approach.

The following concepts do not presume about specific functions of the System on Chip or possible combinations of Systems on Chips.

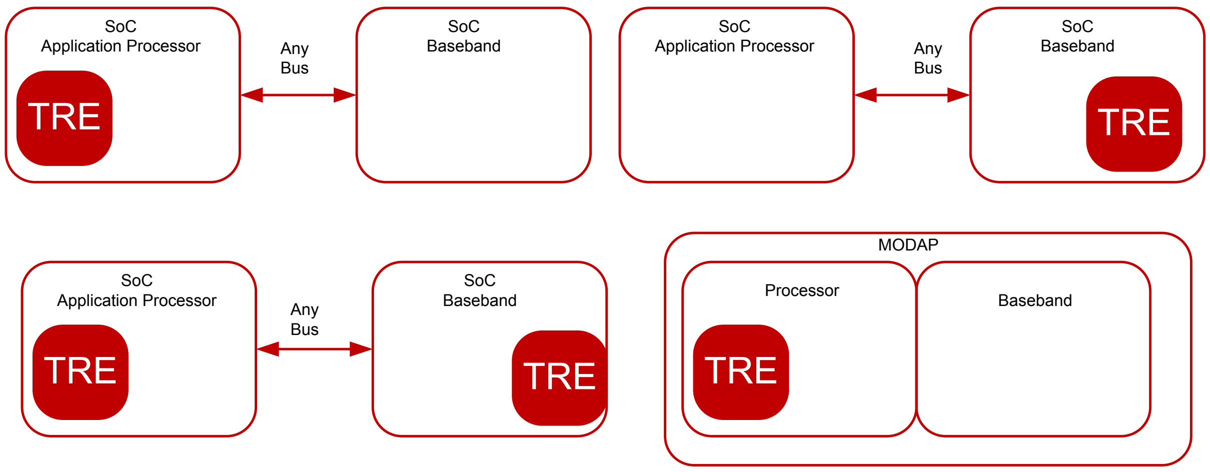


Figure 14: System on Chip concepts

The communication between Systems on Chips is out of the scope of this document.

The Primary Platform is essentially a hardware sub-system similar to a smart card or secure element with implementation alternatives when it comes to NVM.

- Internal NVM within the TRE
- External NVM to the SoC package
- In-package NVM
- External secure NVM

5.7.1 Internal MVN

This architecture integrates a TRE equivalent to a traditional smart card or SE within the SoC and this TRE includes NVM, and therefore it is named iNVM. The iNVM may contain data caching for easing the protection against unpredictable ON/OFF power cycles. It may also be used to support fast access to the most frequent VPP Applications, as well as the ephemeral and non-volatile keys, thereby avoiding the overhead of key pairing with the remote NVM (rNVM). The main storage of the VPP Applications is still in the remote NVM, in order to support Forward Compliance.

As an example, iNVM can use MRAM, CBRAM, RRAM or equivalent technology if sufficiently mature.

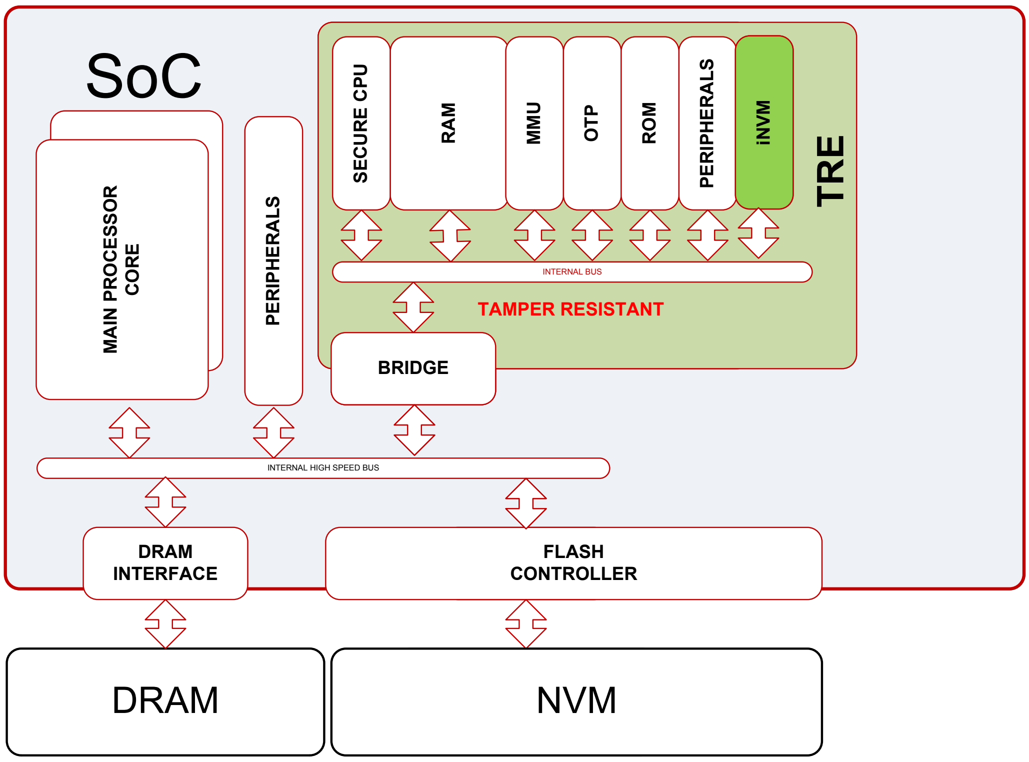


Figure 15: Internal NVM proposal

5.7.2 External MVN

This architecture leverages an external NVM and a limited amount of internal NVM. The limited amount of internal NVM shall be sufficient to operate an anti-rollback mechanism to protect the external NVM. This solution should provide a means (DoS protection) to manage the priority queuing to access to the external NVM. The Memory Controller shall provide a mean to encrypt the data stored in the external memory and may have a direct access to the FLASH controller.

The direct access to the (custom) Flash Controller may offer the following benefits:

- The TRE is autonomous and runs without the device OS assistance and then can be used for the Secure Boot VAS use case (as described before).
- It becomes easier to protect against DoS attack from the OS. (For Instance, such attacks can be based on overloading of the system bus or interference with the priority queuing of the FLASH controller.)

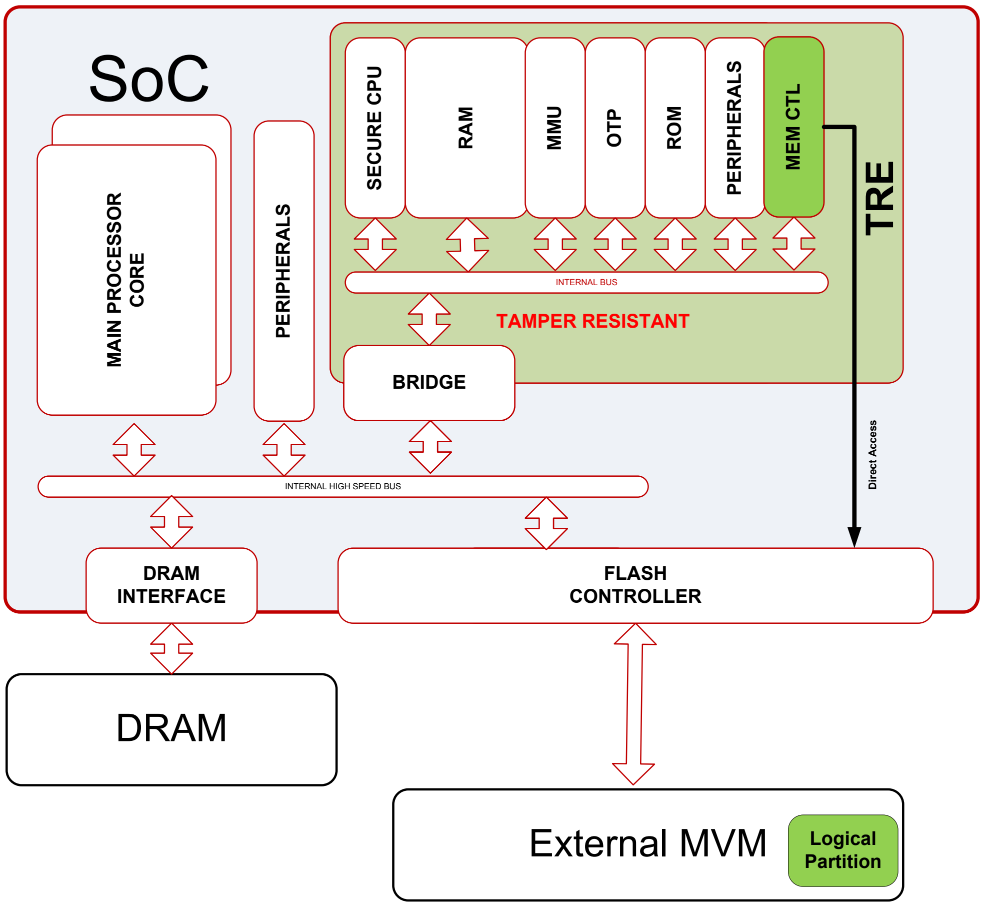


Figure 16: External NVM proposal

5.7.3 In Package NVM

This architecture leverages an in-package external NVM and a customized Flash Controller.

Figure 17 illustrates a possible architecture for a System on Chip (SoC), without the need for eNVM technology that supports large endurance cycles (>500K cycles).

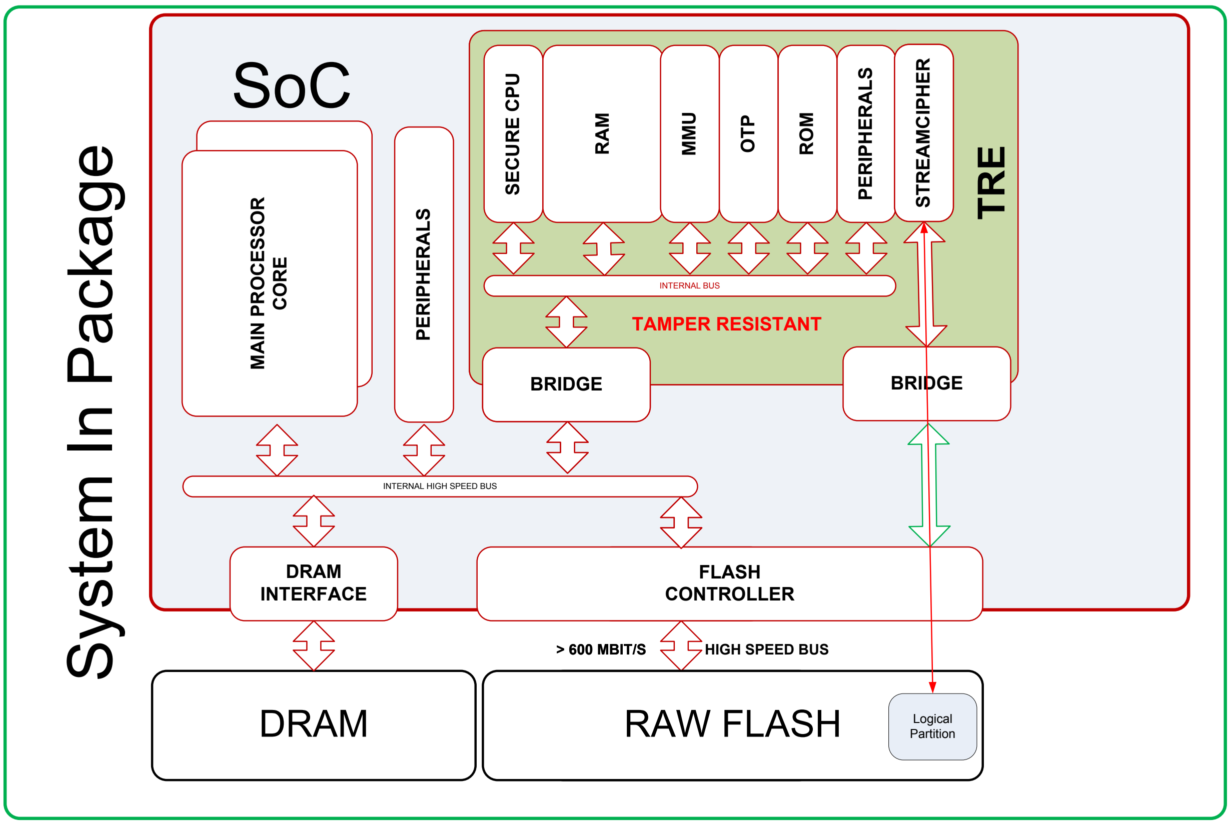


Figure 17: In-Package NVM proposal

The direct access to the Flash Controller may offer the following benefits:

- The TRE is autonomous and runs without the device OS assistance and therefore can be used for the Secure Boot VAS use case.
- It becomes easier to protect against DoS attack from the OS.
- Risk mitigation against NVM memory swapping by mandating a physical attack on the package.

Regarding certification, the FLASH controller is in the ToE.

The above architecture will mitigate the risks related to the non-support of the Perfect Forward Secrecy and the anti-replay and DoS protection properties.

5.7.4 External Secure NVM

This architecture leverages an external secure NVM.

There are 2 options for accessing the NVM, indirect (option1) or direct (option 2) see Figure 18.

This architecture offers the following advantages:

- The solution doesn't require the use of System in Package (SiP) Technology.
- The solution may offer (with option 2) a better protection against DoS and may support the Secure Boot VAS use case.

The solution suffers of the following drawbacks:

- The certification may involve a third party (secure memory maker) then an additional composite certification of the Primary Platform per secure memory maker.
- The solution requires a secure pairing between the SoC and the memory (which can be challenging within an open market).

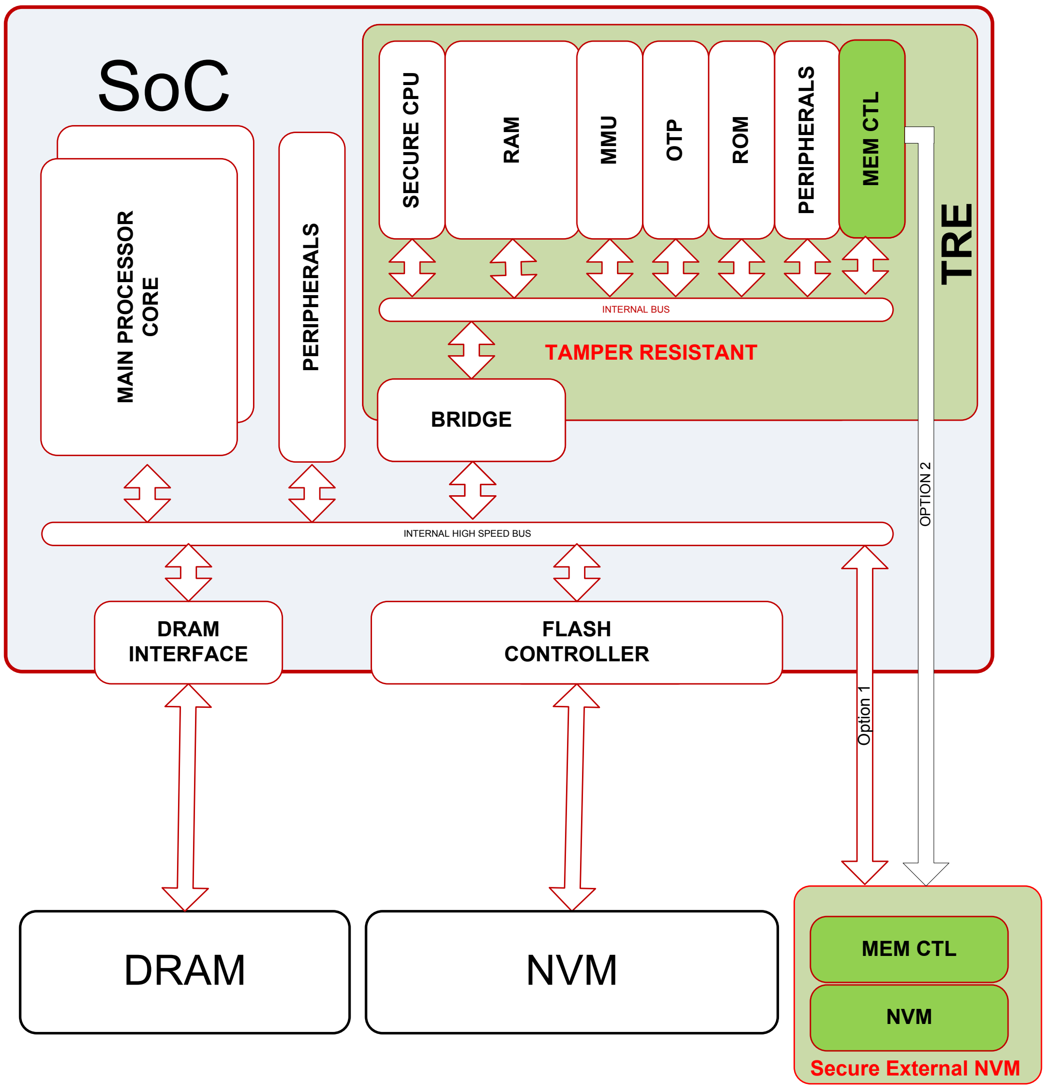


Figure 18: Secure NVM proposal

5.8 Primary Platform Technical Solutions Summary

Table 1 summarizes the technical solutions for the data during storage or transfer into NVM and how they are met or not (-) by each architecture.

	Architecture			
Properties	**Internal NVM**	**External NVM**	**In-package NVM**	**Secure NVM**
Confidentiality during storage	Crypto	Crypto	Crypto	Crypto
Confidentiality during transfer	Crypto	Crypto	Crypto	Crypto
Authenticity during storage	Crypto	Crypto	Crypto	Crypto
Authenticity during transfer	Crypto	Crypto	Crypto	Crypto
Anti-rollback protection/Anti replay	In-die	-In-die + crypto	In-package	In-die
Perfect Forward Secrecy	In-die	-In-die**	In-package	Crypto
Denial of Service attack	Best efforts*	Best efforts*	Best efforts*	Best efforts*

Table 1: NVM technical solutions per architecture

Notes to annotations in Table 1:

- **In-die**: security relies on the fact that the memory is internal to the TRE, on the same die.
- **In-package**: security relies on the fact that the memory is in the same package as the SoC.
- **Crypto**: security relies on digital protections based on strong cryptography
- * Depending on implementation (e.g. if direct access to the flash controller is supported).
- ** Implementation specific solutions can enable support for the Perfect Forward Secrecy in-die with a good level of Tamper Resistance.
- **Perfect Forward Secrecy: "**In cryptography, forward secrecy (FS; also known as perfect forward secrecy) is a property of secure communication protocols in which compromise of long-term keys does not compromise past session keys. Perfect Forward Secrecy support protects past sessions against future compromises of secret keys or passwords. If Perfect Forward Secrecy support is utilized, encrypted communications and sessions recorded in the past cannot be retrieved and decrypted should long-term secret keys or passwords be compromised in the future, even if the adversary actively interfered." (Wikipedia)

5.8.1 Perfect Forward Secrecy

It is therefore recommended that the interface between the TRE and the Flash must be encrypted with a strong algorithm, and that the keys for this encryption must be protected.

If a long term secret (static key) used for this encryption is stored within the TRE, an attacker might get access to it and decrypt data within the flash, or previous exchanges on the interface. One solution would be to use ephemeral keys, however, for reasons of cost effectiveness, we assume that the SoC is not able to store persistent variables within the TRE as there’s no eNVM/eFLASH.

To mitigate the risk, we may consider a System in Package (SiP) or embedded Package on Package (ePOP) solution by which the flash would be embedded in the same package as the TRE. This would force the attacker to perform a physical attack at the package level involving costly equipment and then prevent easily replicable frauds

5.8.2 Replay attack & Denial of Service:

For security reasons we recommend the use of a carefully designed shared FLASH controller (and not a generic one) for the following reasons:

- It can be difficult to prevent access from a main core processor to the control of the private NVM partition dedicated to the TRE (risk of replay attack and DoS).
- The priority to manage the memory transaction queuing from the main core processors and the TRE can be compromised (no guarantee of the memory transaction).
- Flash memory encryption/decryption

The size of the RAM memory in the TRE is a tradeoff between the performance expectation and the latency to access and perform a memory transaction within the FLASH memory. The lower is the latency to access the memory; the smaller is the RAM memory in the TRE.

We may consider the use of a hardware stream cipher for addressing the following objectives:

- Get the best performance in a cost effective manner.
- Not overload the secure CPU with intensive encryption/decryption processing.
- Not share the crypto-processor of the TRE for performing concurrent tasks at different level of security and liability.
- The stream cipher key will use a private seed non accessible from any hardware or software interface.

Optimally the aforesaid stream cipher has to compute/check the integrity of the exchanged data.

The selection of the stream cipher principle shall be validated by security experts.

5.9 Architecture proposal supporting the Primary Platform

Modern microprocessors (i.e. CPU) have a least 2 protection modes

- Privileged: a microkernel runs exclusively in this mode.
- Unprivileged: after starting of a microkernel, threads run exclusively in this mode

And 2 operation modes

- Thread or user: run in privileged or unprivileged modes
- Kernel or supervisor: run only in privileged mode. The CPU runs automatically in this mode when an exception occurs (hardware or software (SVC: SuperVisor call)). This mode allows accessing critical CPU instructions and hardware functions.

Table 2 illustrates the combination between the operation and the protection modes applied to VPP/VPP Application model.

	Unprivileged	Privileged
Thread	VPP Application and VPP	Starting of the Primary Platform only
Kernel	Impossible by design	VPP only

Table 2: Operation and protection modes

Figure 19 illustrates the concepts in Table 2 above.

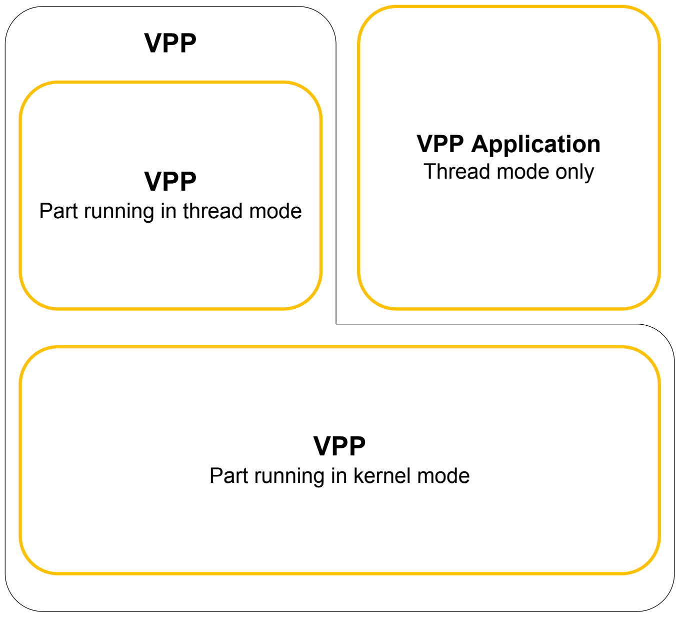
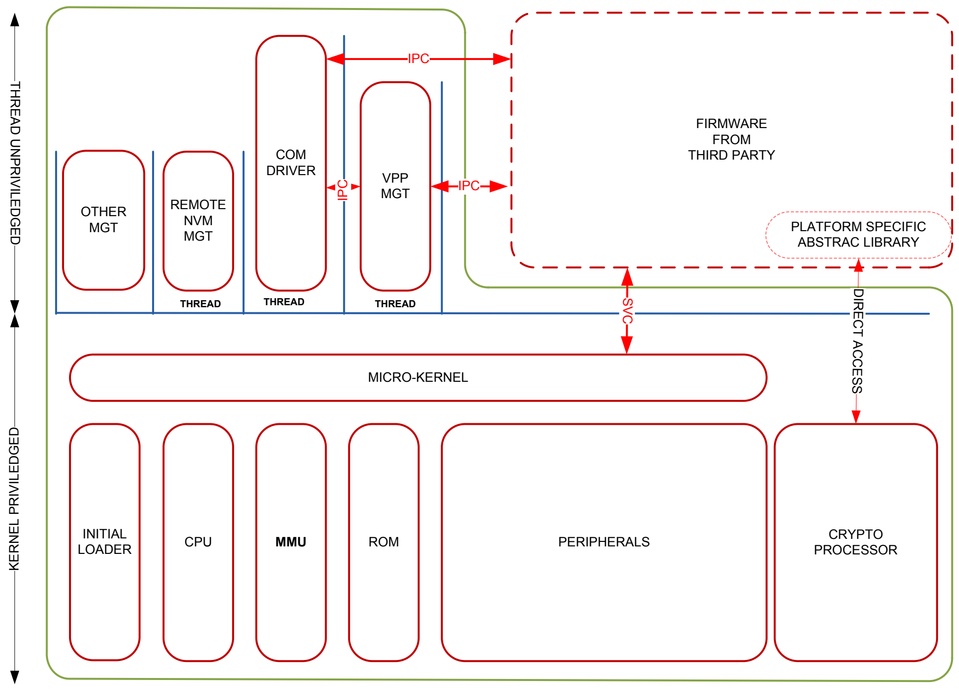


Figure 19: VPP modes split

: illustrates the details of the Primary Platform.



PRIMARY PLATFORM

Figure 20: Primary Platform Architecture Proposal

The Primary Platform is a combination of hardware and software components which are designed in such a way to support the overall Primary Platform requirements.

Secured CPU offers usually at least two modes:

- The kernel mode with privileges. This mode grants access to critical peripherals ensuring the overall security of the TRE.
- The thread mode without privileges. This mode confines the software running in this mode to limited access to defined resources. Any infringements of the limit/rules generates an exception (fault)

The TRE within a SoC enforces by design the use of a microkernel. A microkernel respects a principle of minimality as it is defined as:

“A concept is tolerated inside the microkernel only if moving it outside the kernel, i.e., permitting competing implementations, would prevent the implementation of the system's required functionality.”

(source : Wikipedia and SEL4 website).

5.9.1 Inter-Process Communications (IPC)

Two approaches are possible to perform the Inter Process Communication (IPC) allowing the transfer of data between two threads which are isolated:

- **Indirect**: The µkernel gets data from the transmitting thread and copy the data to the receiving thread. Then the µkernel schedules the receiving thread for processing the said data. The µkernel shall access the content of the data to transfer.
- **Direct**: The transmitting thread shares a memory in which the data are available, with the receiving thread. Then the µkernel schedules the receiving thread for processing the said data. The µkernel has no need for accessing the data to transfer.

A requirements-based Direct IPC enforces the firewalling in Figure 22.

Figure 21

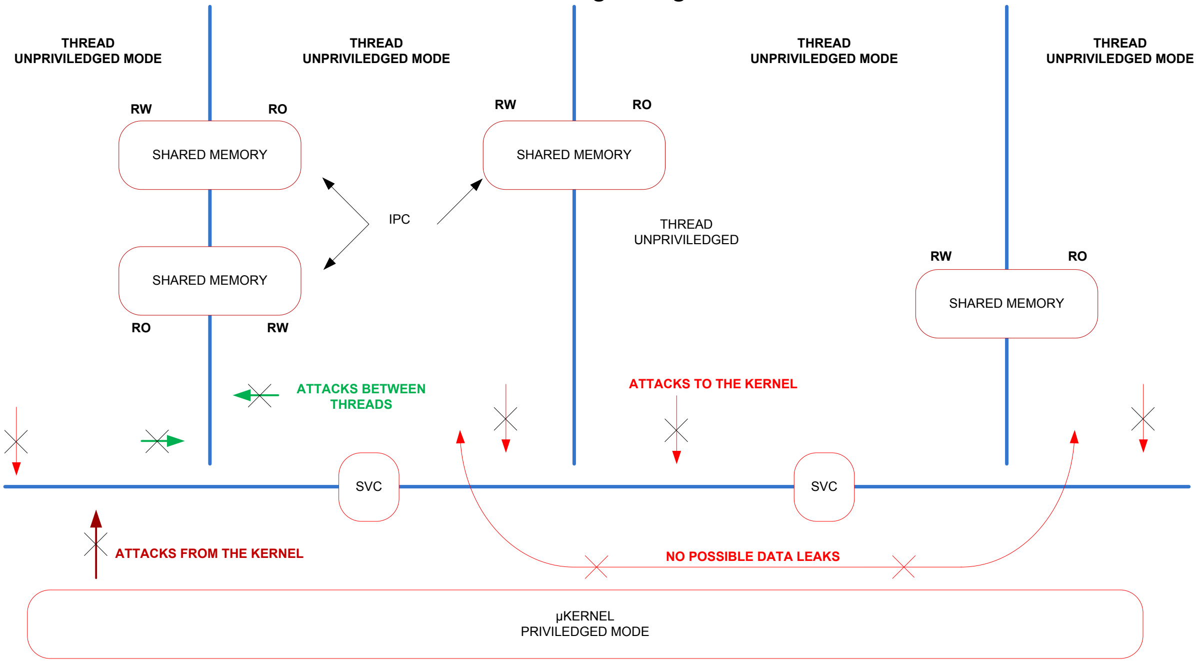


Figure 22: Firewalling

The above model enforces a µkernel using an IPC excluding the copy of data by the µkernel to exchange data between threads. This model may enforce a double protection (hardware and software) for preventing attacks in kernel mode.

We illustrate the indirect IPC by using an example:

Alan (thread A) wants to transfer $10 to Bob (thread B).

1. Indirect IPC: Alan opens his wallet; Kevin (the kernel) takes $10 and gives it to Bob.
2. Direct IPC: Alan gives $10 to Bob

In the first case, you need to trust Kevin twice: Kevin shall only take $10 from Alan's wallet, and Kevin shall give it to Bob and not to somebody else (e.g. data leaks to communication driver).

In terms of certification, the case 1 is obviously more difficult and impacts strongly the certification of Alan.

5.9.2 Fault Attacks

Fault attacks may strongly disturb a CPU which will be no longer valid to ensure the credentials protections based on software countermeasures. Moreover, software countermeasures in kernel mode may strongly impact the overall performance of the Primary Platform and should lead the Primary Platform designer to implement hardware assistance.

5.9.3 Memory Management

In addition to the microkernel, the use of virtual memory management provided by a MMU is expected to support the functional requirements of the Primary Platform.

For information, the die surface of an MMU tailored for secure controller as the one used within legacy smart card controllers is equivalent to about 16 KB of SRAM. An MMU contains the functionality of the MPU (equivalent to the die surface of 4KB of SRAM), and the MPU functionality may be removed when an MMU is implemented.

In consequence, it is more cost effective (die surface) to favor an MMU as soon the RAM size of the TRE is greater than 32 Kbyte, rather than promoting a larger memory to support the use cases as defined in the section 3 of this document.

The secure foundation of the Primary Platform should be based on a minimal microkernel controlling an MMU granting the access to resources. In consequence, both components (the microkernel and the MMU) are critical elements and may need to support a high certification level.

In consequence of the above, all threads (VPP or Application) access only the virtual memory (VME) and the virtual registry of peripherals (VRE). The VME and VRE are duly managed by an MMU (size, usage and rights) and map respectively physical memory (PME) and a physical registry of peripherals (PRE).

5.9.4 The VPP Applications

The counterpart of the Virtual Primary Platform is the VPP Application.

Figure 23 illustrates the VPP Application ecosystem.

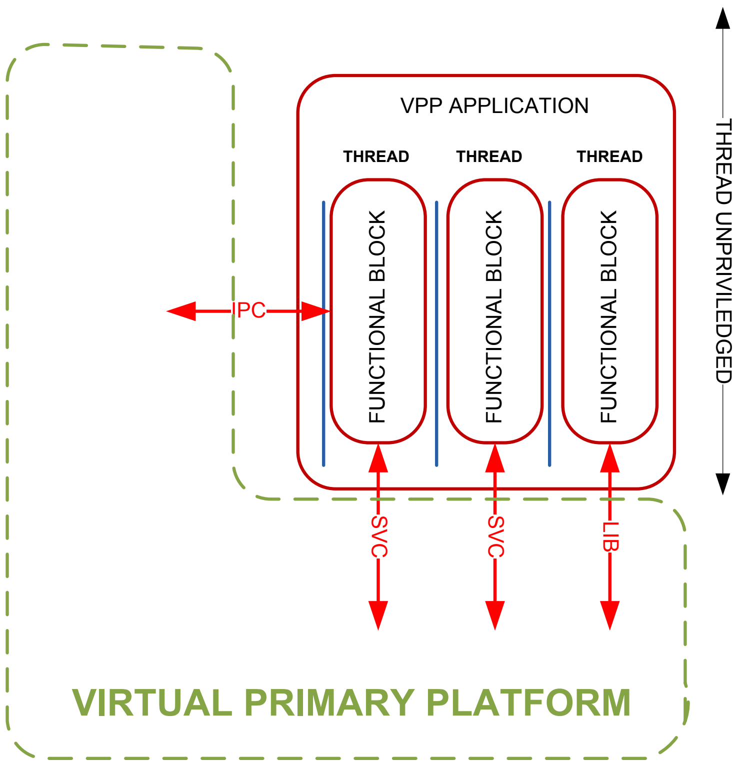


Figure 23: VPP Application

The access to the facilities of the VPP or the microkernel is commonly based on IPC (µkernel Inter Process Communication) and SVC (Super-Visor Call). The access for threads granting the rights to access peripherals is performed via thin and direct abstract interfaces.

The VPP Application can access to virtual hardware as expressed in Figure 24.

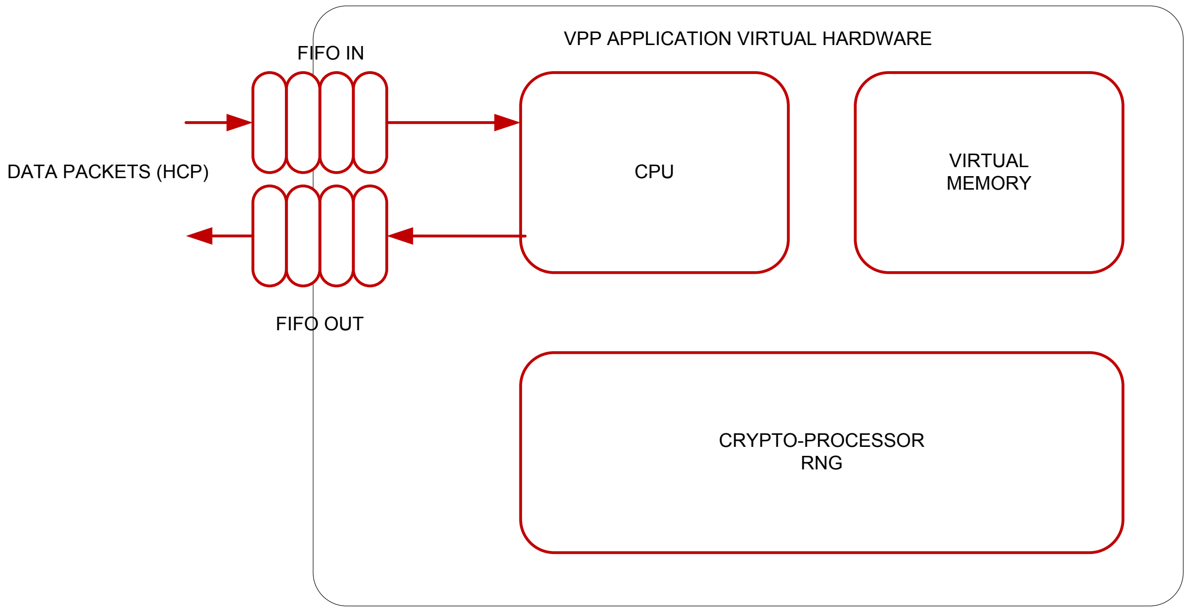


Figure 24: VPP Application virtual hardware

Annex A Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
1.0	10 May 2017	Transfer to GSMA Template for Approval	PSMC/TG	Ian Pannell

Other Information

Type	Description
Document Owner	Sergio Cozzolino
Editor / Company	

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

Your comments or suggestions & questions are always welcome.