Hardware/Software Support for Securing Virtualization in Embedded Systems

Franck Bucheron

IRISA-CAIRN
DGA

December 4, 2014

1. Improve the trust in virtualization for ARM embedded platforms.
2. Modify studied propositions, add new ideas to build a functionnal virtual embedded system as secure as possible.

1. Improve the trust in virtualization for ARM embedded platforms.
2. Modify studied propositions, add new ideas to build a functionnal virtual embedded system as secure as possible.

Starting point: consider that embedded systems efficiently support virtualization

Use a combined *hardware/software (HW/SW) approach*, with:

1. Implementation on a Zynq-7000 HW/SW platform (ARM Cortex-A9 dual-core) and a FPGA (Artix-7),
2. Re-use of existing HW cryptographic IPs,
3. Addition a few other HW blocks: TPM and vTPM,
4. At the SW level, adaptation of a Xen-like hypervisor to our HW architecture.

Introduction

Four different domains:

Introduction

Four different domains:

1. Hardware,
2. Existing software virtualization solutions in the open-source world,
3. Concepts of trusted computing,
4. Threats.

FPGA *and* ASIC: No more opposition between with these 2 technologies.

To prototype hardware, only FPGA is relevant:

1. Dynamicaly reconfigurable,
2. Can keep the configuration if in flash mode,

FPGA *and* ASIC: No more opposition between with these 2 technologies.

To prototype hardware, only FPGA is relevant:

1. Dynamicaly reconfigurable,
2. Can keep the configuration if in flash mode,

To prototype software, only ASIC is relevant:

1. Standard technology optimized,
2. Performances.

FPGA *and* ASIC: No more opposition between with these 2 technologies.

To prototype hardware, only FPGA is relevant:

1. Dynamicaly reconfigurable,
2. Can keep the configuration if in flash mode,

To prototype software, only ASIC is relevant:

1. Standard technology optimized,
2. Performances.

Hybrid platforms

Get avantages of the two types of circuits, plus:

1. No extra daughter card needed,
2. Shared peripherals if needed,
3. Imagination required.

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons
SierraVisor	-paravirtualization -full hardware virtualization	-Zynq-7000 version is closed source

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons
SierraVisor	-paravirtualization -full hardware virtualization	-Zynq-7000 version is closed source
Xtratum	-allowing to run RTOS or real-time executives	-difficult to get the source code -don't run on Zynq-7000

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons
SierraVisor	-paravirtualization -full hardware virtualization	-Zynq-7000 version is closed source
Xtratum	-allowing to run RTOS or real-time executives	-difficult to get the source code -don't run on Zynq-7000
Xen ARM	-Xen ARM PV project -Xen ARMv7+ project	-no port on Zynq-7000

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons
SierraVisor	-paravirtualization -full hardware virtualization	-Zynq-7000 version is closed source
Xtratum	-allowing to run RTOS or real-time executives	-difficult to get the source code -don't run on Zynq-7000
Xen ARM	-Xen ARM PV project -Xen ARMv7+ project	-no port on Zynq-7000
x-hyp	-ARINC like scheduling -reduced trusted base	-runs only on Qemu

Few open-source hypervisors for ARM cores

Review:

Nom	Pros	Cons
SierraVisor	-paravirtualization -full hardware virtualization	-Zynq-7000 version is closed source
Xtratum	-allowing to run RTOS or real-time executives	-difficult to get the source code -don't run on Zynq-7000
Xen ARM	-Xen ARM PV project -Xen ARMv7+ project	-no port on Zynq-7000
x-hyp	-ARINC like scheduling -reduced trusted base	-runs only on Qemu
Embedded Xen	-source code available -runs on a Zynq-7000 platform	-Current limitation to 1 domU

Definition:

TC refers to technologies and proposals for resolving computer security problems through hardware enhancements and associated software modifications.

Definition:

TC refers to technologies and proposals for resolving computer security problems through hardware enhancements and associated software modifications.

1. Trusted Computing Base
   1. Set of all hardware, software and procedural components that enforce the security policy,
   2. Must be as small as possible.

Definition:

TC refers to technologies and proposals for resolving computer security problems through hardware enhancements and associated software modifications.

1. Trusted Computing Base
   1. Set of all hardware, software and procedural components that enforce the security policy,
   2. Must be as small as possible.
2. Trusted Platform Module (TPM) - current version 2.0
   1. Hardware component that provides a set of fixed cryptographic and security functions,
   2. (Originally) intended to be platform independant.

Definition:

TC refers to technologies and proposals for resolving computer security problems through hardware enhancements and associated software modifications.

1. Trusted Computing Base
   1. Set of all hardware, software and procedural components that enforce the security policy,
   2. Must be as small as possible.
2. Trusted Platform Module (TPM) - current version 2.0
   1. Hardware component that provides a set of fixed cryptographic and security functions,
   2. (Originally) intended to be platform independant.
3. Trusted Software Stack (TSS)
   1. Issues low-level TPM commands and receives low-level TPM responses on behalf of high-level applications,
   2. (Originally) intended to be platform independant.

TPM and virtualization

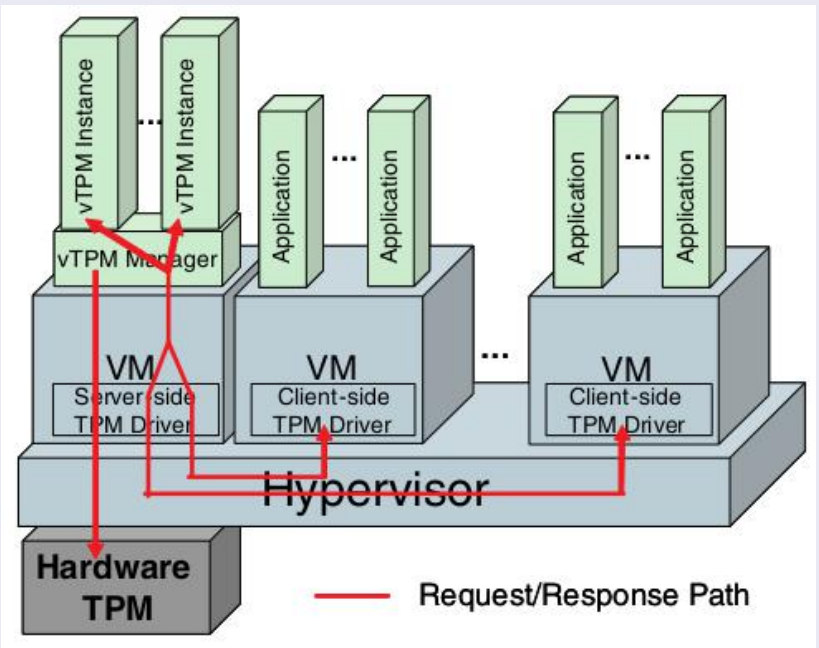
Some studies (mostly dedicated to x86) tried to solve the problem.

TPM and virtualization

Some studies (mostly dedicated to x86) tried to solve the problem.

vTPM: Virtualizing the Trusted Platform Module by Stefan Berger, 2006

Advantages : High fidelity to TCG specifications, no hardware constraints.
Disadvantges : Weak strengh face to hardware or software attack, increase of TCB, hard to implement.

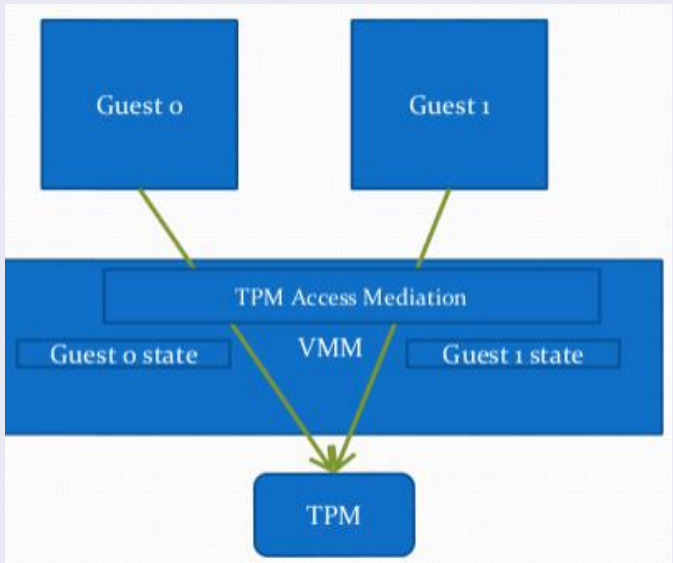


TPM and virtualization

Some studies (most dedicated to x86) tried to solve the problem.

Enhancing Trusted Platform Modules with Hardware-Based Virtualization Techniques by Frederic Stumpf, 2008

Advantages : Stronger.
Disadvantges : Many swap of context in the CPU, exclusively on x86 architecture.

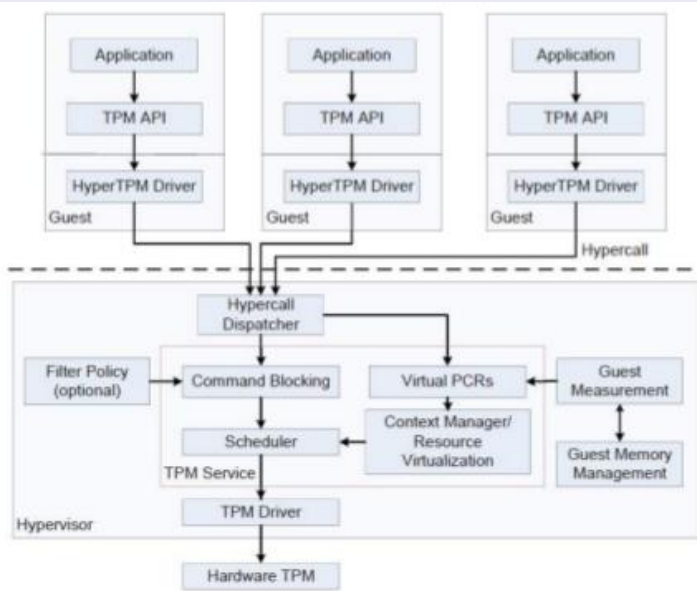


TPM and virtualization

Some studies (most dedicated to x86) tried to solve the problem.

Para-Virtualized TPM Sharing by Paul England, 2008

Advantages : All is done in the VMM, scheduled access to TPM.
Disadvantges : Weak strengh face to HW/SW attack, increase of TCB, hard to implement.



boot: confusion about definitions

Need a boot that proves to deliver more than a *secure initial state*.

boot: confusion about definitions

Need a boot that proves to deliver more than a *secure initial state*.

authenticated Boot	**secure Boot**
Passive method	Active method
Integrity measures are stored securely	Proof to the system is existential
Provides proof to a third party via attestation	Unable to prove configuration to a third party
Require a TPM	Do not require a TPM

boot: confusion about definitions

Need a boot that proves to deliver more than a *secure initial state*.

authenticated Boot	**secure Boot**
Passive method	Active method
Integrity measures are stored securely	Proof to the system is existential
Provides proof to a third party via attestation	Unable to prove configuration to a third party
Require a TPM	Do not require a TPM

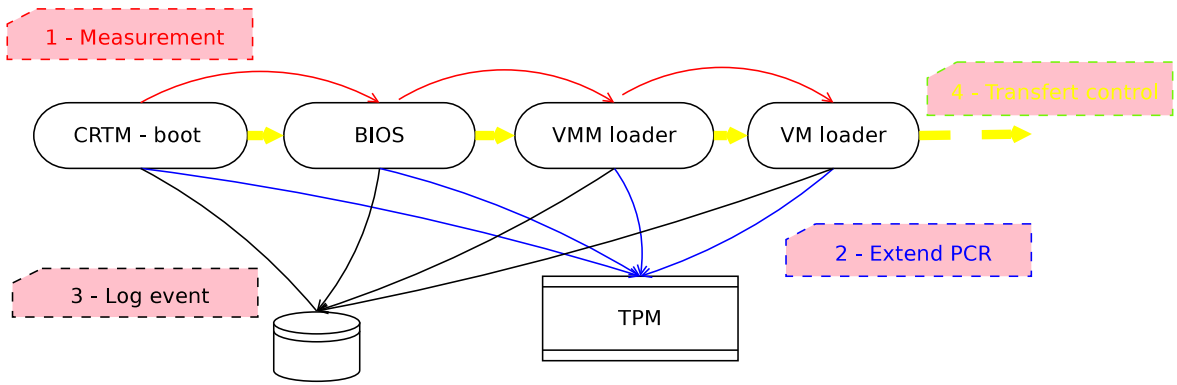


Figure: trusted boot = secure boot + authenticated boot

Hardware threats

At lower level, main goal is to counterfeit the IP or IC, steal or copy it.

1. Backdoor in hardware (solution: have its own foundry),
2. Hardware vulnerability (solution: update hardware),
3. Hardware assisted malware (solution: develop antivirus software).

Hardware threats

At lower level, main goal is to counterfeit the IP or IC, steal or copy it.

1. Backdoor in hardware (solution: have its own foundry),
2. Hardware vulnerability (solution: update hardware),
3. Hardware assisted malware (solution: develop antivirus software).

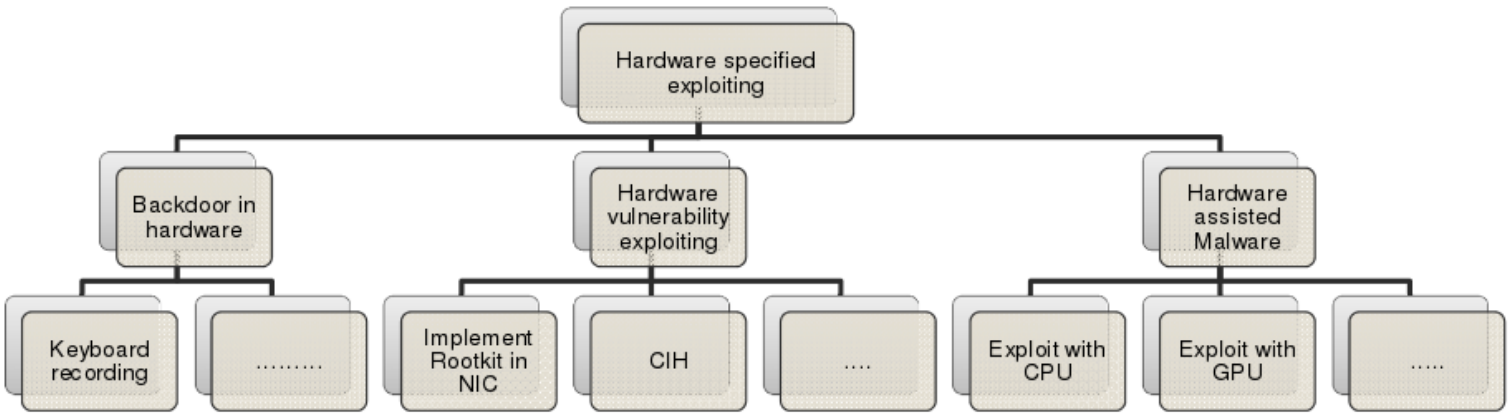


Figure: How to exploit

Software threats

On virtualization (DOS, escalation of privilege or acquire information)

Software threats

On virtualization (DOS, escalation of privilege or acquire information)

1. Compromise the guest (guest to guest, internet to guest, VM migration to guest, Managment VM to guest),
2.
3.

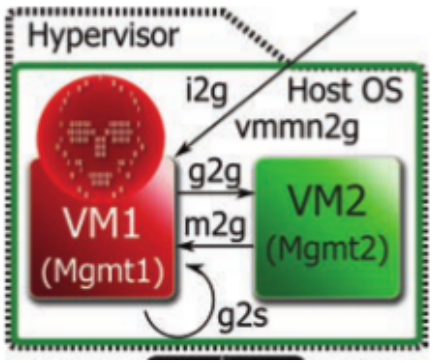


Figure: from A Survey of Security Issues in Hardware Virtualization - 2013

Software threats

Compromise virtualization making DOS, try escalation of privilege or acquire information...

1.
2. Compromise the host OS (guest to host, internet to host, host to the self),
3.

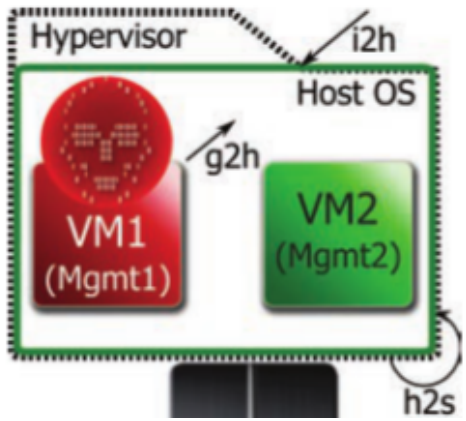


Figure: from A Survey of Security Issues in Hardware Virtualization - 2013

Software threats

Compromise virtualization making DOS, try escalation of privilege or acquire information...

1.
2.
3. Compromise the hypervisor (guest to hypervisor, host OS to hypervisor, Physical/Physical Management Interface to Hypervisor),

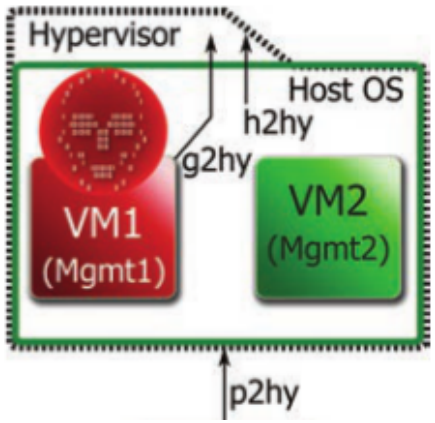


Figure: from A Survey of Security Issues in Hardware Virtualization - 2013

1. Run on ARM cores with no virtualization extensions,
2. Adapt the solution to a dedicated platform (Zynq-7000),
3. Respect the basis of TCG compability for TPM and vTPM.

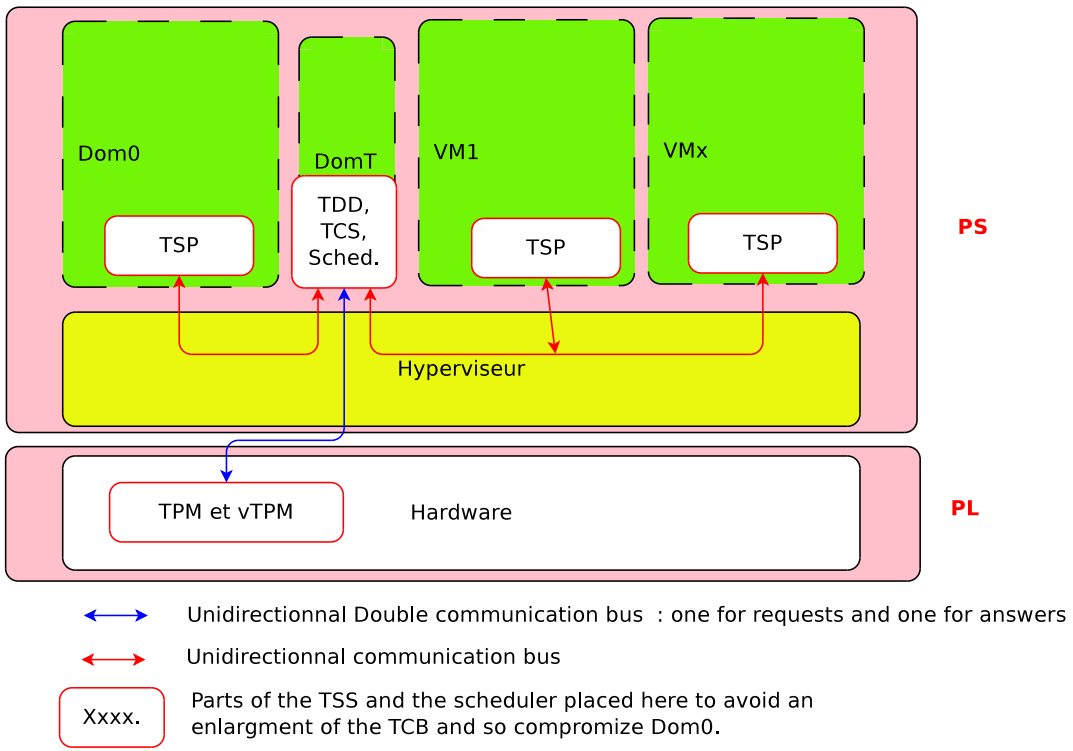


Figure: A leigthweight hypervisor of Xen type with VM looking for trust in an FPGA implemented TPM/vTPM

Problem of the chain of trust

1. The Zynq-7000 offered us only a *secure boot*
2. The AES-256 engine and the hash of the RSA public key are hard coded in Zynq
3. Possibility to change keys and use efuse or BBRAM.

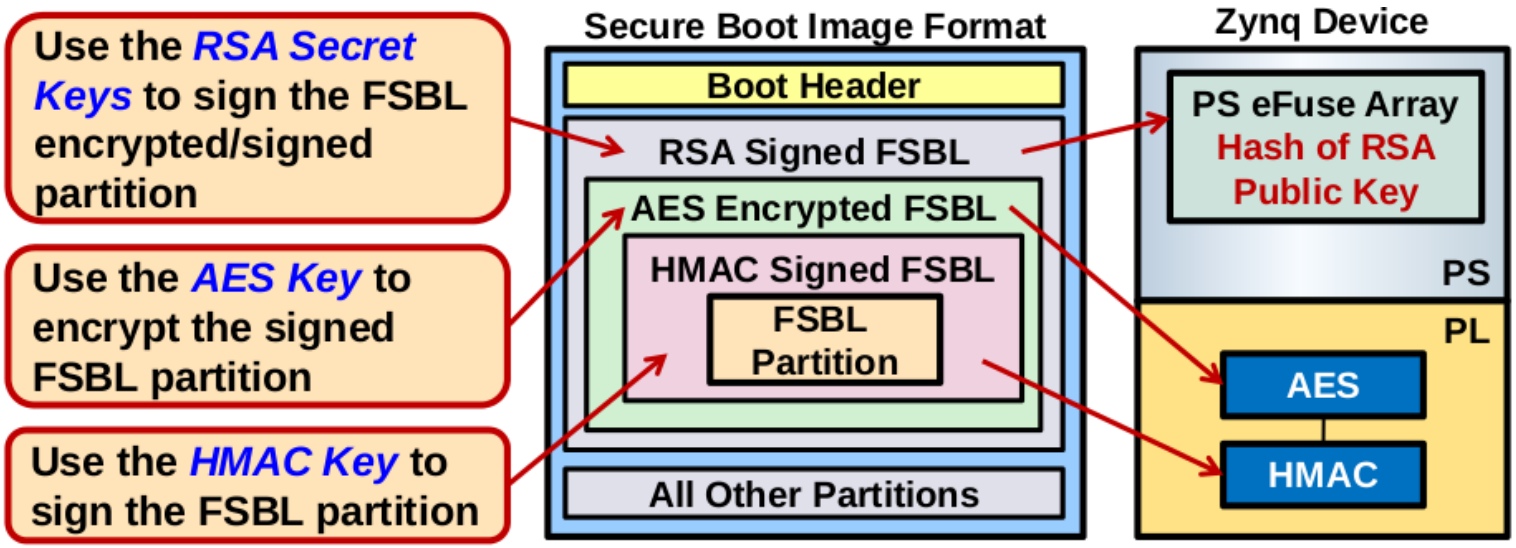
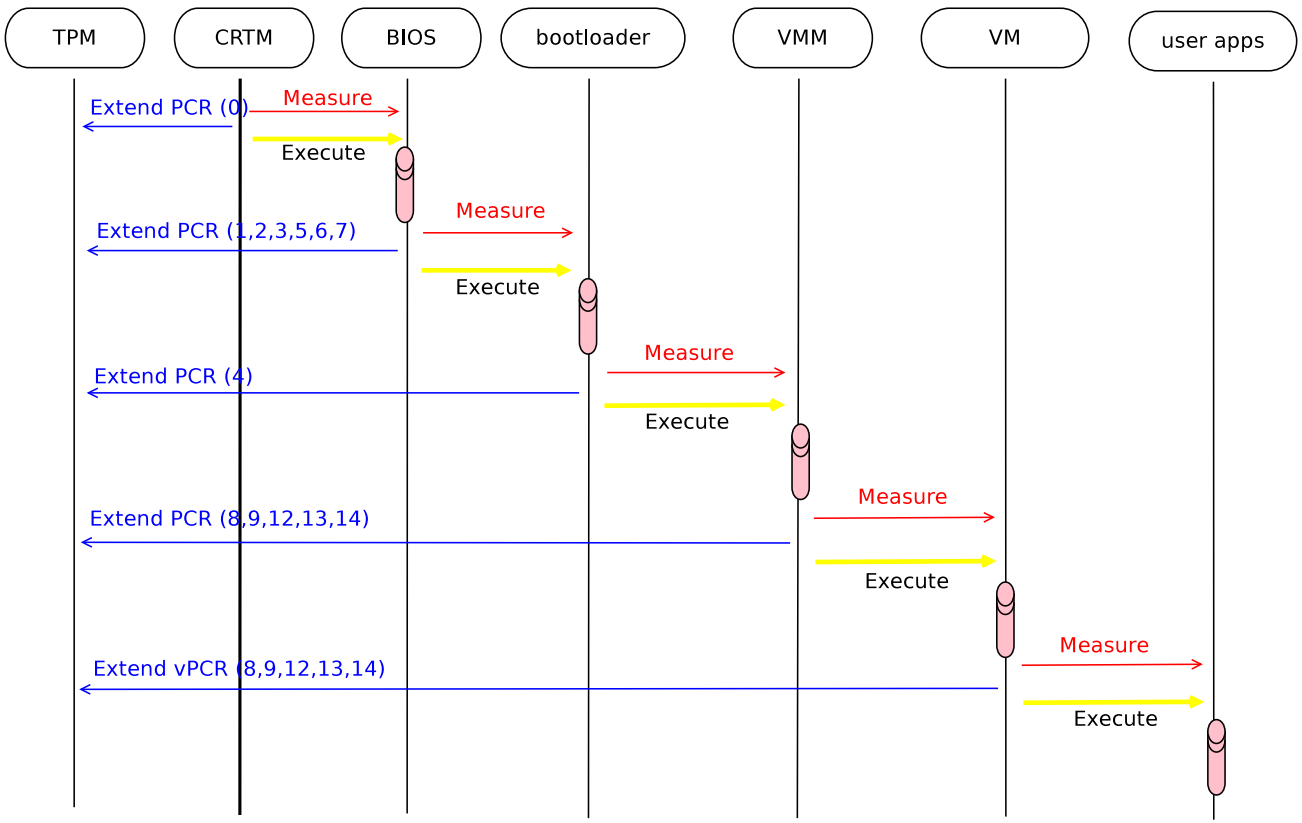


Figure: from AVNET - x-fest2014

Implementation of a measured sequence

Fill the gap between secure boot and launch of VM.

1. Nothing can be done with stage-0 boot,
2. With FSBL : TPM/vTPM bitstream,
3. With SSBL : U-BOOT, VMM and VM.



TPM and vTPM provide the same services

1. The I/O interface will only manage the requests/answers from the DomT and the ID of the requester,
2. The execution engine of the IP will be the central point of execution,
3. Partial reconfiguration will be implemented for large body of logic (typical crypto engines).

TPM and vTPM provide the same services

1. The I/O interface will only manage the requests/answers from the DomT and the ID of the requester,
2. The execution engine of the IP will be the central point of execution,
3. Partial reconfiguration will be implemented for large body of logic (typical crypto engines).

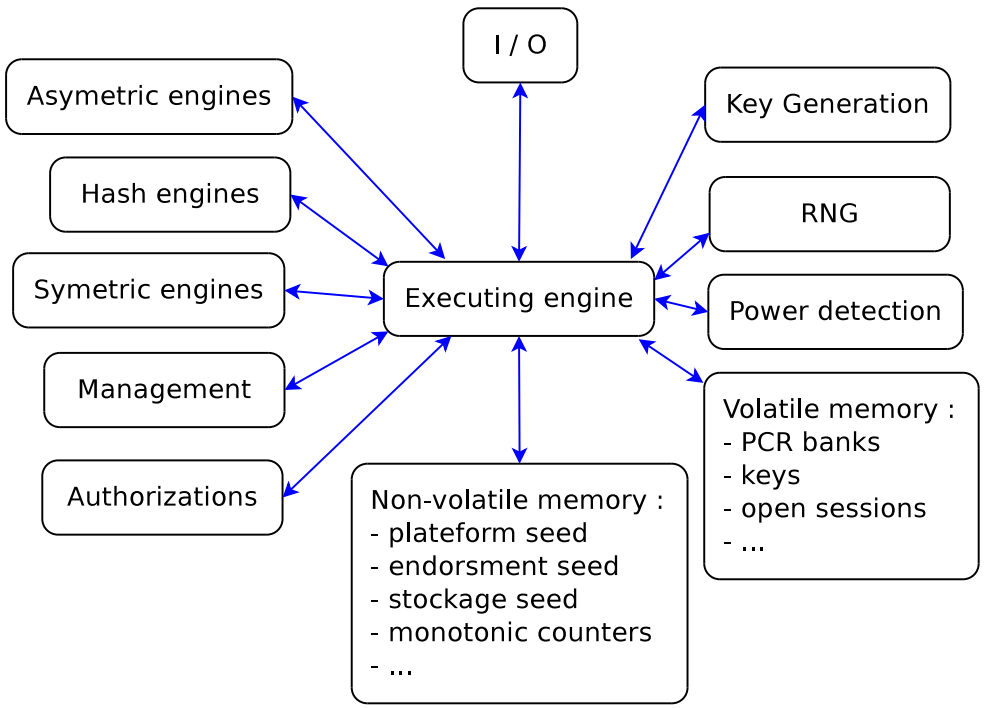


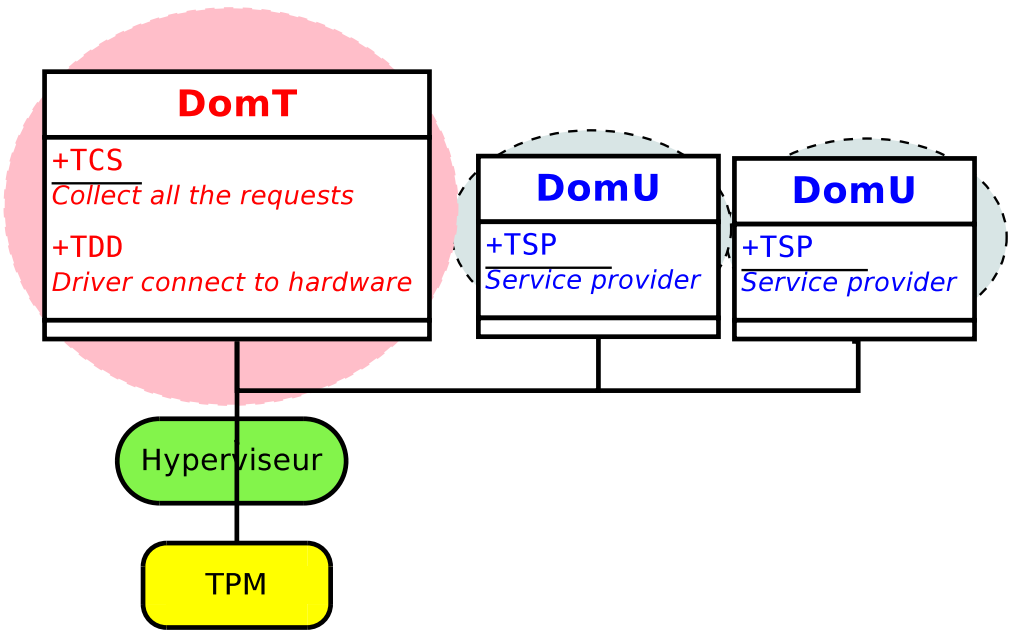
Figure: heart of the TPM/vTPM

DomT

1. Will have a part of the TSS (TDD and TCS commands),
2. Will schedule and verify the requests (from the VMs TSP) and route the answers,
3. Later : will sign requests to the TPM.

DomT

1. Will have a part of the TSS (TDD and TCS commands),
2. Will schedule and verify the requests (from the VMs TSP) and route the answers,
3. Later : will sign requests to the TPM.

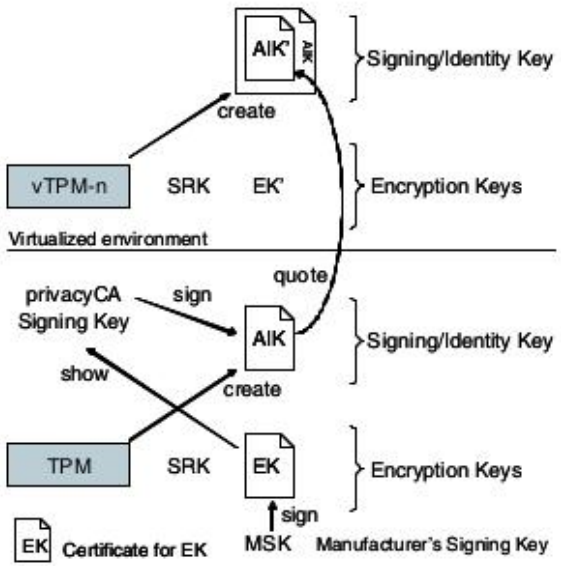


Managment of keys from vTPM: Virtualizing the Trusted Platform Module - 2006

1. Will be inspired from S. BERGER,
2. Independent key hierarchy per vTPM instance,
3. AIK of vTPM signed by AIK of the hardware TPM for commodity.

Managment of keys from vTPM: Virtualizing the Trusted Platform Module - 2006

1. Will be inspired from S. BERGER,
2. Independent key hierarchy per vTPM instance,
3. AIK of vTPM signed by AIK of the hardware TPM for commodity.



Hypervisor

Need for new hypervisor functionnalities

1. Add DomU (for our DomT at least) and ID for VM
2. Add new syscalls
3. Modify source files

Hypervisor

Need for new hypervisor functionnalities

1. Add DomU (for our DomT at least) and ID for VM
2. Add new syscalls
3. Modify source files

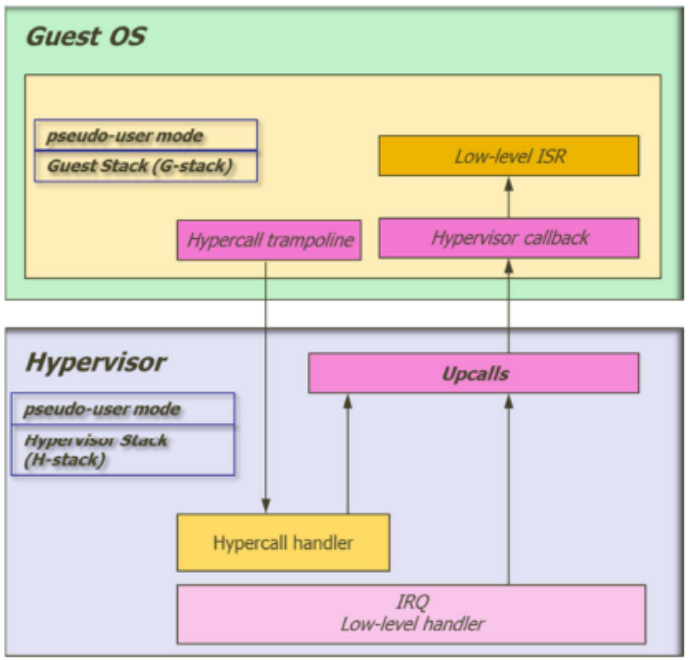


Figure: from EmbeddedXEN: A Revisited Architecture of the XEN hypervisor to support ARM-based embedded virtualization - 2012

A lot of work has to be done

1. Produce the IP (TPM first, vTPM second),
2. Plug the re-used IPs, link them,
3. Modify the hypervisor,
4. Improve the security of implementation,
5. Test and debug.

A lot of work has to be done

1. Produce the IP (TPM first, vTPM second),
2. Plug the re-used IPs, link them,
3. Modify the hypervisor,
4. Improve the security of implementation,
5. Test and debug.

Some other future options

1. IP updates,
2. Migration of VMs,
3. parallelism,
4. Use of TZ (for the DomT ?)

Thanks for your attention

All the reference to the figures can be found on the joined article.