Technologies TPM et Intel TXT

OSSIRB, Rennes – 29/03/2011

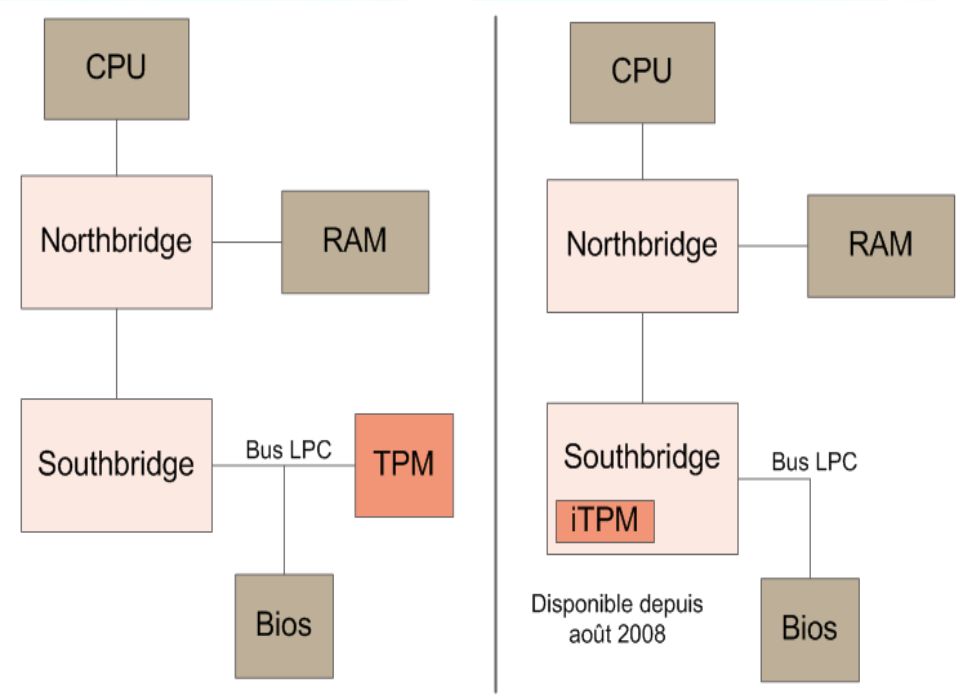
Goulven Guiheux
Frédéric Guihery

Plan

- Technologie TPM
  - Présentation
  - Produits implémentant le TPM
- Technologie Intel TXT
  - Présentation
  - Trusted Boot
  - Attaques et contremesures

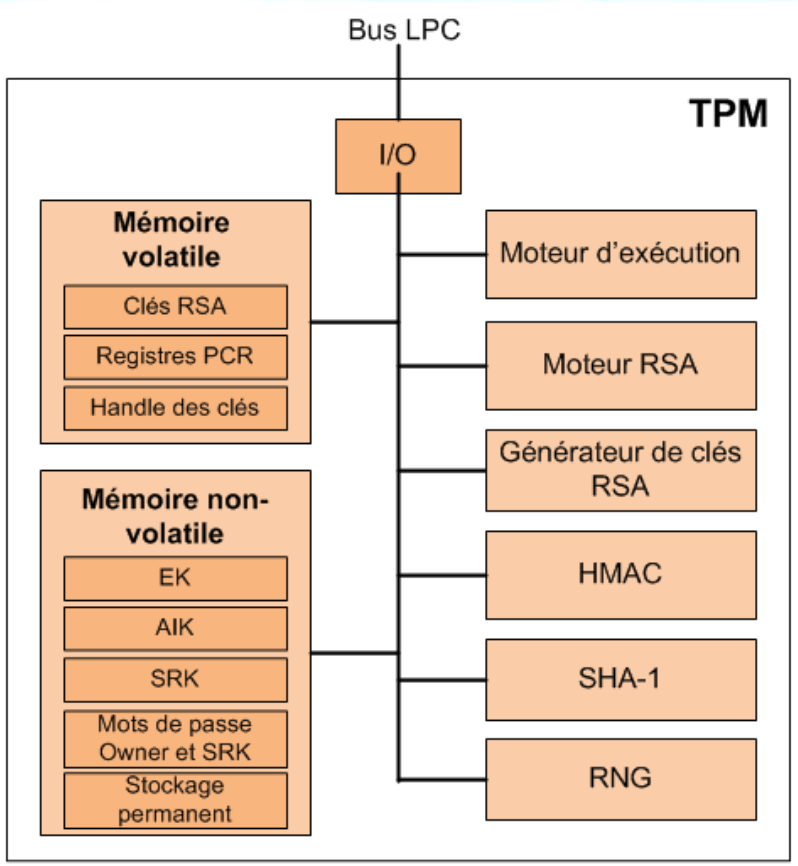
Technologie TPM
Positionnement

- Composant cryptographique
- Passif
- Bus LPC
  - Faible débit
- Constructeurs
  - Atmel
  - Broadcom
  - Infineon
  - Intel
  - ST Microelectronics



Technologie TPM Architecture

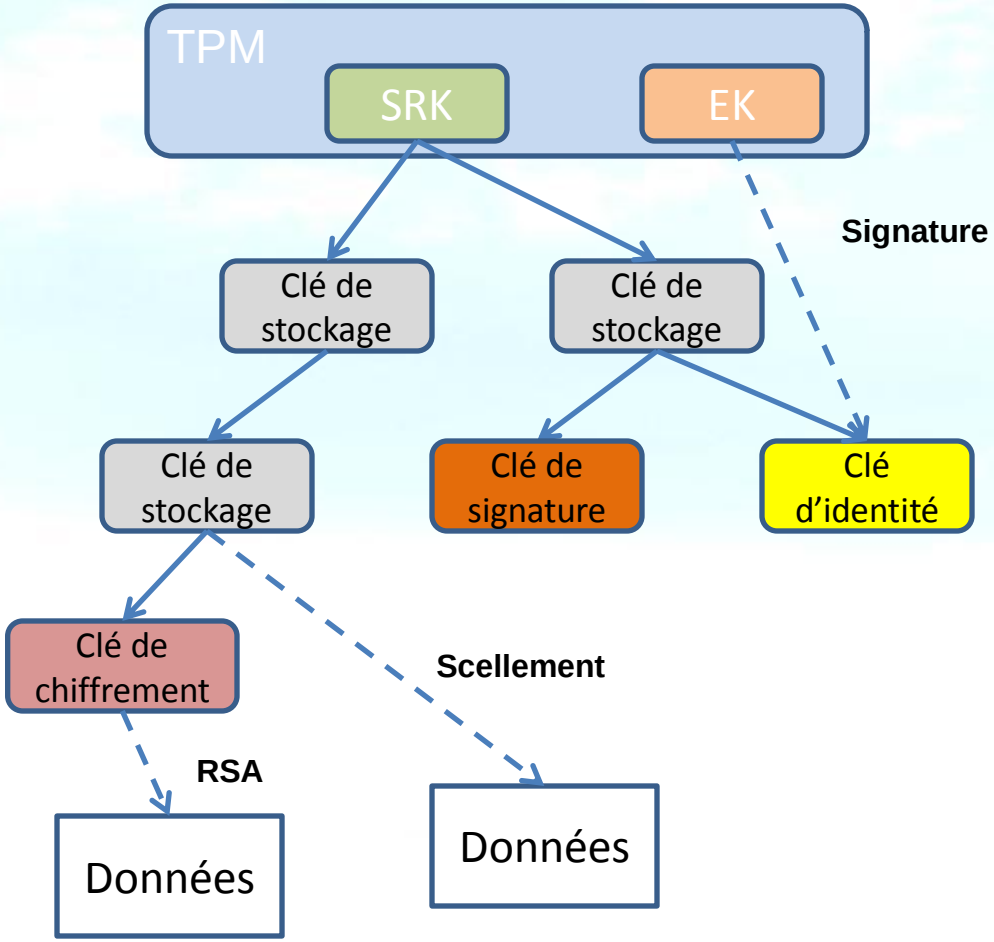
- Cryptographie RSA
- Générateur de clés RSA
- SHA1
- HMAC
- 24 PCR de 160 bits
- Générateur d'aléa
- RAM
- ROM



Technologie TPM
Clés

- Clés RSA
- EK : Endorsement Key (2048 bits min.)
  - Identité du TPM
  - Stockée dans la NV du TPM
- SRK : Storage Root Key (2048 bits min.)
  - Clé racine de stockage sécurisé du TPM
  - Stockée dans la NV du TPM
- Autres clés
  - Partie privée protégée par la clé parente
  - Stockées à l'extérieur du TPM
  - Doit être chargée avant d'être utilisée
  - Chargement nécessite d'utiliser la clé parente (→authentification pour utiliser la clé parente)

Technologie TPM
Architecture des clés



Technologie TPM
Mesure d'intégrité

- Extension des PCR

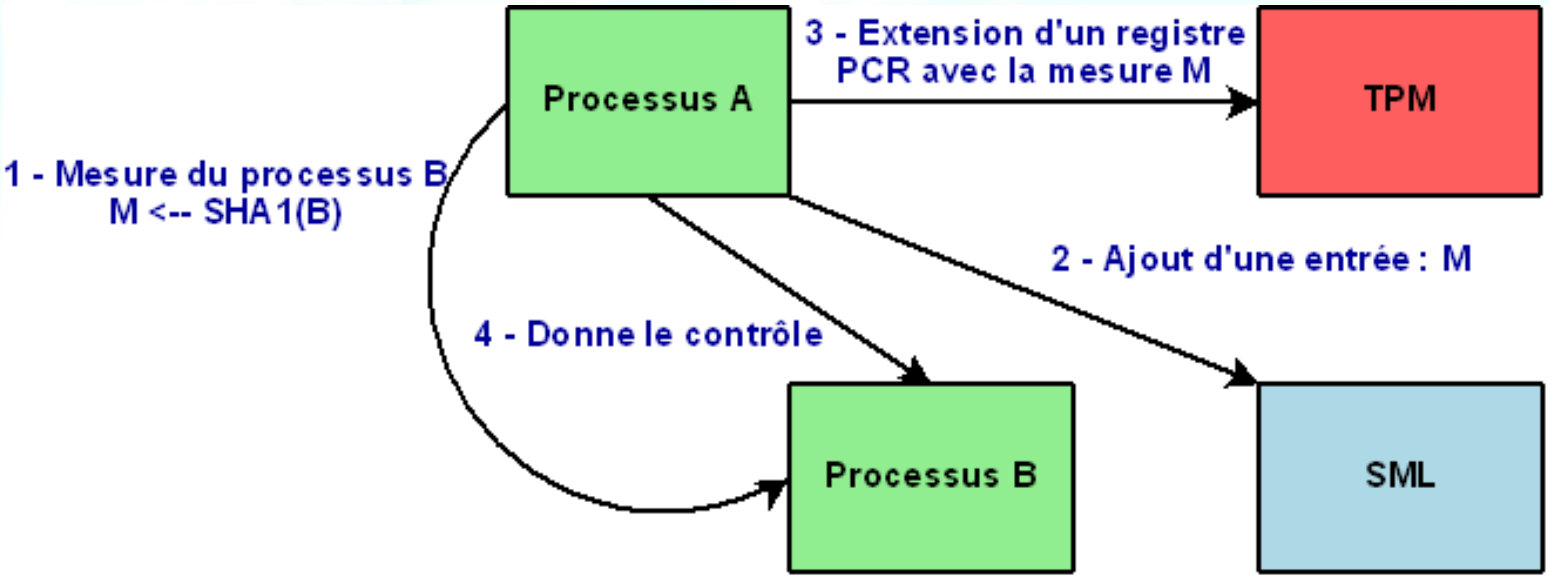
$$PCR[t+1] \leftarrow \text{SHA-1}\big(PCR[t] \| M\big)$$

$$M = \text{SHA-1}(\text{DATA})$$

- SCRTM (Static Core Root Trust of Measurement)
  - Début de la chaîne de mesure
  - Premiers octets du BIOS

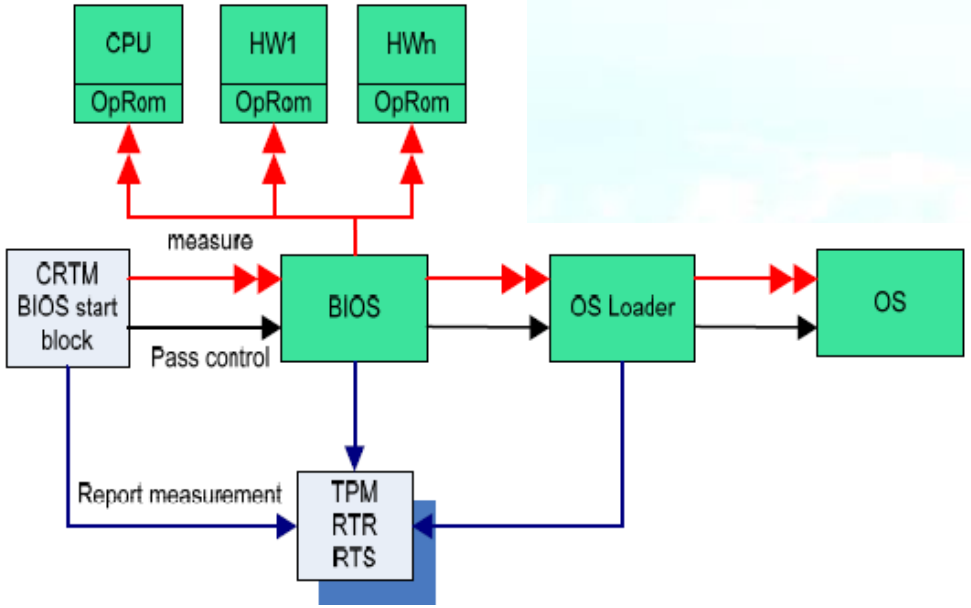
Technologie TPM
Mesure d'intégrité

- Principe de la mesure



Technologie TPM
Mesure d'intégrité

- Chipset utilise les PCR 0 à 7
- Cf. spécifications PC Client du TCG



Technologie TPM
Scellement

- Scellement = chiffrement conditionnel
- Si $Etat_{PCR\ scellement} = Etat_{PCR\ descellement}$
- Alors libération de la donnée scellée
- Conditionnement par rapport à 1 ou plusieurs PCR

Technologie TPM
Attestation distante

- Envoi à un tiers distant :
  - Du fichier SML
  - Des PCR signés par le TPM
- Le tiers distant :
  - Authentifie les PCR
  - Vérifie l'intégrité du SML à l'aide des PCR
  - Vérifie les mesures dans le SML à l'aide d'une base de référence

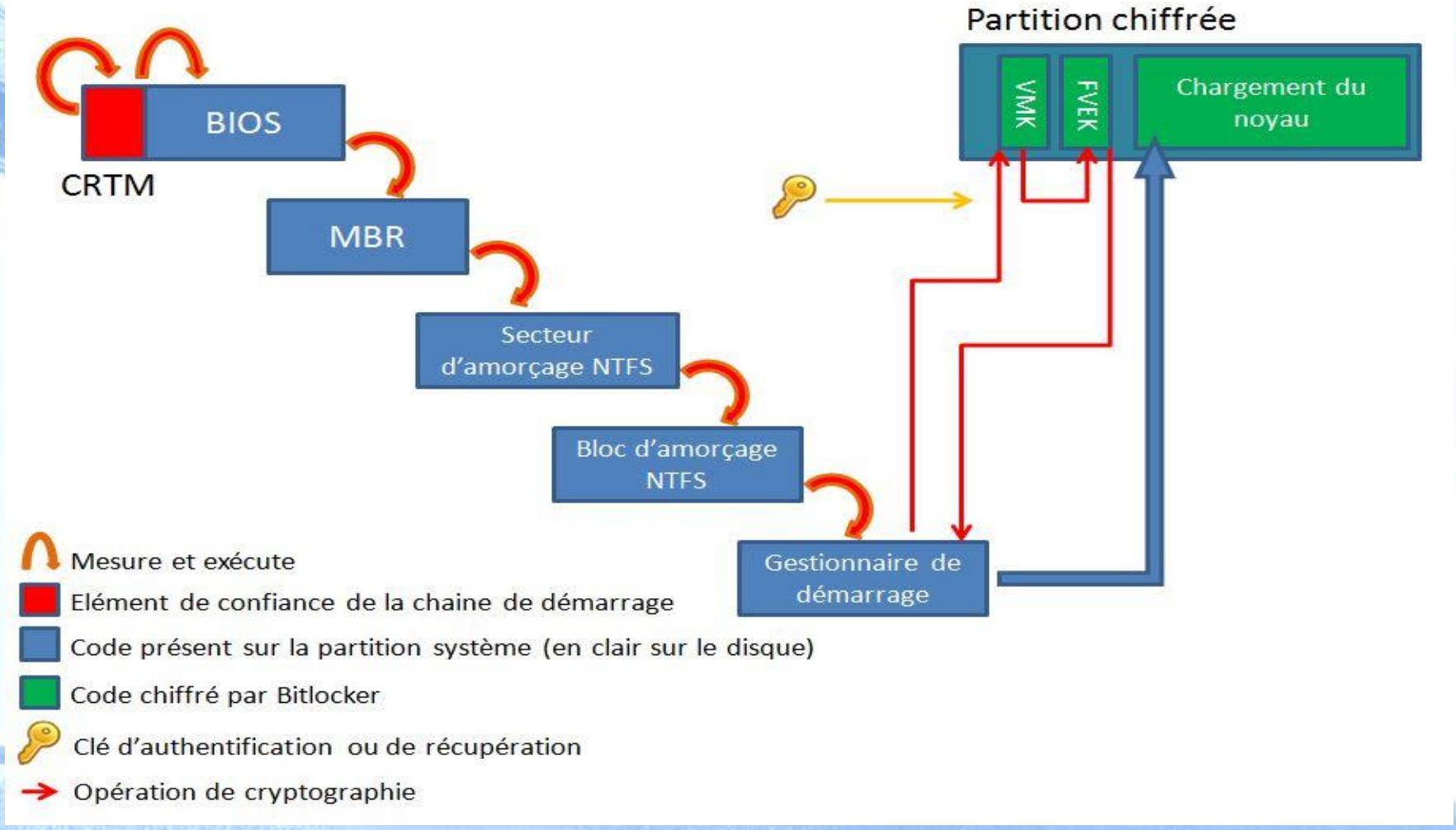
Produit utilisant le TPM

- Bitlocker (Microsoft)
- Trusted Grub (IBM)
- IMA (IBM)
- IMA-appraisal (IBM)
- Trusted and Encrypted Keys (IBM)
- EVM (IBM)

Bitlocker

- Windows Vista / 2008 / 7
- Chiffrement de masse
- Scellement de la clé maître
  - Vérification implicite de l'intégrité de la chaîne de démarrage
- Plusieurs modes :
  - TPM seul
  - TPM + PIN
  - USB
  - TPM + USB
  - TPM + USB + PIN
- Ne protège pas contre :
  - La corruption de l'OS au runtime
  - The Evil Maid

Bitlocker


Partition chiffrée
VMK
FVEK
Chargement du noyau
BIOS
CRTM
MBR
Secteur d'amorçage NTFS
Bloc d'amorçage NTFS
Gestionnaire de démarrage
Mesure et exécute
Elément de confiance de la chaine de démarrage
Code présent sur la partition système (en clair sur le disque)
Code chiffré par Bitlocker
Clé d'authentification ou de récupération
Opération de cryptographie

Trusted Grub

- Patch pour Grub 1
- Mesure uniquement
  - stage 1.5
  - stage 2
  - menu.lst
  - commande de démarrage
  - kernel
  - initrd
- `checkfile` pour certaines versions

IMA

- Integrity Measurement Architecture
- Module Linux (intégré depuis 2.6.30)
- Initialisation des mesures par l'état des PCR 0-7
- Mesure à chaque chargement de binaires :
  - Pilotes
  - Exécutables
  - Librairies dynamiques
- Entretient un fichier SML
- Vérification des pertes d'intégrité
- Passif

IMA-appraisal

- Patch noyau pour IMA
- Ajout d'un attribut étendu sur les fichiers : `security.ima`
- Stockage du SHA1 du fichier dans cet attribut
- Vérification du haché à chaque accès

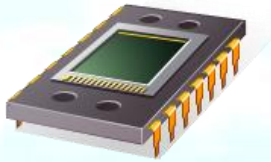
Trusted and Encrypted Keys

- Modules pour le gestionnaire de clés du noyau Linux (2.6.38)
  - Trusted Key :
    - ✓ clé AES scellée par le TPM
    - ✓ Interface flexible :
      - Possibilité de resceller
      - Possibilité de définir la clé parente (par défaut : SRK et mode de passe connu)
  - Encrypted Key (EVM Key) :
    - ✓ clé AES protégée par une autre clé (une *trusted* clé par exemple)
- Première étape pour EVM

EVM

- Extended Verification Module
- Patch noyau
- S'appuie sur Trusted and Encrypted Keys
  - Trusted Key
  - Encrypted Key (EVM Key) : protégée par la Trusted Key
- Ajout d'un attribut étendu : `security.evm`
  - Stockage du HMAC de certains attributs étendus

EVM



scellement

Trusted key

EVM key

HMAC-SHA1

Fichiers

Attributs étendus

security.ima
security.selinux
security.SMACK64
security.capability
security.evm

Autres produits

- eCryptfs :
  - Chiffrement du système de fichiers
  - Peut utiliser le scellement
- Outils de Wave Embassy sous Windows
  - Stockage sécurisé avec le TPM
  - Procédures de recouvrement des clés
- HP Tools
  - Stockage sécurisé avec le TPM

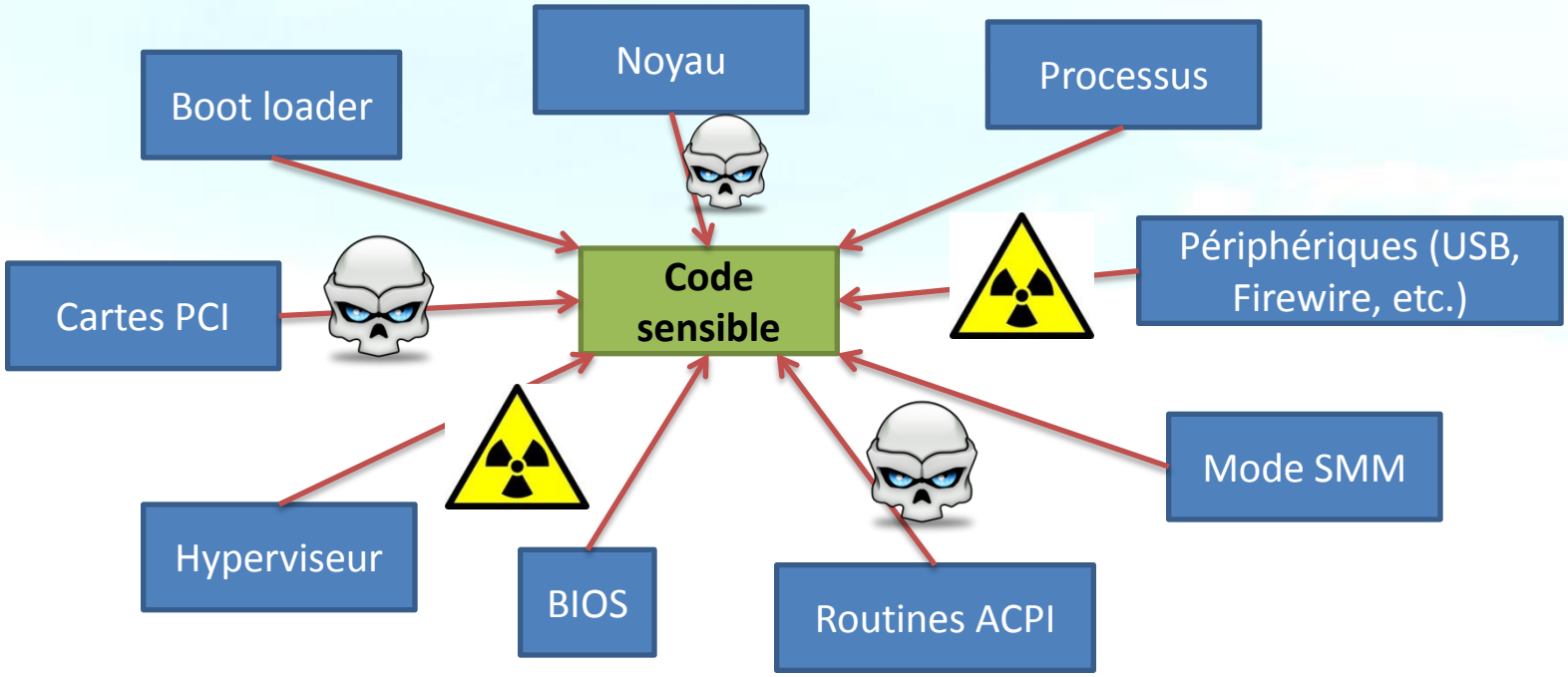
Bilan

- Scellement et stockage sécurisé majoritairement utilisés
- IBM très présent sur les architectures protégées en intégrité
- Attestation distante inexistante sur des produits finis
  - A l'exception de Sirrix (Trusted VPN)

Intel TXT

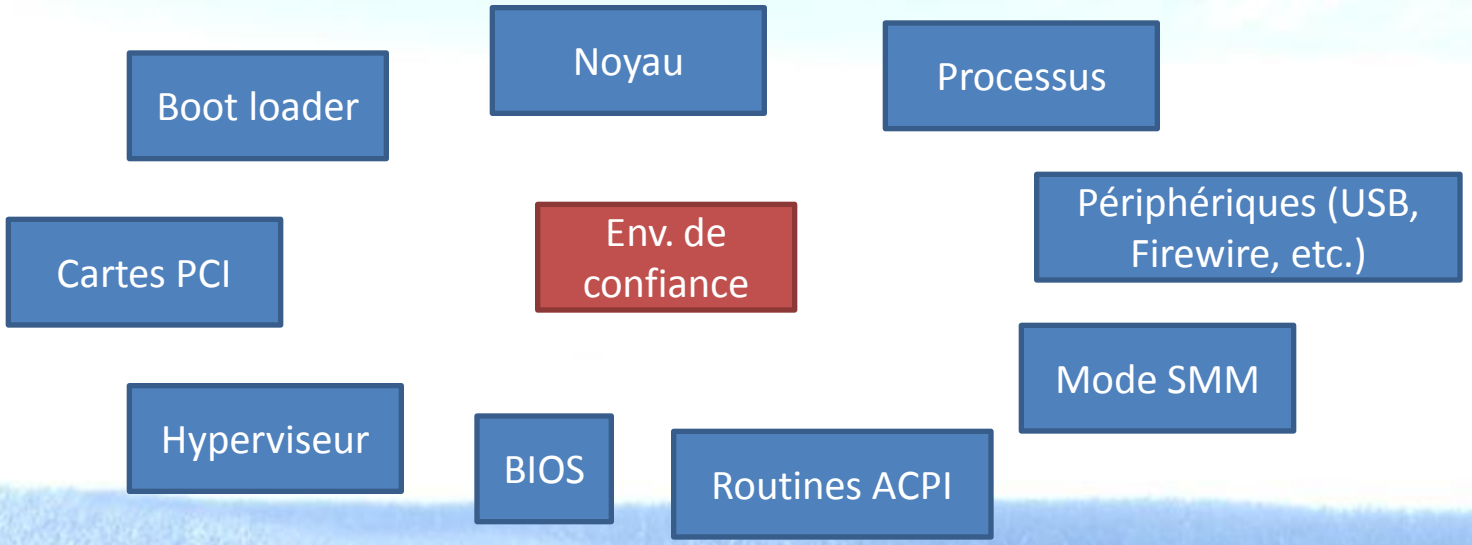
Intel TXT
Introduction

- Comment protéger l'exécution d'un code sensible dans un environnement non de confiance ?



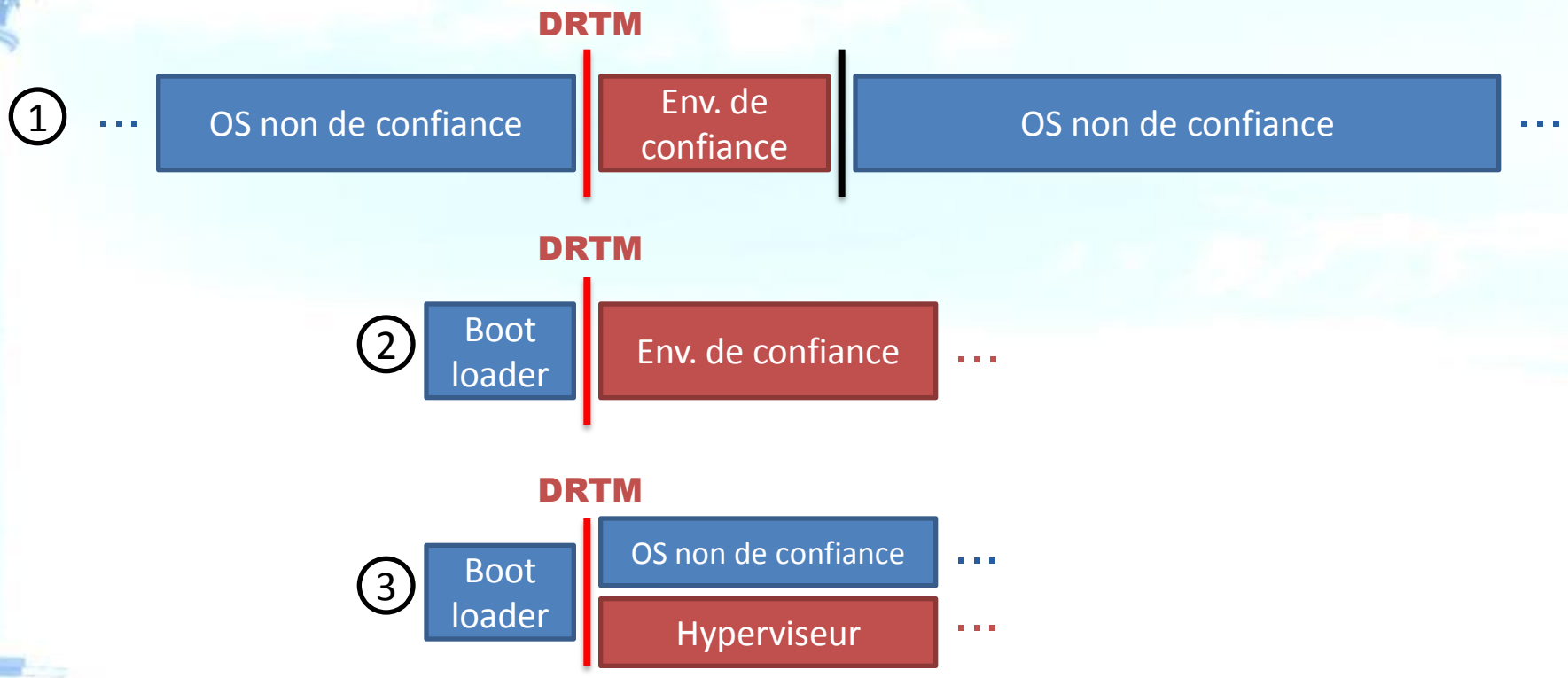
Intel TXT
Objectifs

- Objectifs
  - Démarrage à chaud (*Dynamic Launch*) d'un environnement de confiance
  - Protections mémoire de l'environnement, afin d'éviter toute compromission depuis l'extérieur
  - Vérification d'intégrité au démarrage de l'environnement vis-à-vis d'une politique de sécurité (optionnel)



Intel TXT
Cas d'usages

- Exemples de cas d'usages d'Intel TXT



Intel TXT
Technologies

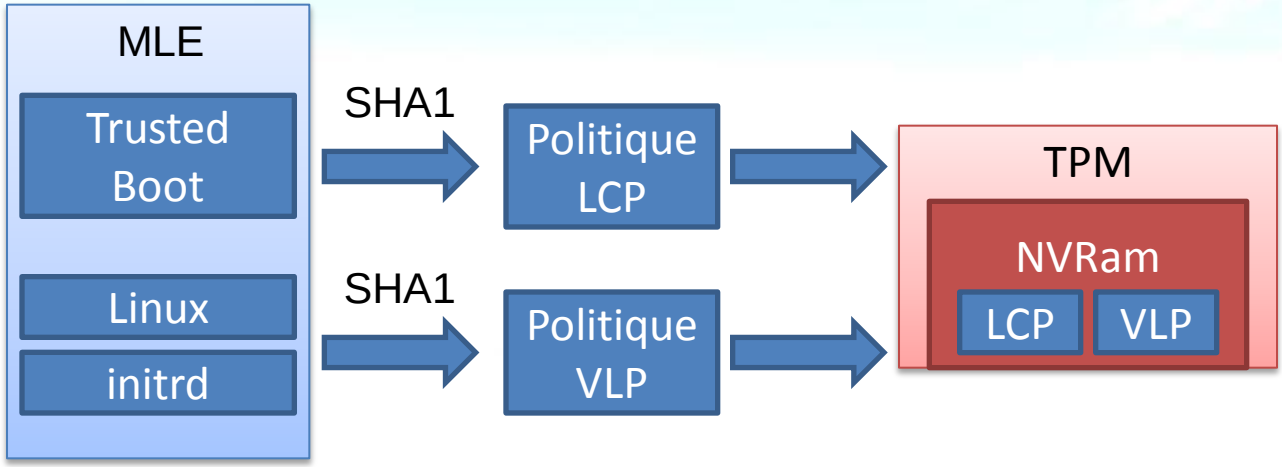
- Technologies sous-jacentes
  - Puce TPM 1.2
  - Processeur supportant la virtualisation matérielle (Intel VT-x)
  - Processeur supportant l'extension SMX (Safer Mode Extensions)
  - Chipset supportant TXT et l'IOMMU (Intel VT-d)

Intel TXT
Trusted Boot

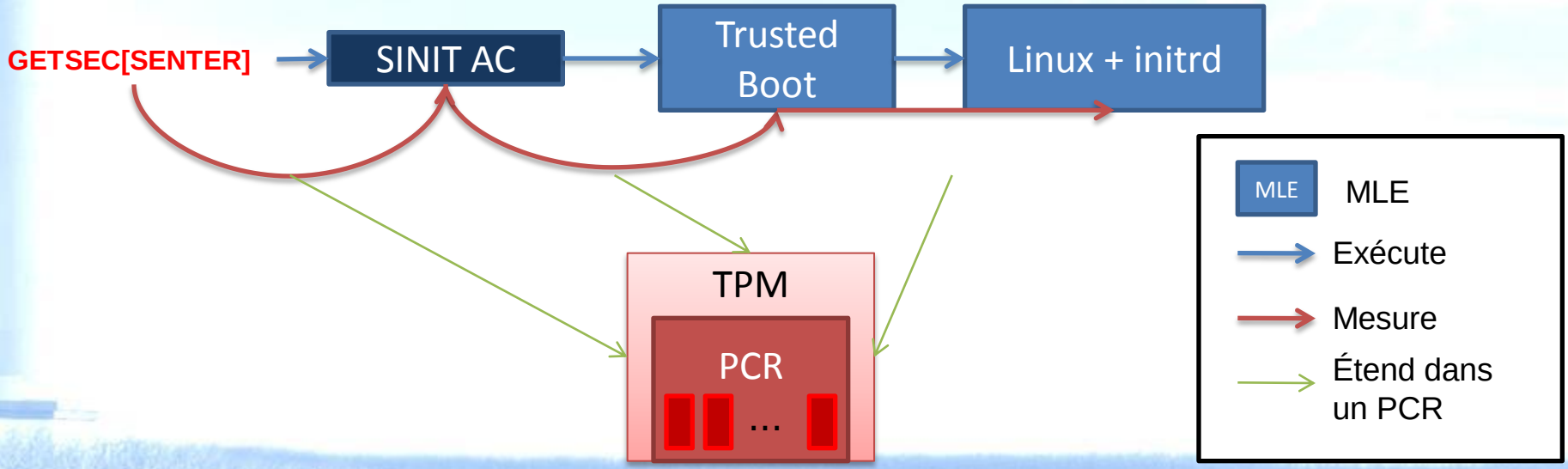
- Trusted Boot
  - Principale implémentation d'Intel TXT
  - Objectif : réaliser une vérification d'intégrité (*Verified Launch*) de l'OS lors de son démarrage
  - OS supportés :
    - ✓ Linux : support d'Intel TXT depuis la version 2.6.32
    - ✓ Xen (depuis la 3.2)

Intel TXT
Définition d'une politique LCP

- Définition des politiques de sécurité
  - Intégrité d'un état connu sauvegardée dans 2 politiques de sécurité
    - ✓ LCP : *Launch Control Policy*
    - ✓ VLP : *Verified Launch Policy*
  - LCP et VLP stockées de manière sécurisée dans le TPM par le propriétaire



- Processus de mesure d'intégrité de l'environnement MLE
  - Séquence initiée par l'instruction GETSEC[SENTER] du jeu d'instructions SMX
  - Mesure puis exécution du code SINIT AC
  - Mesure puis exécution du Secure Loader (Trusted Boot)
  - Mesure puis exécution de Linux



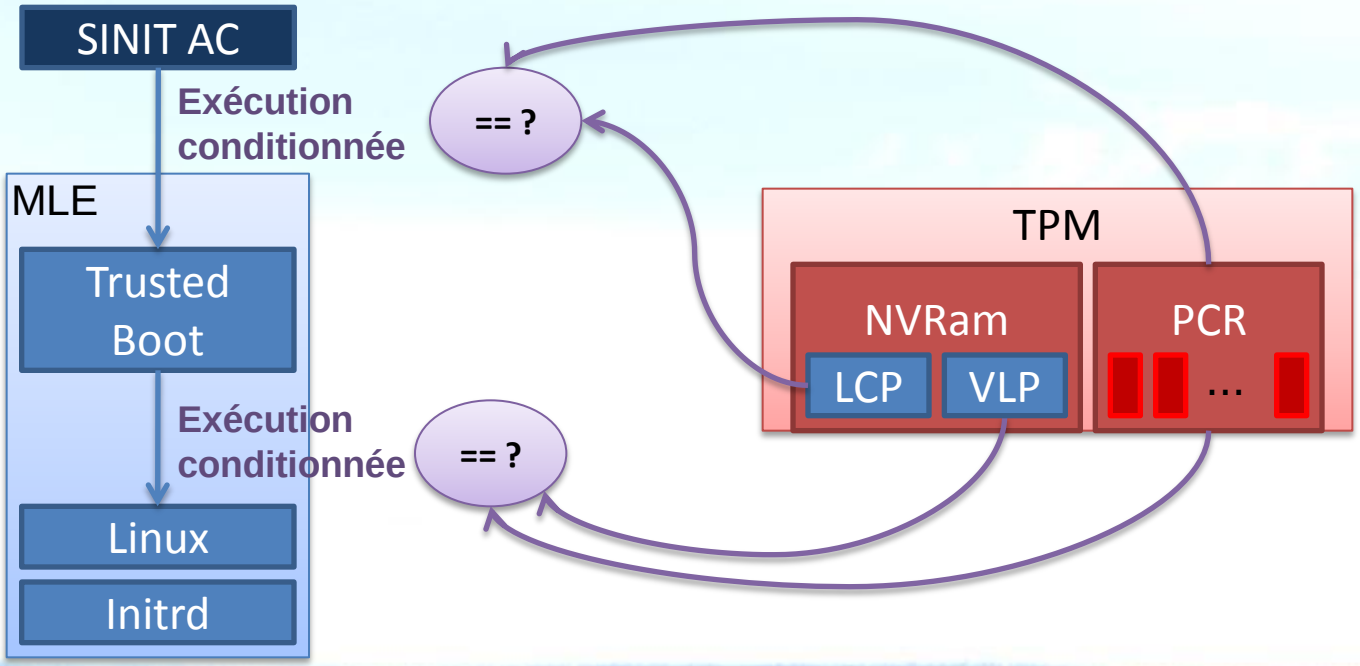
Intel TXT
Exemple de configuration de Grub

- Exemple de configuration de Grub avec Trusted Boot

```
title Trusted Linux
            root (hd0,1)
            kernel /tboot.gz
            module /vmlinuz-2.6.32 root=/dev/sda1 ro
            module /initrd-2.6.32
            module /Q45_SINIT_19.BIN
```

Intel TXT
Verified Launch

- Vérification d'intégrité au démarrage (*Verified Launch*)
  - Permet de conditionner le lancement de l'environnement de confiance en fonction des politiques LCP et VLP
  - Possibilité de continuer ou stopper l'exécution en cas d'intégrité non conforme



Intel TXT
Exemple de vérification d'intégrité

- Exemple de vérification avec Trusted Boot

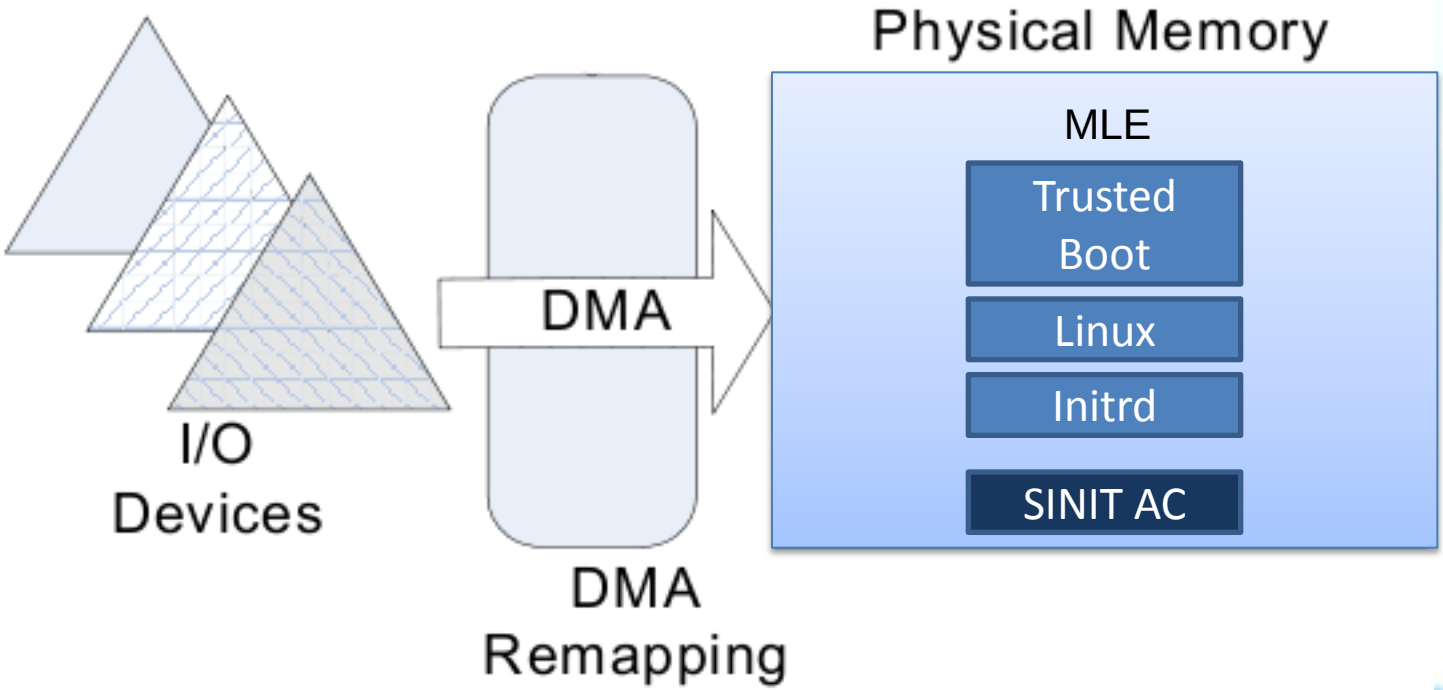
```
TBOOT: verifying module "/boot/vmlinuz root=/dev/sda1 ro"
TBOOT:        OK : 9e e1 ff 2f c0 92 c8 d0 39 95 2c 3c e4 a3 4c d5 da ce 5e e4
TBOOT: verifying module "/boot/initrd.img"
TBOOT:        OK : d7 c3 65 07 cc b2 b4 42 34 5d 8f 56 24 4a 1e cb 0e 21 3e 19
TBOOT: all modules are verified
```

Intel TXT
Le cloisonnement

- Les protections assurées au lancement du MLE
  - Désactivation des cœurs/processeurs secondaires
  - Le code du *Secure Loader* (Trusted Boot) est chargé dans le cache du processeur avant sa mesure et son exécution
  - Désactivation des interruptions (INIT, NMI et SMI comprises)
  - Activation de la protection IOMMU contre les accès DMA
  - Blocage des mécanismes de débogage matériel
  - Communication avec le TPM sur la localité 4 (niveau de privilège maximum, non forgeable de manière logiciel)

Intel TXT
IOMMU

- Protection IOMMU
  - Mise en œuvre d'une MMU dédiée aux I/O afin de bloquer l'accès DMA des périphériques et cartes PCI aux zones mémoires du MLE



Intel TXT
Le code SINIT AC

- Quelques mots sur le code SINIT AC
  - Binaire fourni par Intel, spécifique au chipset et/ou au processeur, chargé de vérifier la configuration du chipset/processeur
  - Signé numériquement par Intel (hash de la clé publique dans un registre public du chipset)
  - Disponible, pour chaque chipset, sur le site de Trusted Boot
  - Parfois directement présent dans le BIOS (ex : DQ45CB)
  - Le code du SINIT AC est chargé dans le cache ACRAM du processeur, avant d'être authentifié puis exécuté

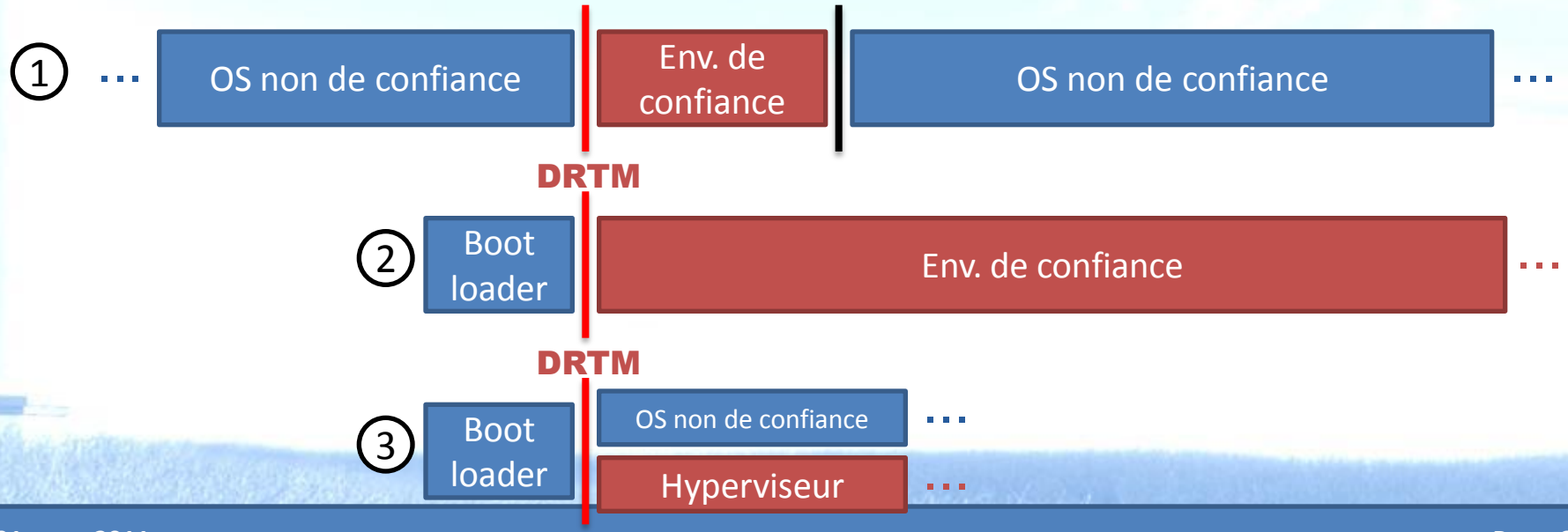
Intel TXT
Robustesse ?

- Attaques publiques contre Intel TXT
  - Vecteur ACPI (Duflot & Levillain, CanSecWest 2009 et SSTIC 2009)
  - Vecteur SMM (Duflot & Levillain, CanSecWest 2009 et SSTIC 2009 ; Wojtczuk & Rutkowska, BlackHat DC 2009)
  - Tables IOMMU (Wojtczuk & Rutkowska, Blog ITL 2009)

- Réponses possibles/proposées
  - Intel a publié un patch pour l'attaque IOMMU
  - Réaliser les opérations sensibles dans la partie totalement cloisonnée du MLE (interruptions SMI ignorées)
  - Nécessité de prendre en compte les routines SMI et ACPI dans le processus de mesure du DRTM
  - Nouveaux processeurs d'Intel (et d'AMD) incluant un moniteur de mode SMM (STM : *SMI Transfer Monitor*)

Intel TXT
Les usages

- Usages actuels d'Intel TXT
  - Essentiellement académique (projets Flicker, P-MAPS et TrustVisor)
  - Supporté dans Linux, depuis la version 2.6.32 (en association avec Tboot)
  - Supporté dans Xen (en association avec TBoot)
  - Présent dans VMWare ESXi 4.1 pour la vérification d'intégrité de l'hyperviseur au démarrage
- Types d'usages potentiellement réalisables :

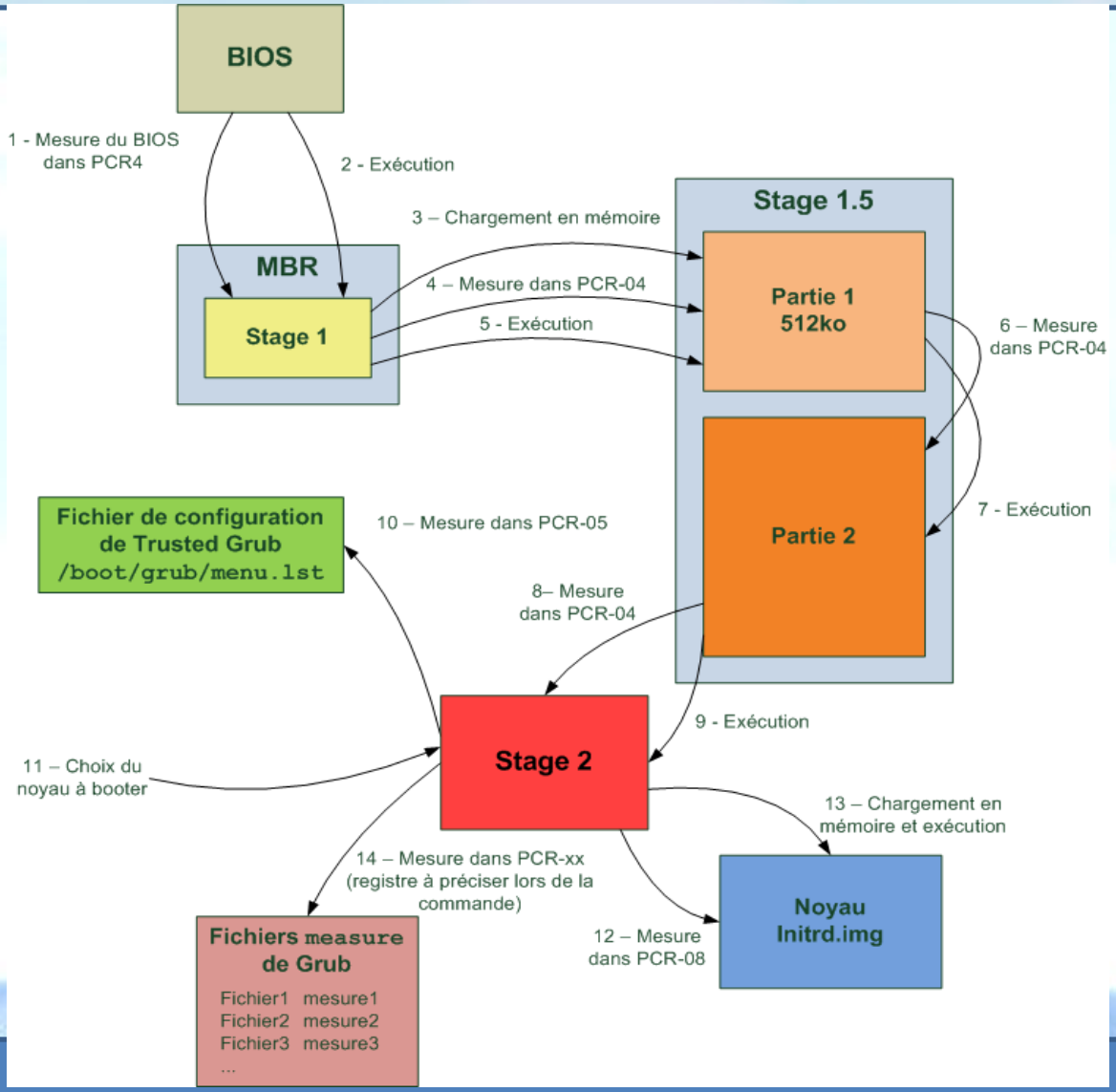


Intel TXT
Conclusion

- Conclusion
  - Technologie complexe à appréhender (nécessite de comprendre les interactions bas-niveau entre chipset/processeur/TPM)
  - Technologie prometteuse, mais encore trop peu utilisée
  - Malgré quelques vulnérabilités de jeunesse, TXT permet de renforcer significativement la sécurité d'un poste local :
    - ✓ Vérification d'intégrité au démarrage et/ou au *runtime*
    - ✓ Protection d'un environnement d'exécution sensible
  - Possibilité de nouveaux modèles de sécurité sur x86

Annexes

Annexe


BIOS
1 - Mesure du BIOS dans PCR4
2 - Exécution
MBR
Stage 1
3 – Chargement en mémoire
4 – Mesure dans PCR-04
5 - Exécution
Stage 1.5
Partie 1 512ko
Partie 2
6 – Mesure dans PCR-04
7 - Exécution
Fichier de configuration de Trusted Grub /boot/grub/menu.lst
10 – Mesure dans PCR-05
8– Mesure dans PCR-04
9 - Exécution
Stage 2
11 – Choix du noyau à booter
13 – Chargement en mémoire et exécution
14 – Mesure dans PCR-xx (registre à préciser lors de la commande)
12 – Mesure dans PCR-08
Noyau Initrd.img
Fichiers measure de Grub
Fichier1 mesure1
Fichier2 mesure2
Fichier3 mesure3
...

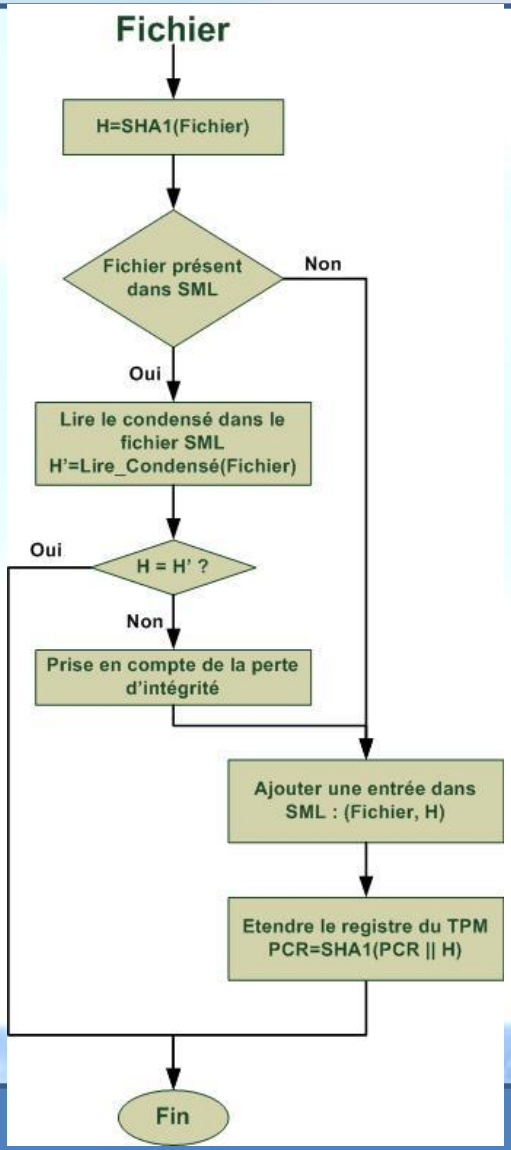
Annexe

```
testtpm:/home/ggx#cat /sys/class/misc/tpm0/device/pcrs
PCR-00: A2 A5 93 F8 A7 28 BE 85 E7 A3 F6 ED 41 B2 19 6C A1 CB E2 41
PCR-01: B6 3A CB 6C C6 C1 E9 1E D8 73 89 4F 8C F0 BE 74 50 5C 0A 74
PCR-02: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-03: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-04: 26 20 D1 0C B5 F0 79 6A 54 9D 80 F8 E9 16 F9 79 0A 41 25 9E
PCR-05: 6B 0D F7 D9 48 42 8D AB 1D 44 86 64 19 24 07 6B E8 6E 9E 10
PCR-06: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-07: 3A 3F 78 0F 11 A4 B4 99 69 FC AA 80 CD 6E 39 57 C3 3B 22 75
PCR-08: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-09: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-10: 6F E7 47 71 88 5F 7A A3 25 E1 49 07 49 D2 4A ED 9C BB 0F F1
PCR-11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR-17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR-23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Annexe
IMA – Fichier SML

```
10 03ee0cce68447c947f8e8eb37881c27570347fa4 boot_aggregate
10 ea8239dfed9dd11bd538f9c3234e0d7b71672fff /bin/sh
10 ebb4f3db0b83c1e717e3d05f702e4608a9c2ea08 /lib/ld-linux.so.2
10 671aba5cb6df951463e57c963e8c327fc6cfb5ab /lib/libcrypt.so.1
10 445babe91e586090cb7f9782b44ba115ceec6b7f /lib/libm.so.6
10 411ab19e995e06b1d7378e1daea9926ccba1ea20 /lib/libc.so.6
10 68b297cd8fe07a3e54e6ce9d2e66afa799e472a3 /sbin/depmod
10 407285ba377ea035f2ff6d61c47ead8befa7cd5f /sbin/modprobe
10 ee3295941fd593c95c03268ff961036d248dd8c0 thermal_sys
10 432f769f5864adc1bc5daf17fbdf2e8c99e7ffd4 fan
10 ccfaf0b02d678c9b3f6eba54968e832505a50793 processor
10 033d934e266c6aff719278a28d6dd7e74fe70c6c thermal
10 b3f9dc0a1cc001f88d43609c7dea156d280deca8 /sbin/udevd
10 91700dcba332de54094beef00fea0e11dd414b9f /lib/libselinux.so.1
10 8abd11a221cd68b6f544e8a4b74a59dfbffd839c /lib/libdl.so.2
10 cea55a2390cfc6c4947e980be3aa457cea1ec335 /lib/libsepol.so.1
10 78750ed652666c01f4e7de51768b673e217c536b /sbin/udevtrigger
10 07bcb87e7acca43b0f3f0ba67f480402519f87b3 /sbin/udevsettle
10 08e158dc445c50aa32b1e1adbed2a3f13828ee2d sd_mod
10 1cc363d95a64c3ebf26127355d894724b0415a53 /bin/true
10 cdce54d429fe002b7377da94c74c6aa1014f6f4f /lib/klibc-rOj3PRLKBA9FcF5ZuoqKQLmOWcA.so
10 ecbc9f0925e1f7ec370fec9082ab30d4111f9df2 usbcore
10 8d4105ed57deb1a417805a88a1a5c3df332273ba ieee1394
10 8631dc01e9e12aa1af8be015874d2cf53cd8475c ehci_hcd
10 ed1937e5f4bf1e3c07949df24981d538bce24618 /lib/udev/usb_id
10 5546a5dcad4b9f77ab05d310daa3ae1a3d503e69 ide_core
10 8c818fb30300cf0ea4c278a52c3c671f50f2a529 /lib/udev/scsi_id
10 df2c5070d5d3322009086cd3d464d30473b0666e libphy
10 3988ef17d2e9264a671d0c84135a7dda852cd977 /lib/udev/edd_id
10 f774ca3c6a5a55ace882946548d0bf6f6bf03e4a ohci1394
10 d7190161192e5ce5ea32405fe5abdcf81c4221f0 /lib/udev/vol_id
10 e76488eaae2df61955da3bb3e8f087212a99307c piix
10 fc89d5eb576dcad375163f710807eb1e6f4ebbe3 /lib/libvolume_id.so.0
10 d34755fcb7c6e18480ef90c2eff55c52d7f7e0a9 uhci_hcd
10 ff82002ad570af494266f7b16ba4a9d5080cefc2 tg3
10 0d339ef60e3d2bd4d9ec11f38972ff5eac9e4eb3 ata_generic
10 32b6258f0e46ef4669d3e1573096244acb078181 usbhid
```

Annexe


Fichier
H=SHA1(Fichier)
Fichier présent dans SML
Non
Oui
Lire le condensé dans le fichier SML
H'=Lire_Condensé(Fichier)
Oui
H = H' ?
Non
Prise en compte de la perte d'intégrité
Ajouter une entrée dans SML : (Fichier, H)
Etendre le registre du TPM
PCR=SHA1(PCR || H)
Fin