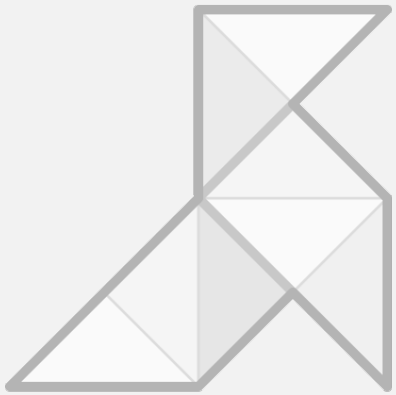
Démarrage vérifié et logiciels libres : réconcilier liberté et sécurité



Embedded Freedom

Paul Kocialkowski
contact@paulk.fr

Dimanche 19 novembre 2017

Capitole du Libre

Introduction : démarrage sur x86 et ARM

Évolution des logiciels de démarrage : x86

- Années 1980 :
  - **Initialisation** sommaire du matériel
  - **Chargement** du système d'exploitation
  - **Appels** au BIOS par interruptions (utilisé par CP/M, DOS)
  - Objectif : **abstraction du matériel**
  - Mémoire en **lecture seule**

Évolution des logiciels de démarrage : x86

- Années 1980 :
  - **Initialisation** sommaire du matériel
  - **Chargement** du système d'exploitation
  - **Appels** au BIOS par interruptions (utilisé par CP/M, DOS)
  - Objectif : **abstraction du matériel**
  - Mémoire en **lecture seule**
- Années 1990 :
  - **Complexité** grandissante du matériel
  - Prise en charge dans le système, **initialisation seulement**
  - Mémoire **réinscriptible** (mises à jour)

Évolution des logiciels de démarrage : x86

- Années 1980 :
  - **Initialisation** sommaire du matériel
  - **Chargement** du système d'exploitation
  - **Appels** au BIOS par interruptions (utilisé par CP/M, DOS)
  - Objectif : **abstraction du matériel**
  - Mémoire en **lecture seule**
- Années 1990 :
  - **Complexité** grandissante du matériel
  - Prise en charge dans le système, **initialisation seulement**
  - Mémoire **réinscriptible** (mises à jour)
- Années 2000 :
  - Services **pendant l'exécution** (SMM, ACPI)
  - Unified Extensible Firmware Interface (UEFI)

  abstraction du matériel

Évolution des logiciels de démarrage : ARM

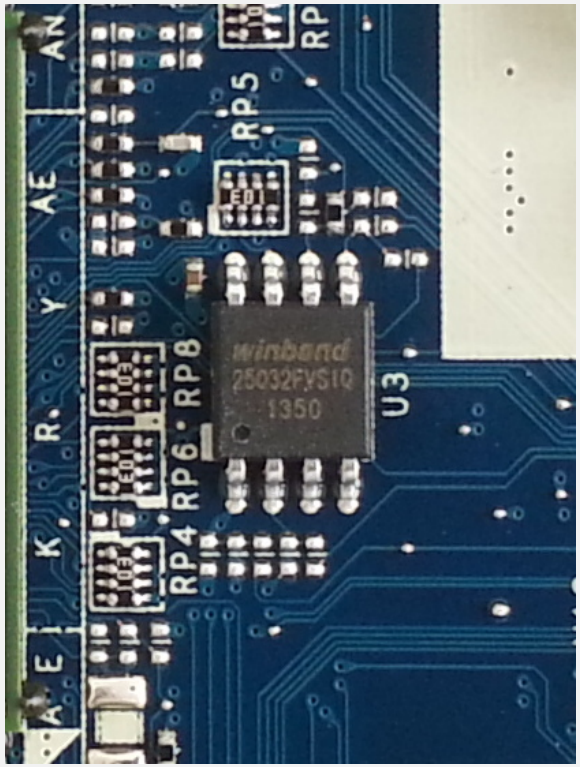
- Années 2000 :
  - **Initialisation** du matériel
  - Prise en charge **dans le système**
  - Pas de services **pendant l'exécution**

Évolution des logiciels de démarrage : ARM

- Années 2000 :
  - **Initialisation** du matériel
  - Prise en charge **dans le système**
  - Pas de services **pendant l'exécution**
- Années 2010 :
  - Services **pendant l'exécution** (ATF, PSCI)
  - Unified Extensible Firmware Interface (UEFI)
    unification de l'interface pour serveurs ARMv8

Medium de stockage physique

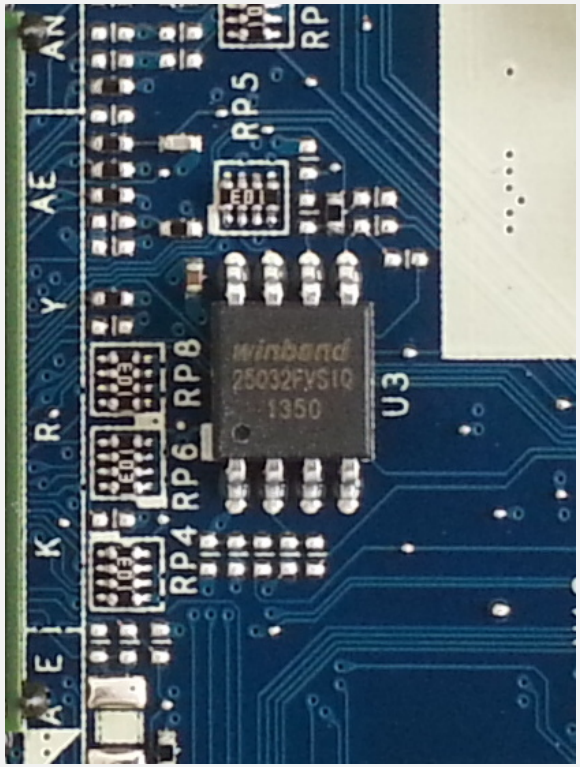
Stockage du **logiciel de démarrage** :



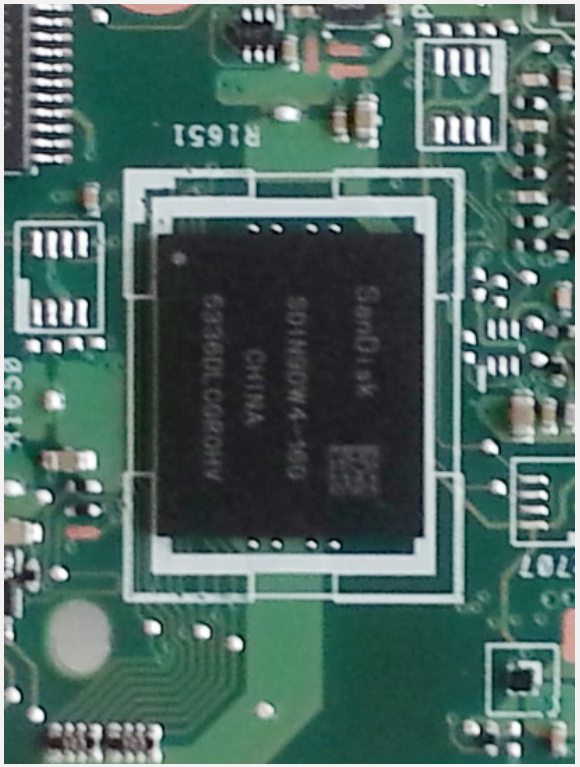
Flash SPI (x86, ARM)

Medium de stockage physique

Stockage du **logiciel de démarrage** :

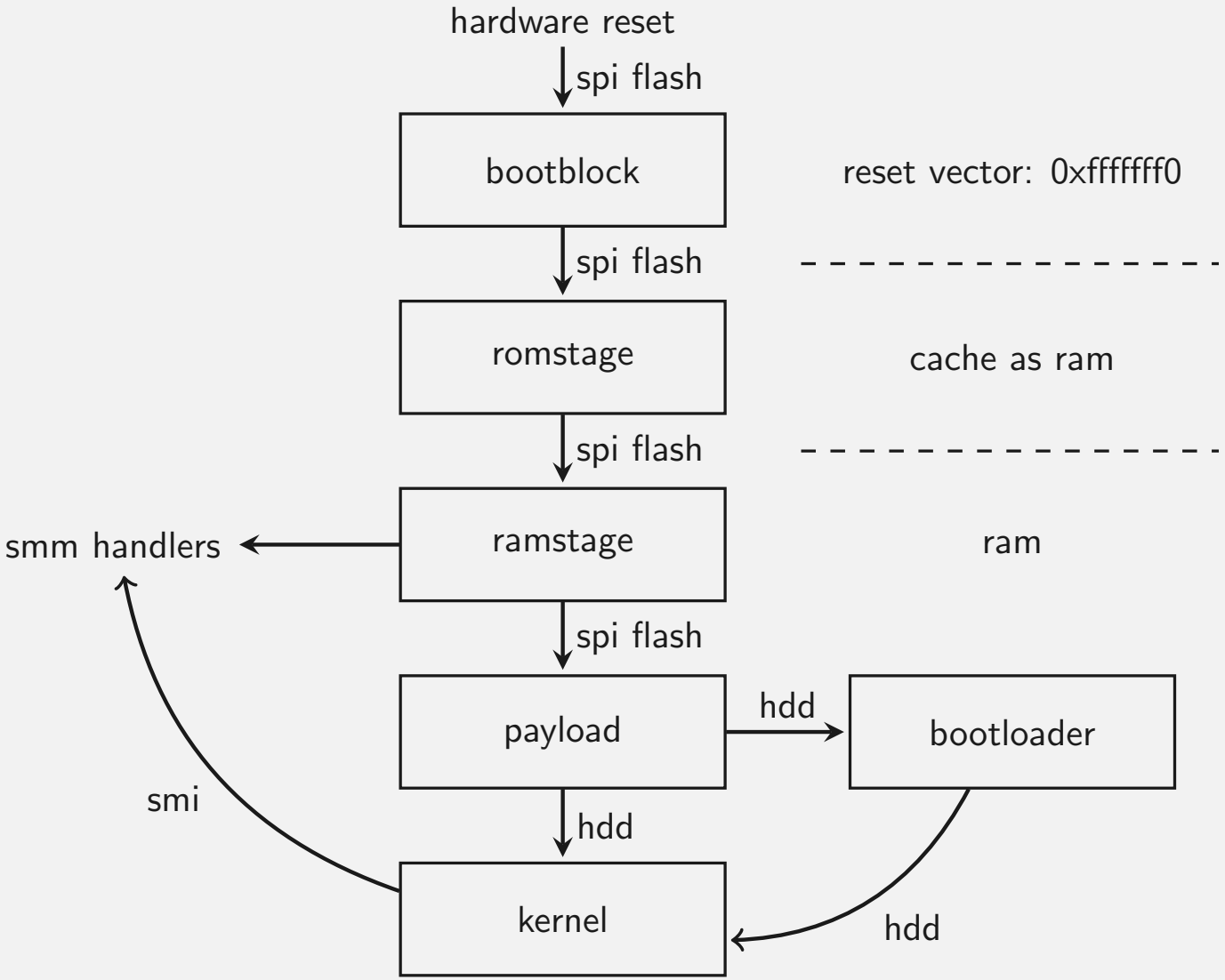


Flash SPI (x86, ARM)

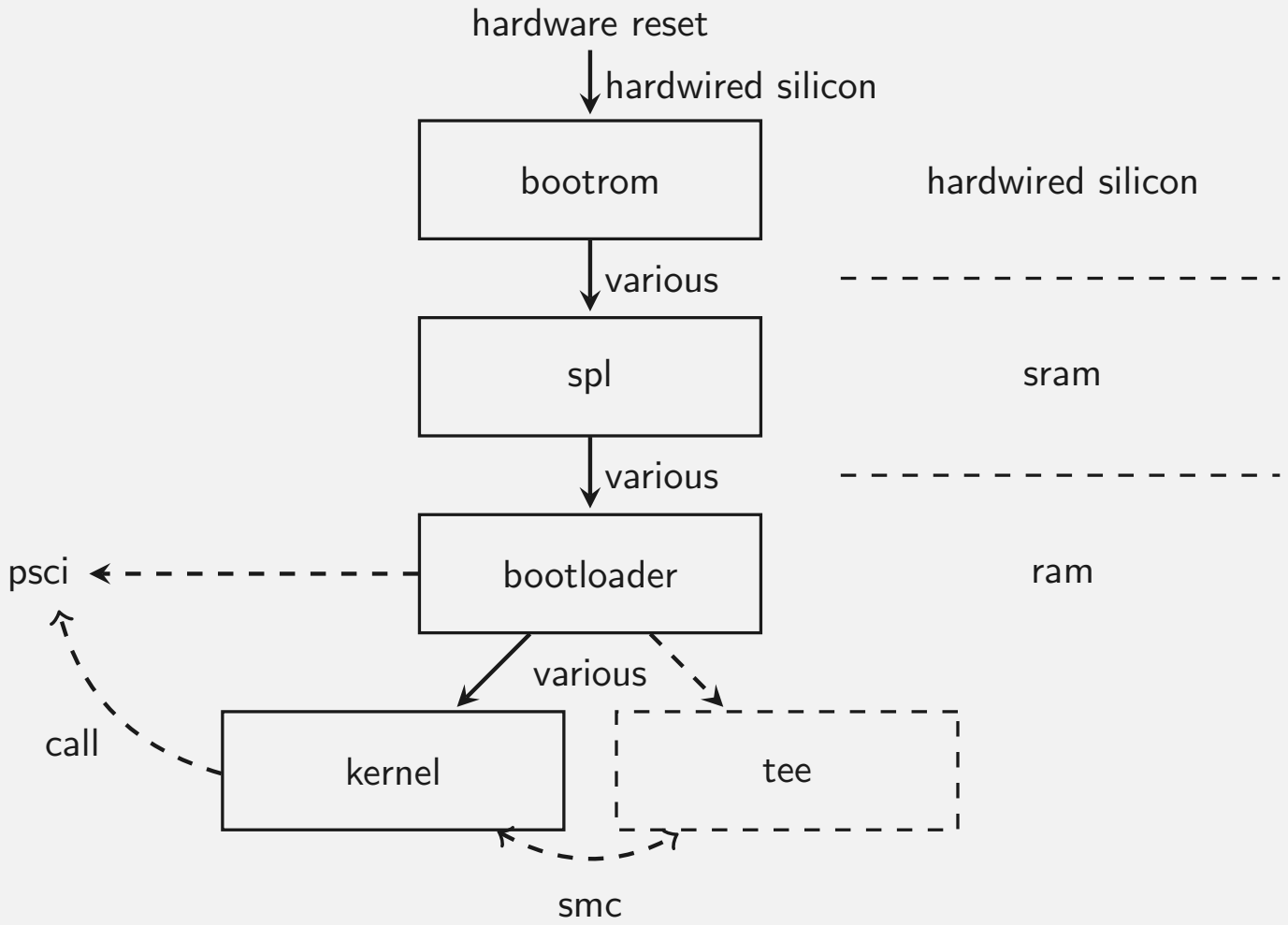


NAND, eMMC (ARM)

Processus de démarrage : x86



Processus de démarrage : ARM



Environnements d'exécution privilégiés

Exécution **privilégiée** :

- Exécution sur le **même processeur**
- Niveau de permissions **le plus élevé**
- Interrompt et **prends le pas** sur le système

Environnements d'exécution privilégiés

Exécution **privilégiée** :

- Exécution sur le **même processeur**
- Niveau de permissions **le plus élevé**
- Interrompt et **prends le pas** sur le système

Utilisation :

- Gestion d'énergie
- Prise en charge matérielle spécifique
- Émulation de périphériques
- Opérations sensibles (sécurité, DRM)
  pas confiance dans le système

Environnements d'exécution privilégiés : SMM

System Management Mode (SMM)

Exécution **privilégiée** (x86) :

- Exécution au niveau *Ring -2* du CPU

Environnements d'exécution privilégiés : SMM

System Management Mode (SMM)

Exécution **privilégiée** (x86) :

- Exécution au niveau *Ring -2* du CPU

Passage en SMM :

- Signal **SMI#**
- SMI émis **en logiciel**
- Accès **I/O protégé**

Environnements d'exécution privilégiés : SMM

System Management Mode (SMM)

Exécution **privilégiée** (x86) :

- Exécution au niveau *Ring -2* du CPU

Passage en SMM :

- Signal **SMI#**
- SMI émis **en logiciel**
- Accès **I/O protégé**

Implémentation :

- **Au sein** du logiciel de démarrage

Environnements d'exécution privilégiés : TrustZone

Trusted Execution Environments (TEE)

Exécution privilégiée (ARM) :

- Exécution dans le **monde sécurisé** (*Secure World*)
- **Verrouillage** de zones mémoire

Environnements d'exécution privilégiés : TrustZone

Trusted Execution Environments (TEE)

Exécution privilégiée (ARM) :

- Exécution dans le **monde sécurisé** (*Secure World*)
- **Verrouillage** de zones mémoire

Appel au TEE :

- Instruction **SMC** (Secure Monitor Call)

Environnements d'exécution privilégiés : TrustZone

Trusted Execution Environments (TEE)

Exécution privilégiée (ARM) :

- Exécution dans le **monde sécurisé** (*Secure World*)
- **Verrouillage** de zones mémoire

Appel au TEE :

- Instruction **SMC** (Secure Monitor Call)

Implémentation :

- Code **distinct** du logiciel de démarrage
- **Chargée** par le logiciel de démarrage

Vérification du démarrage

Besoin de vérification du démarrage

Surface d'attaque très large :

- Accès aux **données** de l'utilisateur
- Accès aux **ressources matérielles**
- Chargement de **l'environnement d'exécution privilégié** exécuté pendant la durée du vie du système
- Chargement du **système d'exploitation**

Besoin de vérification du démarrage

Surface d'attaque très large :

- Accès aux **données** de l'utilisateur
- Accès aux **ressources matérielles**
- Chargement de **l'environnement d'exécution privilégié** exécuté pendant la durée du vie du système
- Chargement du **système d'exploitation**

Composants :

- Chaque **étage** du/des logiciels(s) de démarrage
- **Environnement d'exécution privilégié** (si séparé)
- **Noyau du système** (ou pas)

Implémentation : démarrage mesuré

Philosophie :

- **Indication d'état** d'intégrité du démarrage
- **Libération d'un secret** si l'état est cohérent

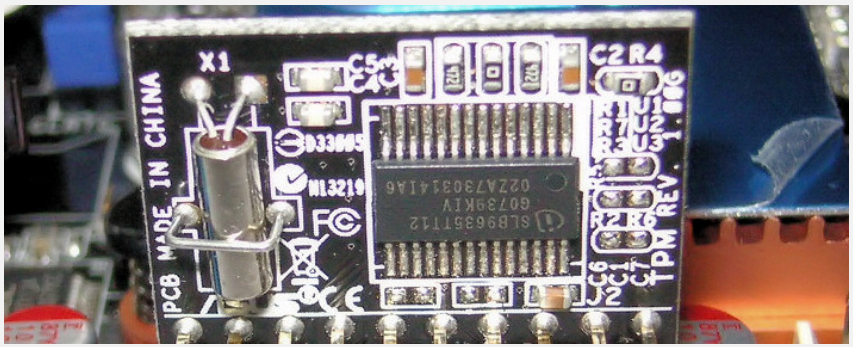
Implémentation : démarrage mesuré

Philosophie :

- **Indication d'état** d'intégrité du démarrage
- **Libération d'un secret** si l'état est cohérent

Utilisation du **TPM** (Trusted Platform Module) :

- Registres *PCR* :
  - Initialisés avec **hash final** et **secret** puis **verrouillés**
  - Opération *hash extend* (avec précédent)
- **Extension du hash** avec chaque composant
- **Racine de confiance** (CRTM), **accès physique**



Implémentation : démarrage vérifié

Philosophie :

- **Vérification** de chaque composant
- **Erreur et arrêt** si mauvaise signature

Implémentation : démarrage vérifié

Philosophie :

- **Vérification** de chaque composant
- **Erreur et arrêt** si mauvaise signature

Implémentation :

- Vérification de **signature** avec **clef publique**
- Clef privée tenue **secrète**
- Implémentations cryptographiques :
  - Implémentations **logicielles**
  - **Coprocesseurs** matériels
  - Utilisation du **TPM**
- **Racine de confiance** (CRTM)

Core Root of Trust and Measurement (CRTM)

Garantir la **racine de confiance**

Intégrité du premier composant :

- **Vérification matérielle** (BootROM) :
  - Code au sein du **design matériel**
  - Clefs en **mémoire OTP**
- Stockage en **lecture seule**
- **Composant intermédiaire** (stockage-CPU) de confiance

Core Root of Trust and Measurement (CRTM)

Garantir la **racine de confiance**

Intégrité du premier composant :

- **Vérification matérielle** (BootROM) :
  - Code au sein du **design matériel**
  - Clefs en **mémoire OTP**
- Stockage en **lecture seule**
- **Composant intermédiaire** (stockage-CPU) de confiance

Sécurité physique des composants :

- Accès au **TPM**
- Accès au **composant intermédiaire**
- Accès au **CPU**

Solutions (inopérantes) adoptées dans l'industrie

Intégrité **sans vérification** :

- Medium du stockage en **lecture seule**
- Ne couvre pas le noyau du système

Solutions (inopérantes) adoptées dans l'industrie

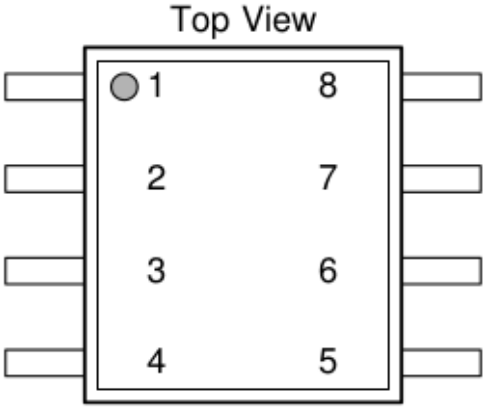
Intégrité **sans vérification** :

- Medium du stockage en **lecture seule**
- Ne couvre pas le noyau du système

Implémentation par $\overline{WP}$/GPIO :

- Pin $\overline{WP}$ des puces flash SPI
- Contrôlé par **GPIO** (mises à jour)
- Simple à contrôler flashrom/board_enable.c :

```
static int intel_piix4_gpo27_lower(void)
{
        return intel_piix4_gpo_set(27, 0);
}
```



1. $\overline{CS}$
2. *DO*
3. $\overline{WP}$
4. *GND*
8. *VCC*
7. $\overline{HOLD}$
6. *CLK*
5. *DI*

Solutions (inopérantes) adoptées dans l'industrie

Approches basées sur les **plateformes** :

- Accès SPI **désactivé/lecture seule** et **verrouillé**
- Zones **protégées** (PRx)
- Ne concerne pas :
  - Accès privilégiés (SMM, TEE)
  - Contrôleurs internes (ME, IMC)
  - Accès physiques au stockage
- Logique **spécifique** à la plateforme

Solutions (inopérantes) adoptées dans l'industrie

Approches basées sur les **plateformes** :

- Accès SPI **désactivé/lecture seule** et **verrouillé**
- Zones **protégées** (PRx)
- Ne concerne pas :
  - Accès privilégiés (SMM, TEE)
  - Contrôleurs internes (ME, IMC)
  - Accès physiques au stockage
- Logique **spécifique** à la plateforme

UEFI Secure Boot :

- Vérification du **système**
- Clefs pour chaque système
- Généralement **désactivable** par accès physique
- Pas plus de vérification du **logiciel de démarrage**

Solutions (opérantes) adoptées dans l'industrie

Vérification **matérielle** (BootROM) :

- Clef en **mémoire OTP**
- Vérification implémentée au sein du **design matériel**
- Racine de confiance **fiable**

Solutions (opérantes) adoptées dans l'industrie

Vérification **matérielle** (BootROM) :

- Clef en **mémoire OTP**
- Vérification implémentée au sein du **design matériel**
- Racine de confiance **fiable**

Implémentations :

- Plateformes **ARM**
- Intel **Bootguard**

Problématiques de liberté et sécurité

Contrôle de la part de l'utilisateur

Problématiques générales :

Clefs de vérification de signatures :

- Déjà **installées** par le fabricant
- Tenues **secrètes** par le fabricant
- **Pas remplaçables** par l'utilisateur

Contrôle de la part de l'utilisateur

Problématiques générales :

Clefs de vérification de signatures :

- Déjà **installées** par le fabricant
- Tenues **secrètes** par le fabricant
- **Pas remplaçables** par l'utilisateur

Confiance envers le fabricant :

- **Dépendance** de l'utilisateur
- **Intérêt** privé, intérêt général et individuel

Conséquences pour l'utilisateur

Point de vue du **logiciel libre** :

- Impossible de **modifier** le logiciel
- Impossible **d'utiliser** du logiciel libre
- Problématiques du **logiciel propriétaire**

Conséquences pour l'utilisateur

Point de vue du **logiciel libre** :

- Impossible de **modifier** le logiciel
- Impossible **d'utiliser** du logiciel libre
- Problématiques du **logiciel propriétaire**

Effet **boule de neige** :

- Logiciel(s) de **démarrage**
- Environnements d'exécution **privilégiés**
- **Système** d'exploitation ?

Conséquences pour l'utilisateur

Point de vue du **logiciel libre** :

- Impossible de **modifier** le logiciel
- Impossible **d'utiliser** du logiciel libre
- Problématiques du **logiciel propriétaire**

Effet **boule de neige** :

- Logiciel(s) de **démarrage**
- Environnements d'exécution **privilégiés**
- **Système** d'exploitation ?

Incompatibilité entre liberté et "sécurité" ?

Problématiques de sécurité

Capacités des composants concernés :

- Accès aux **données** de l'utilisateur
- Accès aux **ressources matérielles**
- Chargement de **l'environnement d'exécution privilégié**

Problématiques de sécurité

Capacités des composants concernés :

- Accès aux **données** de l'utilisateur
- Accès aux **ressources matérielles**
- Chargement de **l'environnement d'exécution privilégié**

Mises à jour des logiciels :

- Possible uniquement **pour le fabricant** (délais, EOL)
- Au-delà du **court terme** ?
- **Vulnérabilités** identifiées, réutilisables

Problématiques de sécurité

Capacités des composants concernés :

- Accès aux **données** de l'utilisateur
- Accès aux **ressources matérielles**
- Chargement de **l'environnement d'exécution privilégié**

Mises à jour des logiciels :

- Possible uniquement **pour le fabricant** (délais, EOL)
- Au-delà du **court terme** ?
- **Vulnérabilités** identifiées, réutilisables

Validation technique du code :

- Aucune capacité de **revue** individuelle et collective
- Aucune **reproductibilité** du code

Motivations de l'industrie

Bilan pour l'utilisateur :

- Perdant sur la **liberté**
- Perdant sur la **sécurité**
- Solutions parfois **inopérantes**

Motivations de l'industrie

Bilan pour l'utilisateur :

- Perdant sur la **liberté**
- Perdant sur la **sécurité**
- Solutions parfois **inopérantes**

Motivations et intérêts des fabricants :

- Accès aux clefs **DRM**
- **Restrictions** de l'utilisateur
  choix du système, cartes Wi-Fi, etc

Extension du problème

Logiciel présent **partout** :

- Processeur principal : démarrage, privilégié, système
- Processeur de gestion : ME, IMC, SMU, BMC...
- Processeurs auxiliaires : GPU, VPU, DSP...
- Contrôleurs : EC, xHCI, multimedia, batterie...
- Périphériques : Wi-Fi, bluetooth, GPS, webcam...

Extension du problème

Logiciel présent **partout** :

- Processeur principal : démarrage, privilégié, système
- Processeur de gestion : ME, IMC, SMU, BMC...
- Processeurs auxiliaires : GPU, VPU, DSP...
- Contrôleurs : EC, xHCI, multimedia, batterie...
- Périphériques : Wi-Fi, bluetooth, GPS, webcam...

Conséquences pour la sécurité :

- Accès aux **données** variable
- Accès au **reste du matériel** variable

Remédiations possibles

État des lieux du logiciel libre

Projets libres de démarrage :

- coreboot
- U-Boot, Barebox

État des lieux du logiciel libre

Projets libres de démarrage :

- coreboot
- U-Boot, Barebox

Composants propriétaires habituels :

- **x86** : Option ROM/VGA BIOS, microcodes CPU
- **Intel x86** : FSP, MRC, ME
- **AMD x86** : IMC, SMU, PSP
- **Micrologiciels** divers : xHCI, ethernet, . . .

État des lieux du logiciel libre

Projets libres de démarrage :

- coreboot
- U-Boot, Barebox

Composants propriétaires habituels :

- **x86** : Option ROM/VGA BIOS, microcodes CPU
- **Intel x86** : FSP, MRC, ME
- **AMD x86** : IMC, SMU, PSP
- **Micrologiciels** divers : xHCI, ethernet, ...

Projets **entièrement libres** :

- Libreboot
- Paper

Liberté sans vérifications de signatures

Plateformes propices :

- Démarrage **libre**
- Pas de **vérifications/clefs du fabricant**
- Prise en charge matérielle **libre**

Liberté sans vérifications de signatures

Plateformes propices :

- Démarrage **libre**
- Pas de **vérifications/clefs du fabricant**
- Prise en charge matérielle **libre**

Exemples :

- i.MX de Freescale/NXP
- A, H et R de Allwinner
- RK de Rockchip
- OMAP, AM de Texas Instruments
- *Tegra* de NVIDIA

Liberté avec vérifications de signatures

Rendre la racine de confiance **à l'utilisateur** :

- Installation des clefs en mémoire OTP :
  - N emplacements disponibles :
    fabricant, *distribution*, *personnel*
  - Bit de révocation par clef

Liberté avec vérifications de signatures

Rendre la racine de confiance **à l'utilisateur** :

- Installation des clefs en mémoire OTP :
  - N emplacements disponibles :
    fabricant, *distribution*, *personnel*
  - Bit de révocation par clef
- Medium de stockage **reprogrammable** :
  - Lecture seule en utilisation normale
  - Lecture-écriture par choix de l'utilisateur

Liberté avec vérifications de signatures

Rendre la racine de confiance **à l'utilisateur** :

- Installation des clefs en mémoire OTP :
  - N emplacements disponibles :
    fabricant, *distribution*, *personnel*
  - Bit de révocation par clef
- Medium de stockage **reprogrammable** :
  - Lecture seule en utilisation normale
  - Lecture-écriture par choix de l'utilisateur
- **Intermédiaire matériel** de confiance :
  - Déplacement de la racine de confiance
  - Vérification ou initialisation TPM
  - Stockage des clefs

Garanties nécessaires

Installation des clefs en **mémoire OTP** :

- Gestion des $N - n$ emplacements libres (OTP) ?
- Action à réaliser physiquement

Garanties nécessaires

Installation des clefs en **mémoire OTP** :

- Gestion des $N - n$ emplacements libres (OTP) ?
- Action à réaliser physiquement

Medium de stockage **reprogrammable** :

- Suppression des données utilisateur
- Suppression des données fabricant (DRM)

Garanties nécessaires

Installation des clefs en **mémoire OTP** :

- Gestion des $N - n$ emplacements libres (OTP) ?
- Action à réaliser physiquement

Medium de stockage **reprogrammable** :

- Suppression des données utilisateur
- Suppression des données fabricant (DRM)

Intermédiaire matériel de confiance :

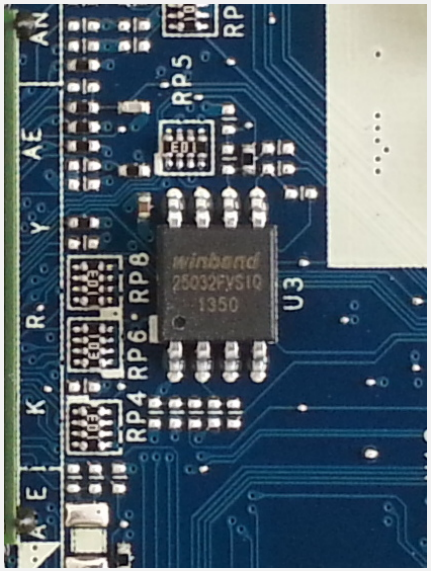
- Intégrité physique (suppression des clefs)

Études de cas

Études de cas

Générique

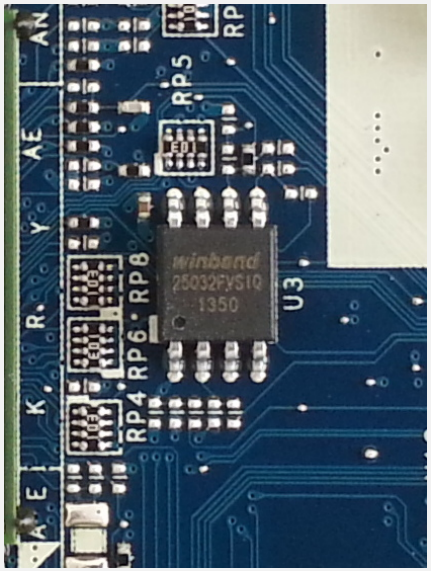
coreboot, GRUB et PGP



Composants :

- Logiciel libre de démarrage : **coreboot**
- *Payload* avec vérification PGP : **GRUB**
- Stockage en **lecture seule** $\overline{WP} = 0\ V$
- Sécurisation de **l'accès physique**

coreboot, GRUB et PGP



Composants :

- Logiciel libre de démarrage : **coreboot**
- *Payload* avec vérification PGP : **GRUB**
- Stockage en **lecture seule** $\overline{WP} = 0$ V
- Sécurisation de **l'accès physique**

Design matériel associé :

- Démarrage sur flash SPI
- Pull-down externe sur $\overline{WP}$

Études de cas

Appareils CrOS

Modèle de sécurité CrOS

Chromebooks, Chromeboxes, Chromebases, Chromebits

Modèle de sécurité CrOS

Chromebooks, Chromeboxes, Chromebases, Chromebits

Plateformes :

- **x86** : Intel Sandybridge-Kaby Lake
- **ARM signé** : Samsung Exynos
- **ARM non-signé** : Rockchip RK3288, nVidia Tegra K1

Modèle de sécurité CrOS

Chromebooks, Chromeboxes, Chromebases, Chromebits

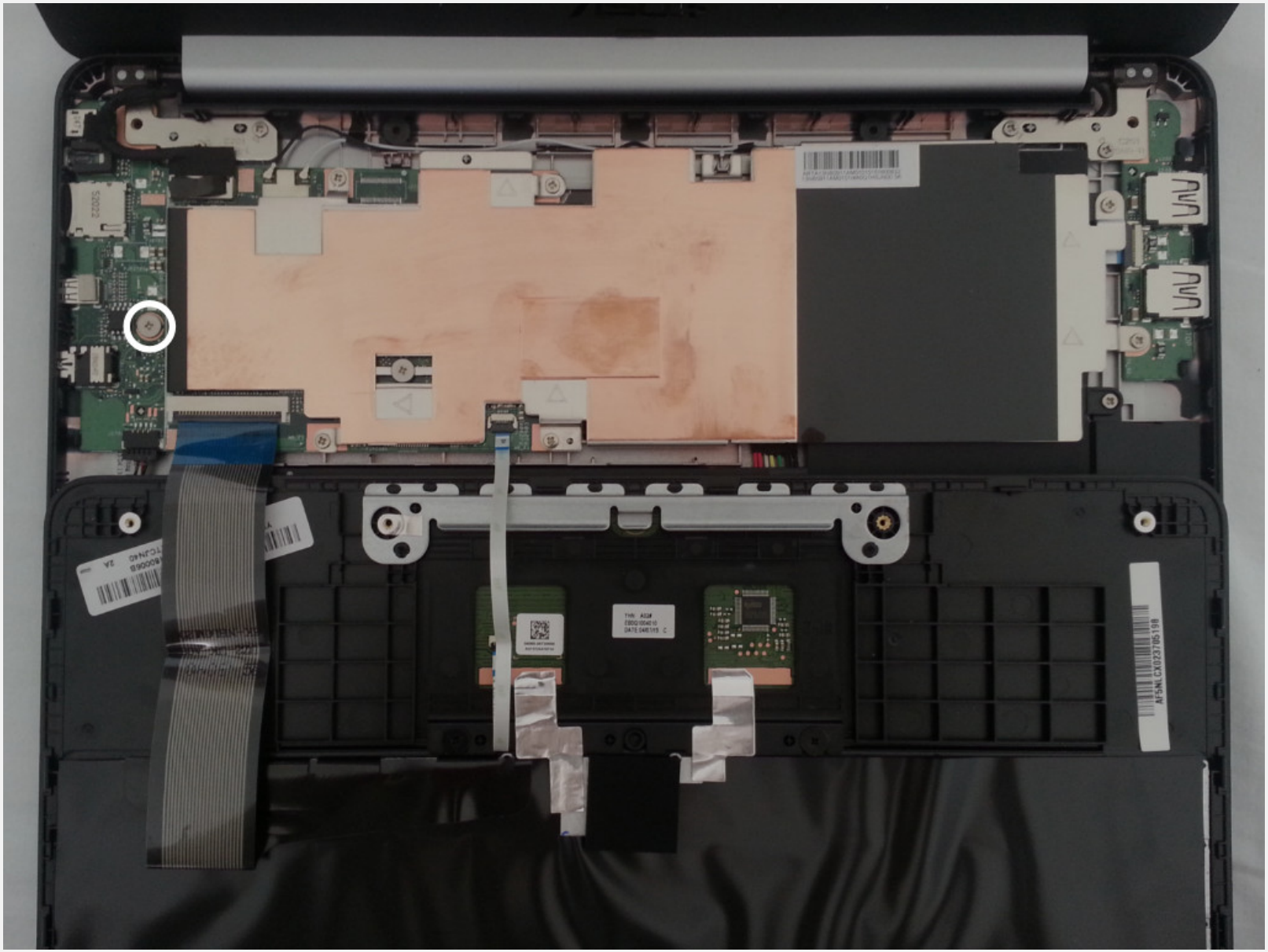
Plateformes :

- **x86** : Intel Sandybridge-Kaby Lake
- **ARM signé** : Samsung Exynos
- **ARM non-signé** : Rockchip RK3288, nVidia Tegra K1

Modèle de sécurité :

- Flash SPI en lecture seule avec une vis
- Clefs de vérification sur la flash SPI
- Ne couvre pas l'accès physique

La vis $\overline{WP}$



Vérification du démarrage

Composants logiciels :

- Démarrage : **coreboot**
- *Payload* : **Depthcharge**
- Vérification : **Vboot**
- Logiciel EC : **Chrome EC**

Vérification du démarrage

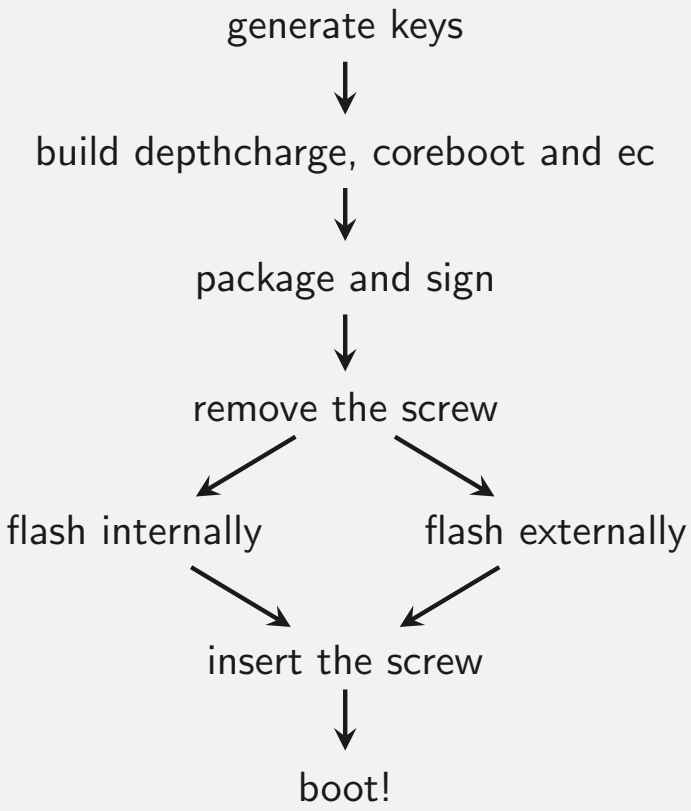
Composants logiciels :

- Démarrage : **coreboot**
- *Payload* : **Depthcharge**
- Vérification : **Vboot**
- Logiciel EC : **Chrome EC**

Modes de démarrage :

- Mode **normal**
- Mode **récupération**
- Mode **développeur**

Remplacement du logiciel et clefs :



Approche et contraintes

Approche **communautaire** :

- **Histoire** de l'équipe *CrOS firmware*
- Développeurs **sympathiques et utiles**
- **Contributions et patchs** bienvenus
- **Code source** : `https://chromium.googlesource.com`

Approche et contraintes

Approche **communautaire** :

- **Histoire** de l'équipe *CrOS firmware*
- Développeurs **sympathiques et utiles**
- **Contributions et patchs** bienvenus
- **Code source** : `https://chromium.googlesource.com`

Design matériel associé :

- Démarrage sur flash SPI
- Vis et pull-up/down sur $\overline{WP}$

Merci !

Merci !

Questions ?