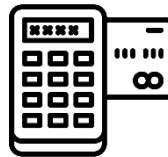


Etude et implémentation de mécanismes de protection d'exécution de multiples applications embarquées

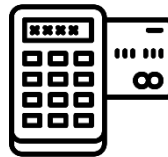
Abderrahmane Sensaoui

Systemes Légers



- **Très connectés :**
 - Grande surface d'attaque.
- **Traient des données sensibles :**
 - Clés cryptographique.
 - Données privées.
- **Embarqués dans des systèmes critiques :**
 - Dégâts sur un environnement.

Systemes Légers



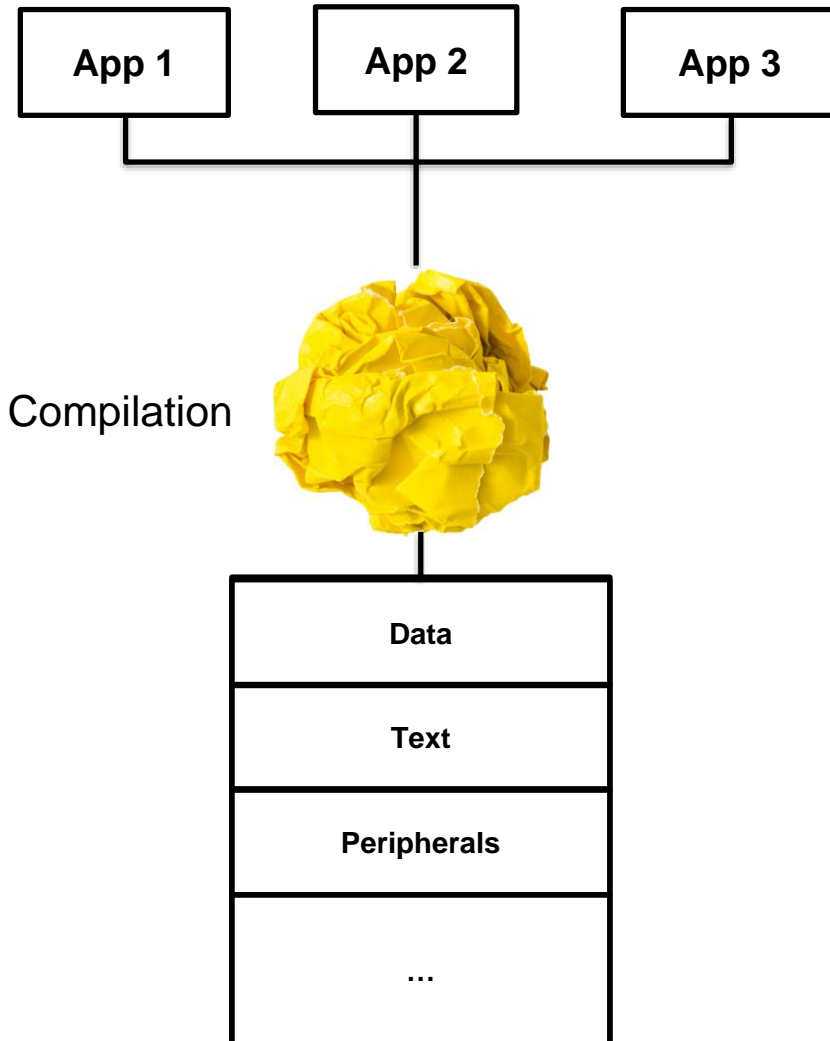
- **Contraintes :**

- Faibles ressources mémoire (ex. environ 1MB de flash et 192KB de RAM).
- Temps critique et déterministe.
- Mémoire physique.

- **Multiples applications :**

- Différentes applications sont exécutées sur le même système.

Défis de Sécurité



- **Fuite des données**
Le modèle de mémoire plat et le manque de MMU facilite aux attaquants les accès aux données sensibles.
- **Aucune Séparation**
Toutes les applications partagent le même plan de mémoire.
- **Vérification logicielle compliquée**
La surface d'attaque devient rapidement grande et complique la vérification.
- **Confiance**
Comment établir la confiance en une donnée, application, ou système?

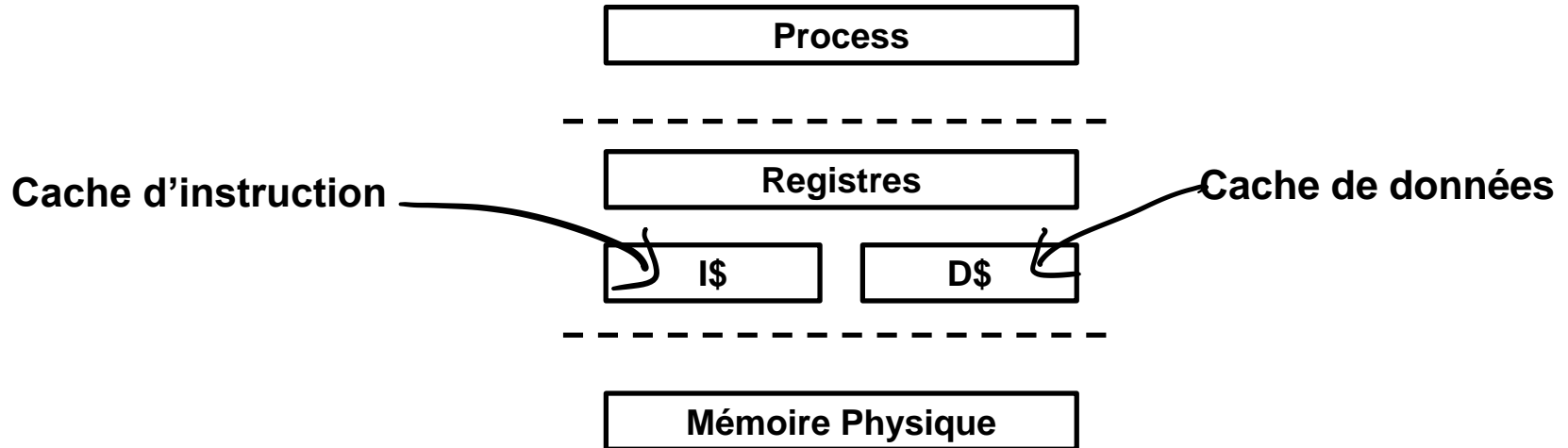
Objectifs

- **Etude de l'existant**
Mener une étude profonde pour évaluer l'existant et identifier les limitations.
- **Une solution matérielle/logicielle**
Proposer une solution co-design matérielle/logicielle.
- **Rentabilité**
Trouver un compromis entre le coût, les performances et le niveau de protection garanti.

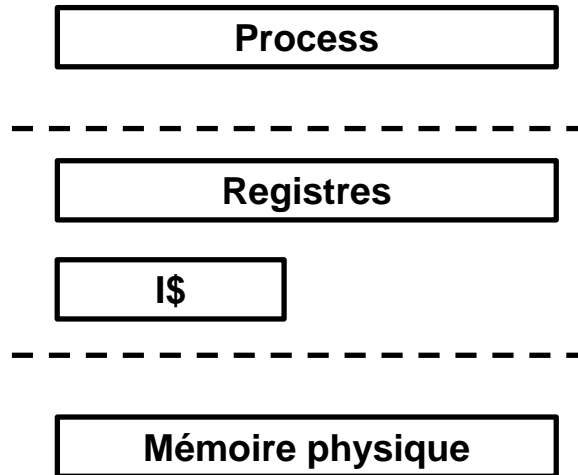
Agenda

- I. Prérequis**
- II. Evaluation des solutions existantes
- III. Toubkal : Architecture hybride d'isolation et d'attestation
- IV. Perspectives et Conclusion

La Hiérarchie de la Mémoire

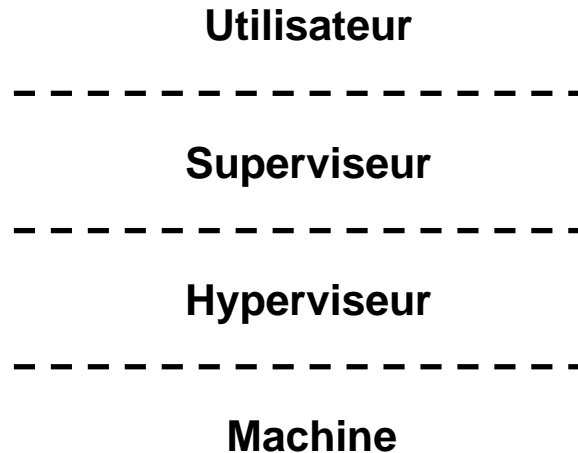


La Hiérarchie de la Mémoire



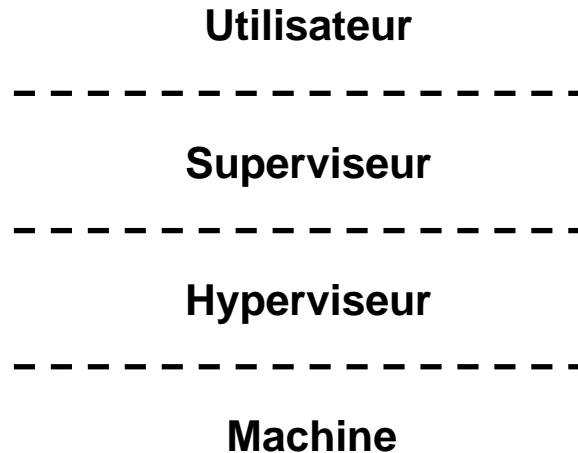
- Accès directs aux mémoires physiques

Modes d'exécution



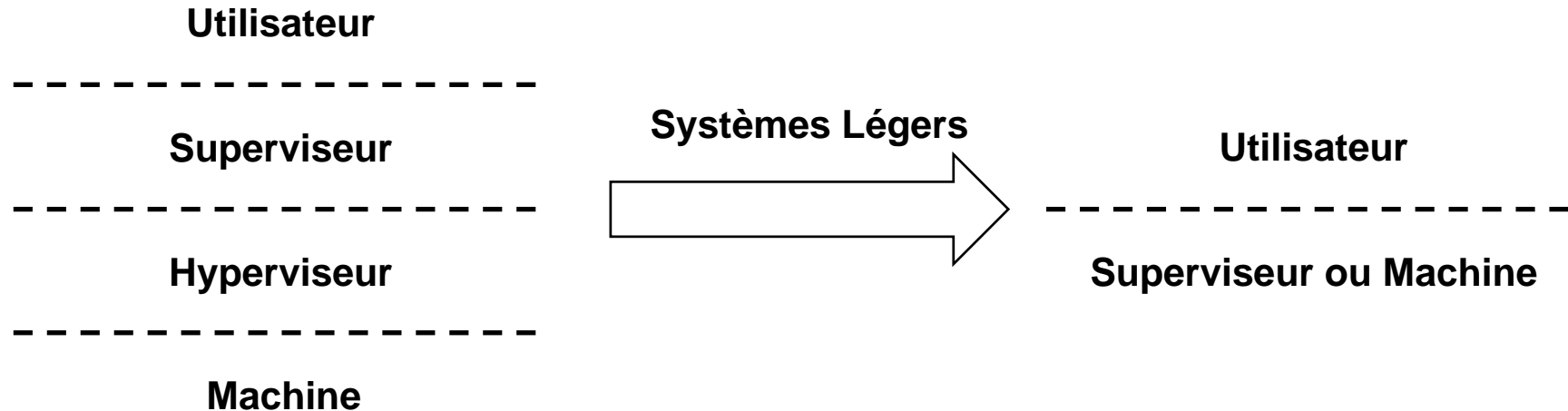
- Quatre modes d'exécution

Modes d'exécution



- Quatre modes d'exécution

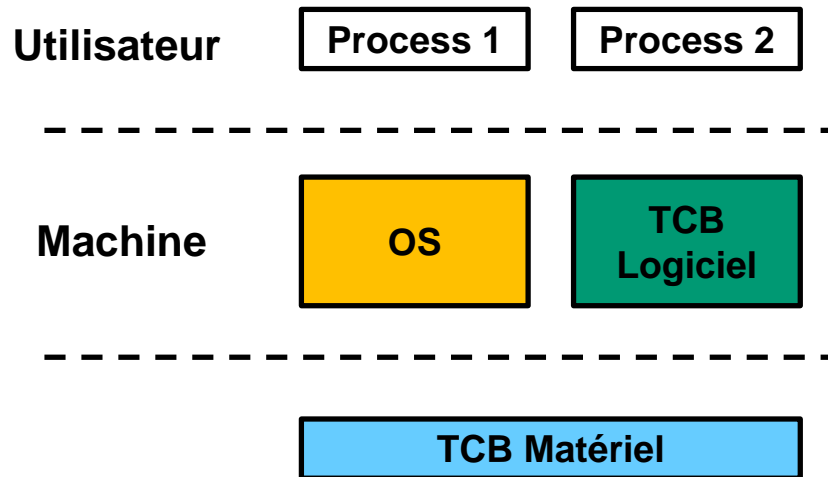
Modes d'exécution



- Quatre modes d'exécution
- Seulement deux modes pour les systèmes légers

Trusted Computing Base

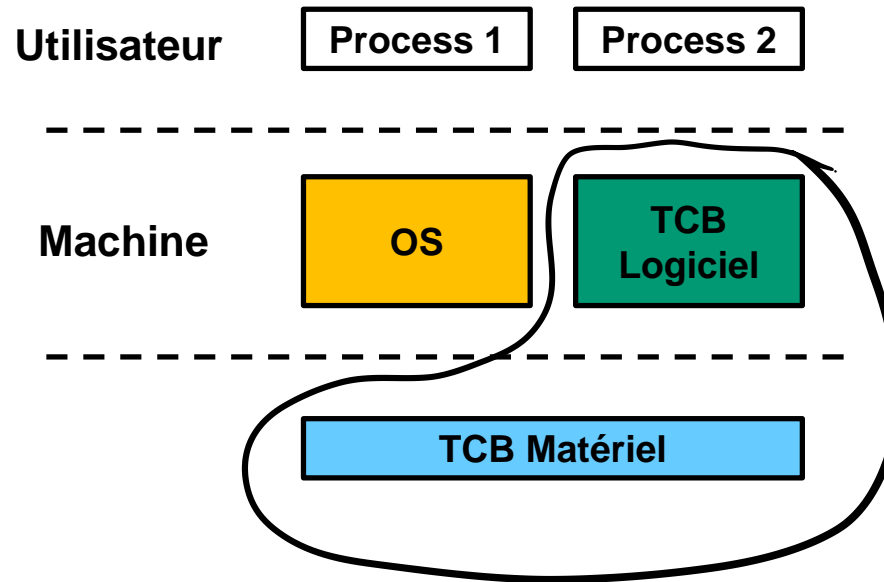
“An entity can be trusted if it always behaves in the expected manner for the intended purpose” – **TCG 2004**



Trusted Computing Base

- **Trusted Computing Base (TCB)**

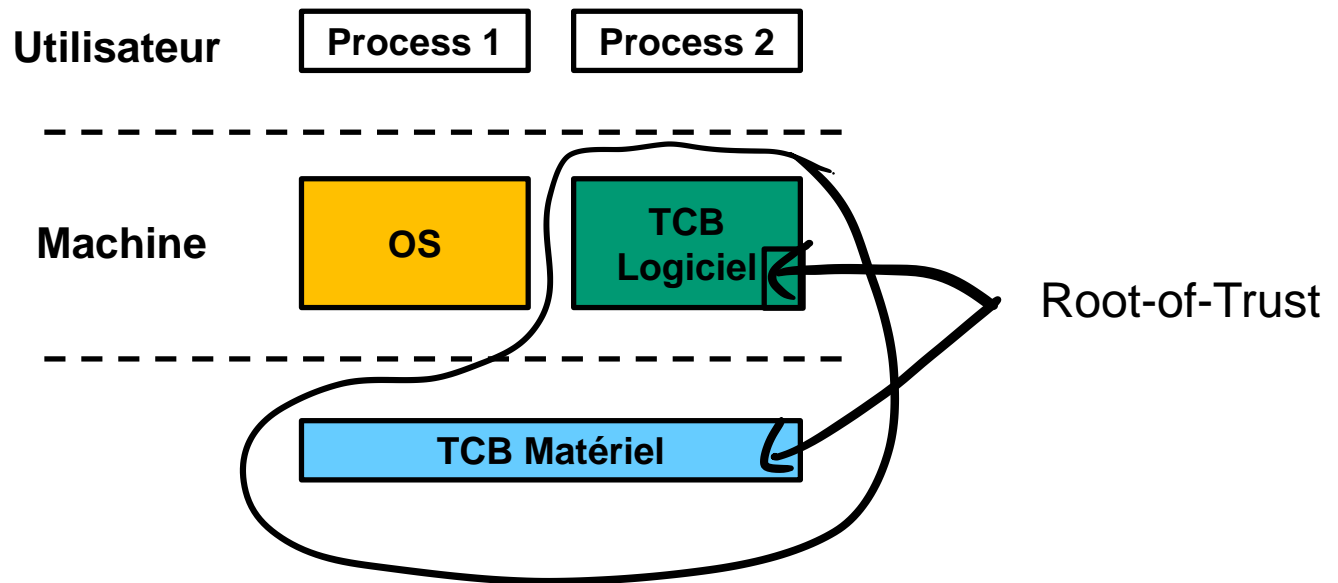
Un ensemble de composants matériels et logiciels responsables du renforcement de la sécurité du système.



Trusted Computing Base

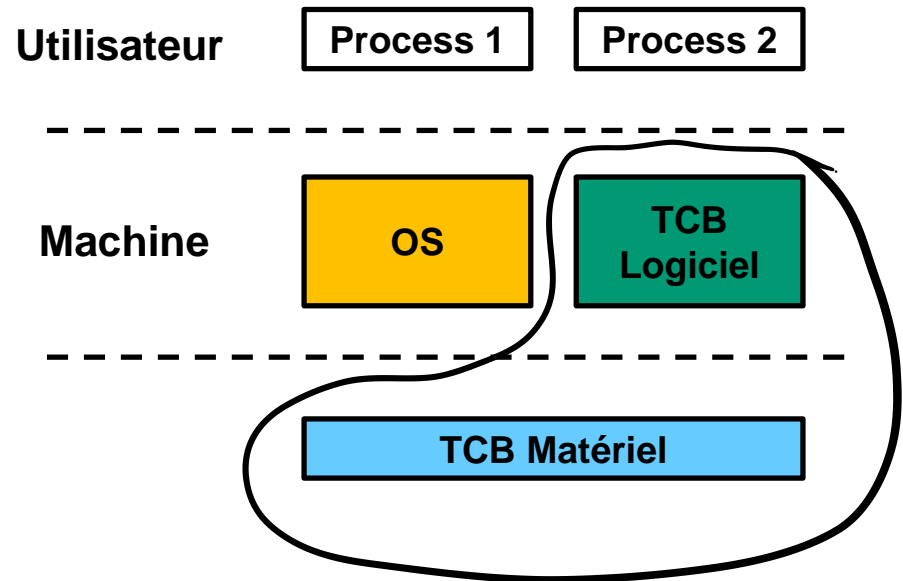
- **Trusted Computing Base (TCB)**

Un ensemble de composants matériels et logiciels responsables du renforcement de la sécurité du système.



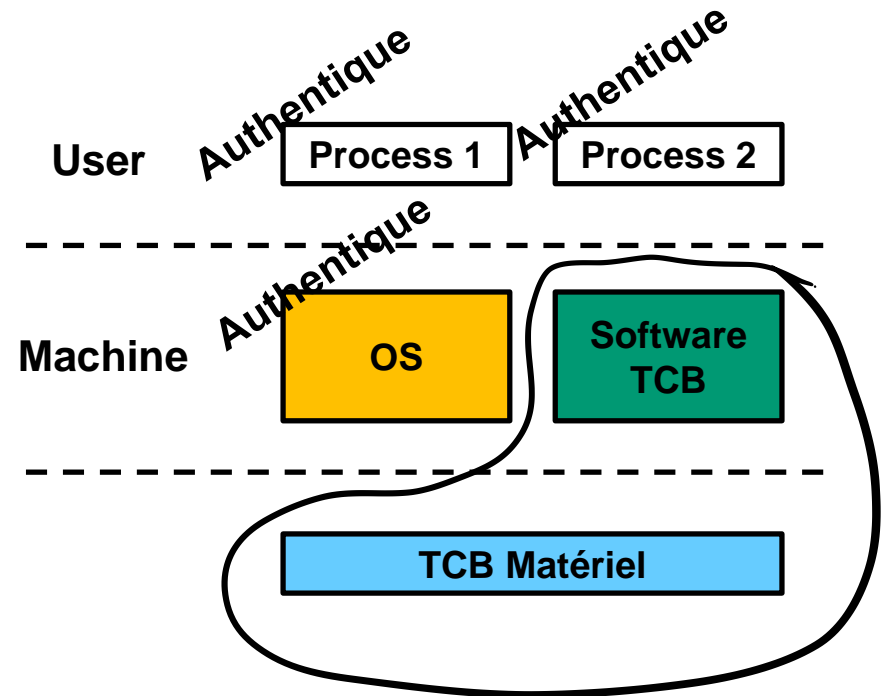
Trusted Computing Base

- Trusted Computing Base
- Propose des mécanismes de protection



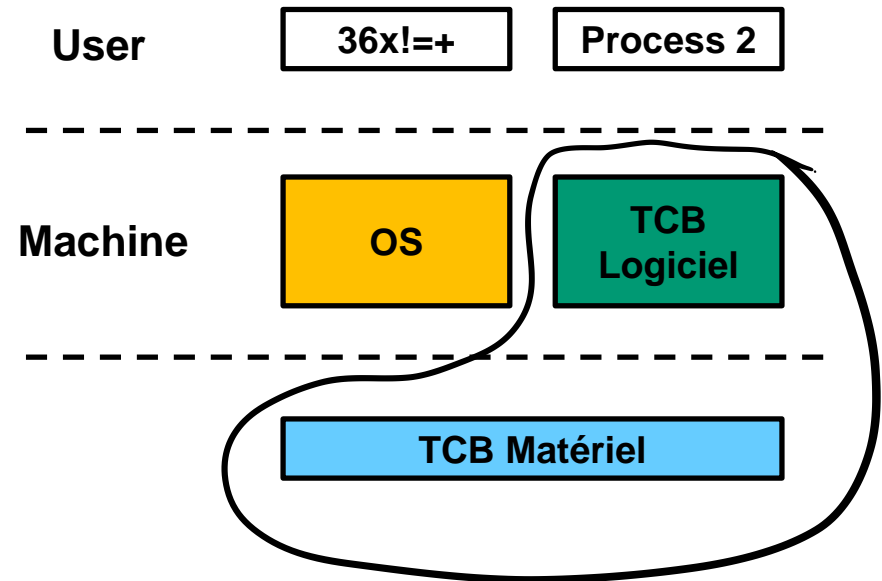
Trusted Computing Base

- Trusted Computing Base
- Propose des mécanismes de protection
 - Attestation



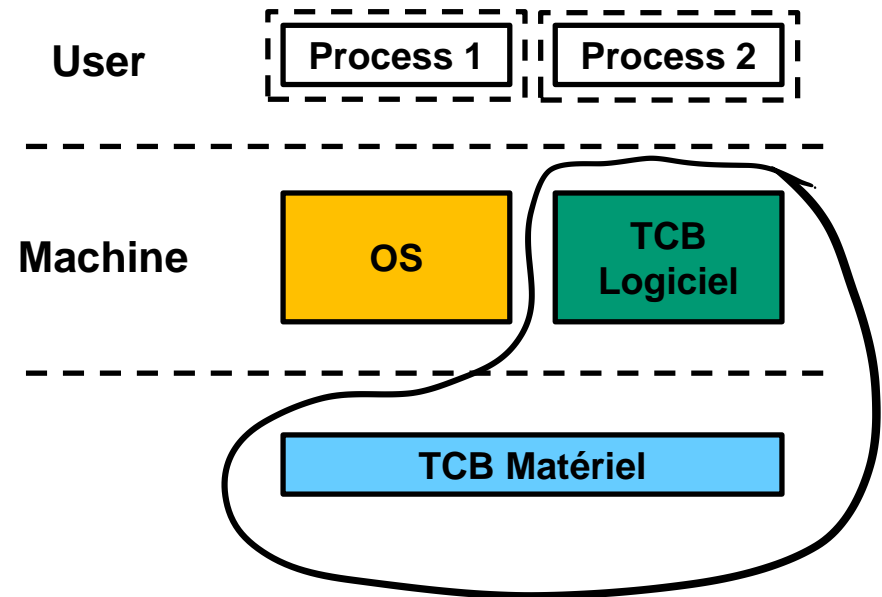
Trusted Computing Base

- **Trusted Computing Base**
- **Propose des mécanismes de protection**
 - Attestation
 - Confidentialité du Code

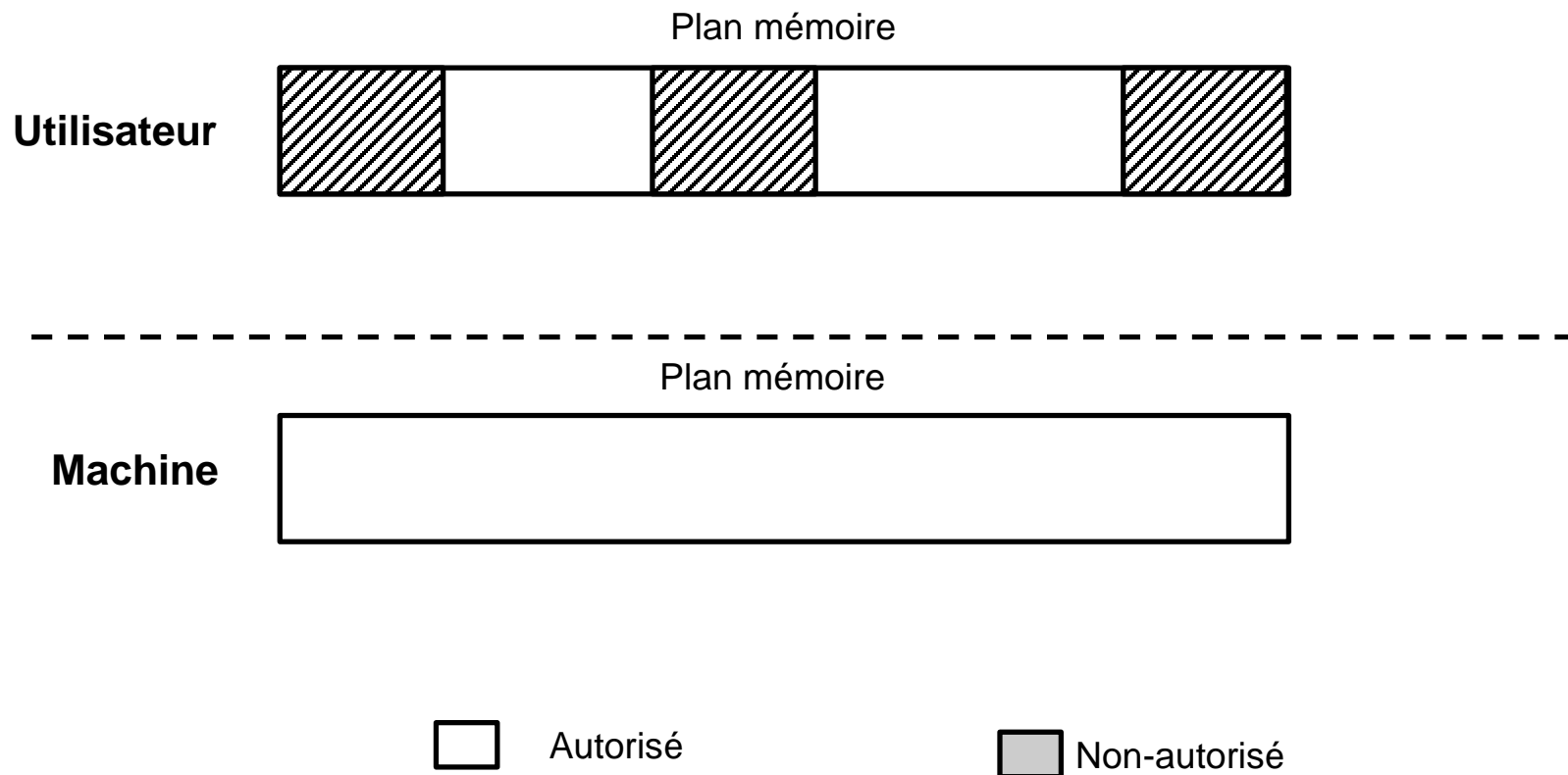


Trusted Computing Base

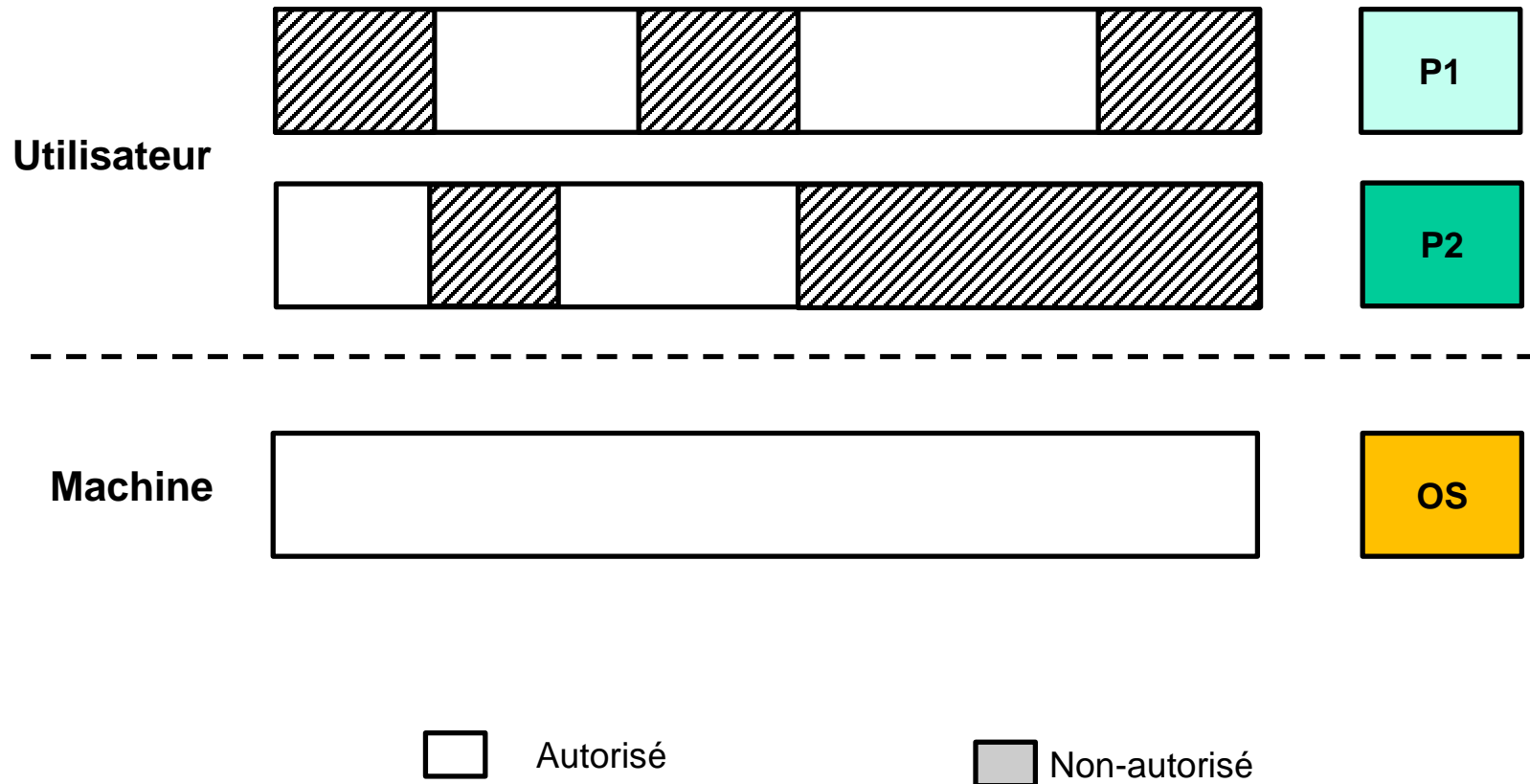
- **Trusted Computing Base**
- **Propose des mécanismes de protection**
 - Attestation
 - Confidentialité du Code
 - Isolation



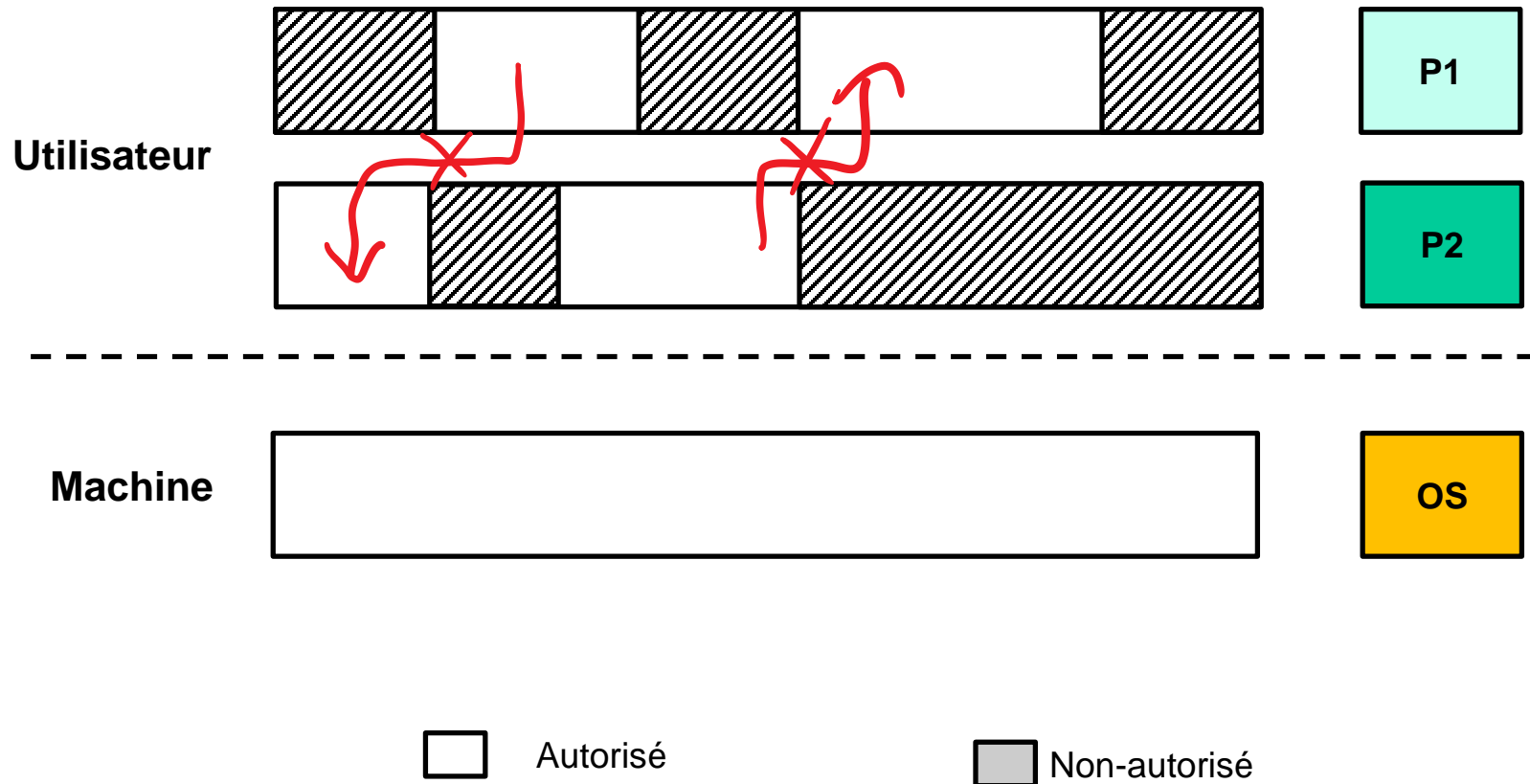
Isolation



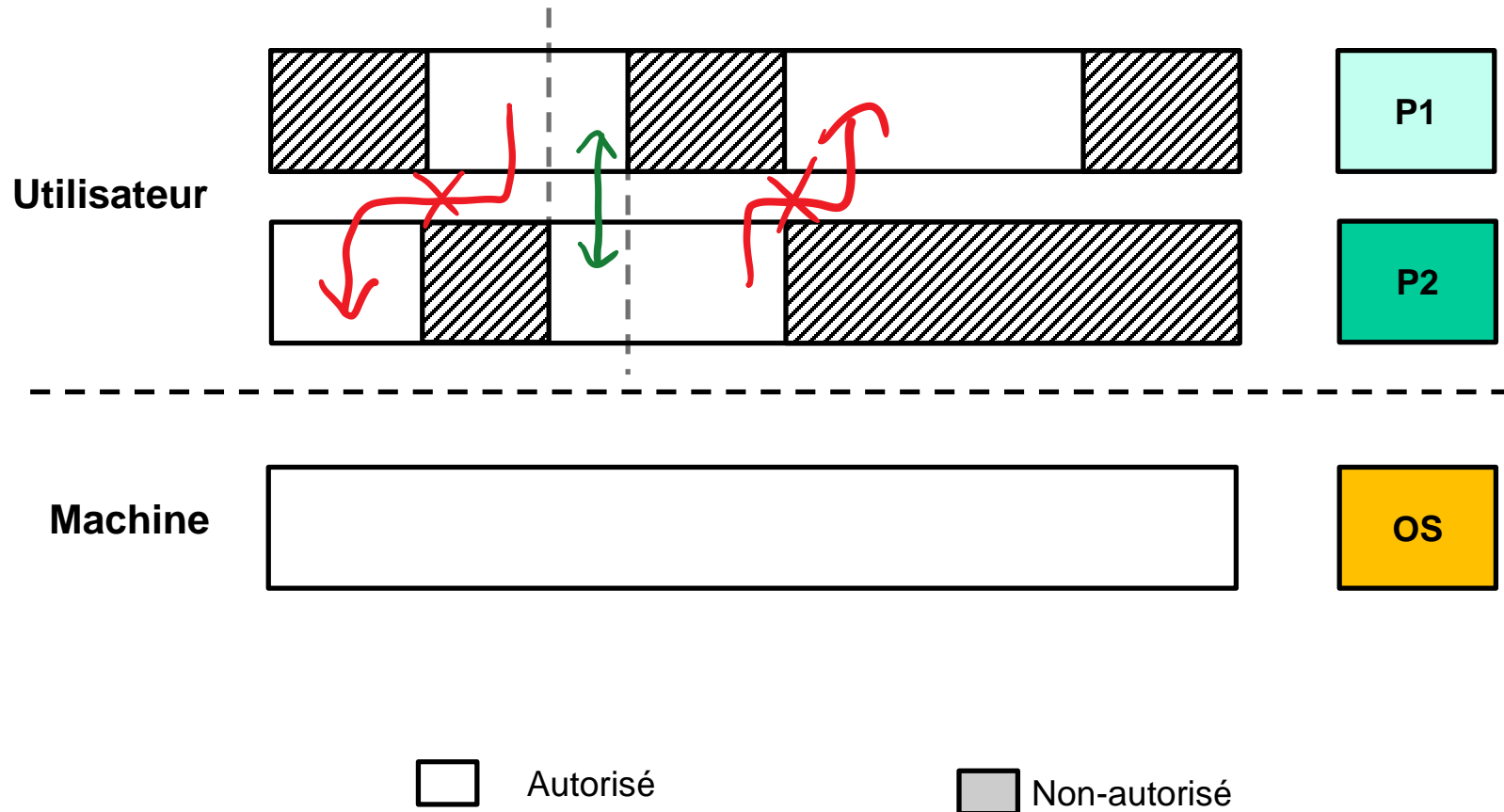
Isolation



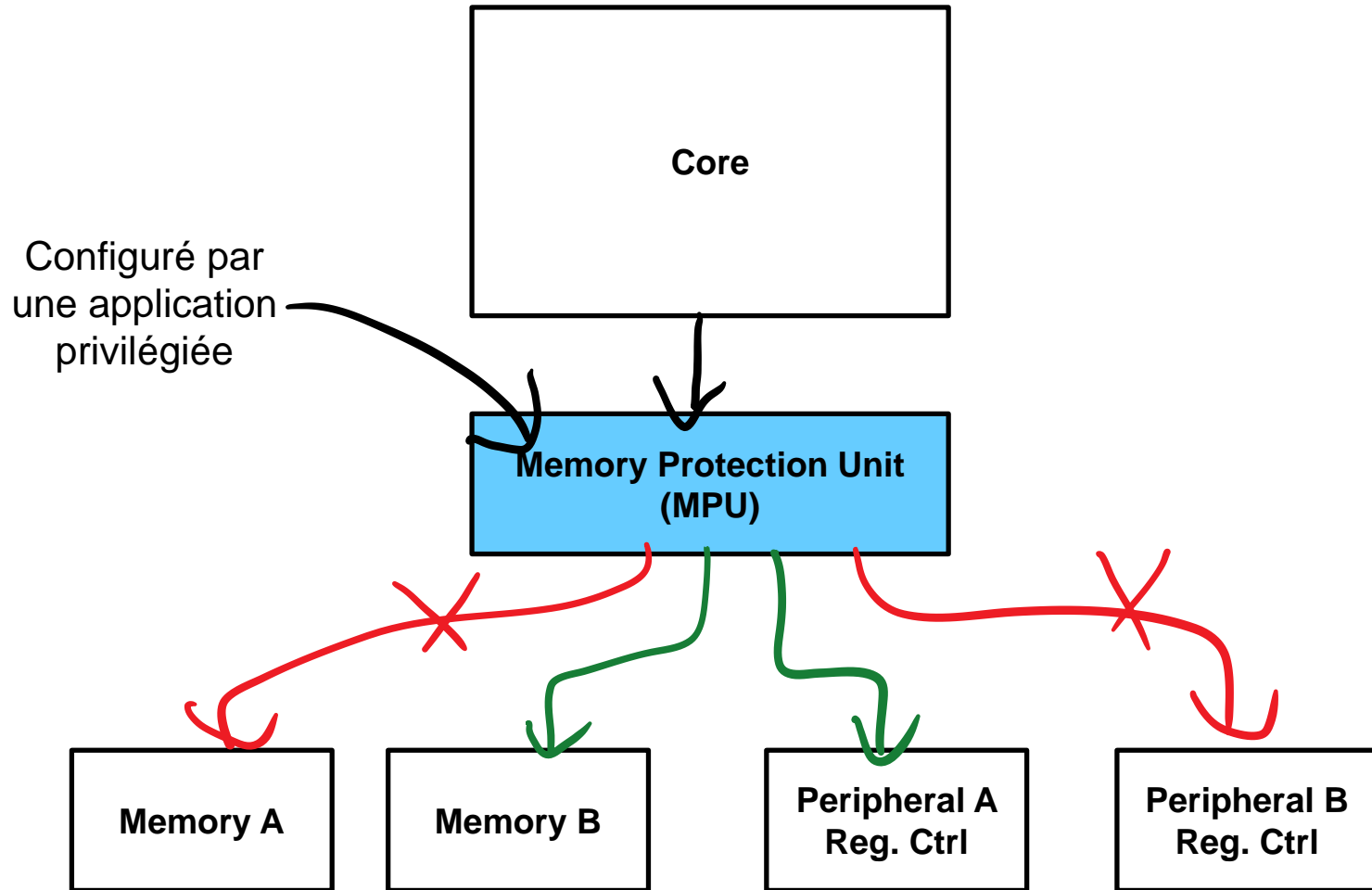
Isolation



Isolation

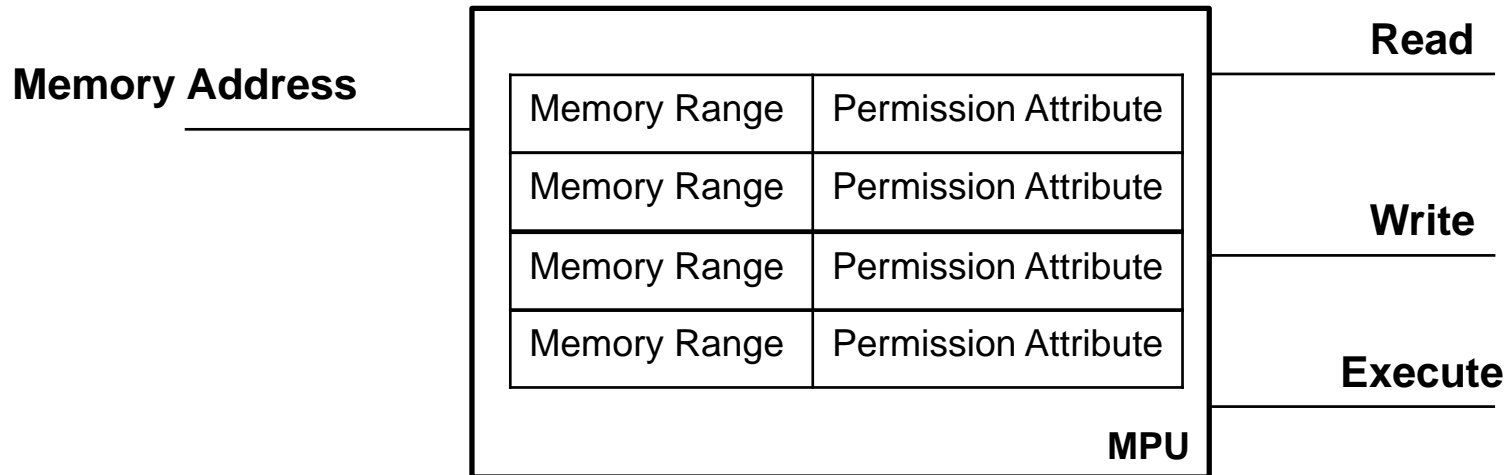


Isolation



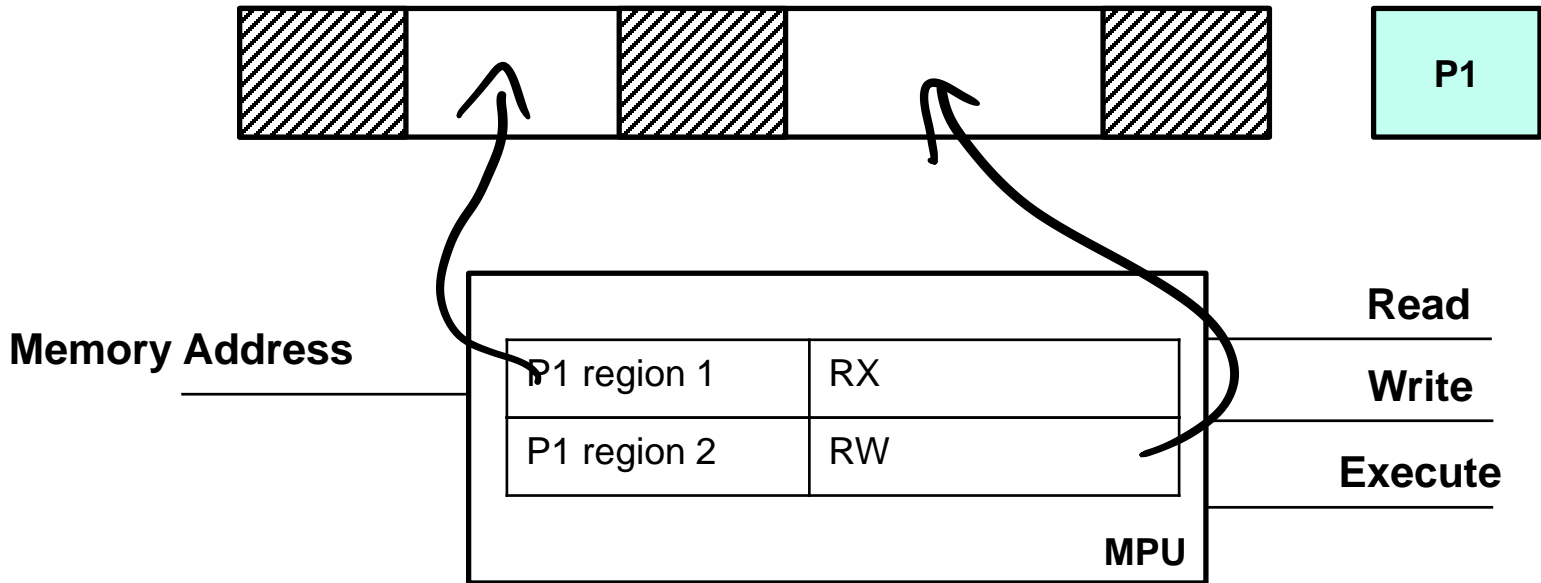
Isolation

Le modèle de la MPU utilisé dans nos systèmes



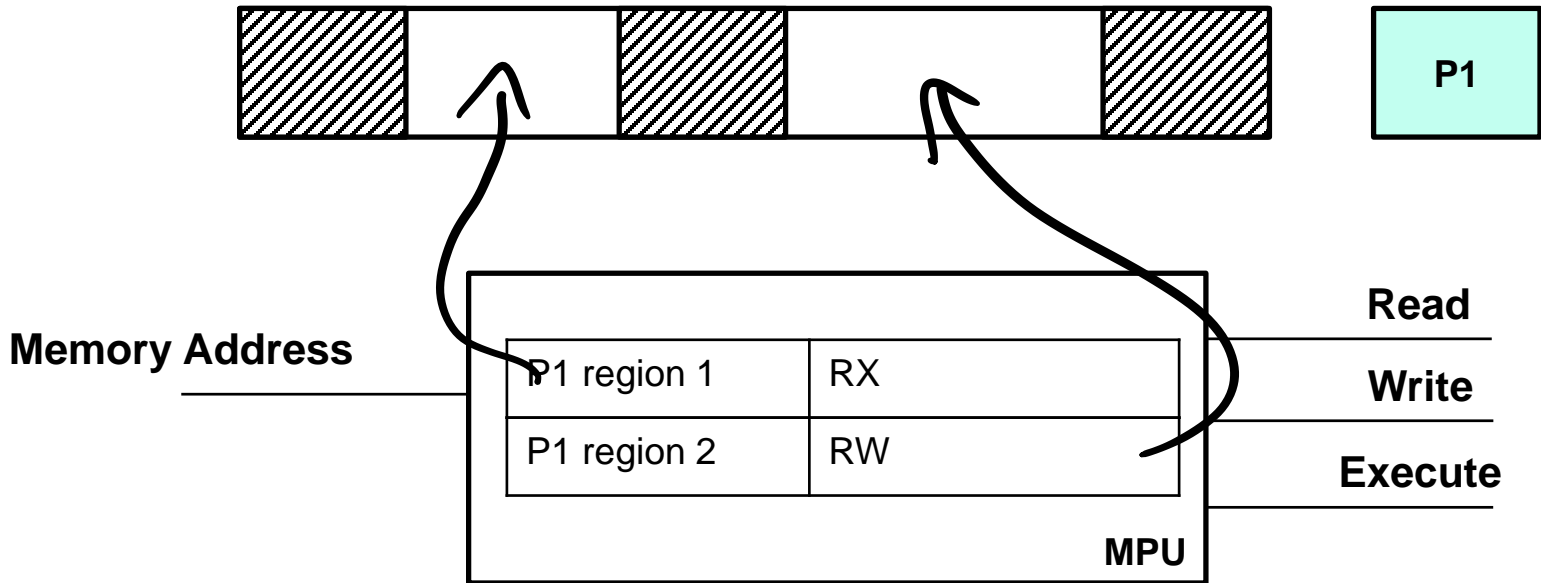
Exemple: Armv7 Memory Protection Unit (MPU), Armv8 MPU, Rocket Chip Physical Memory Protection (PMP)

Isolation



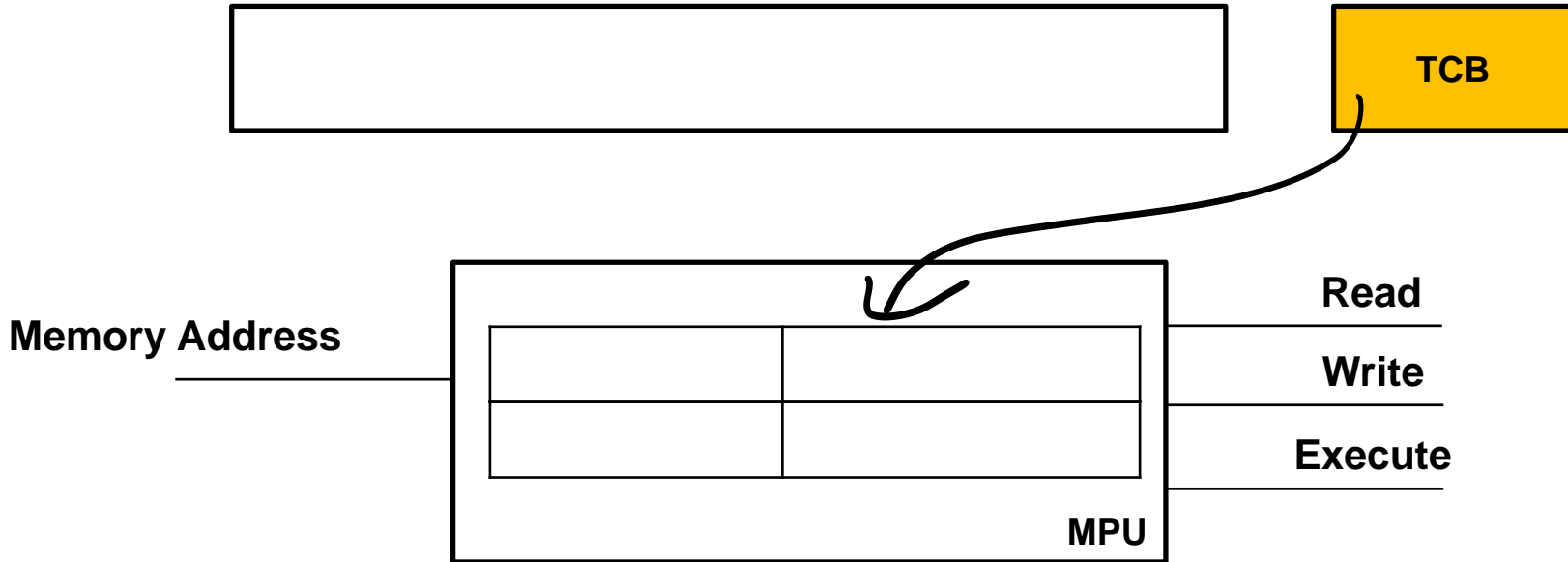
- P1 exécute son code.

Isolation



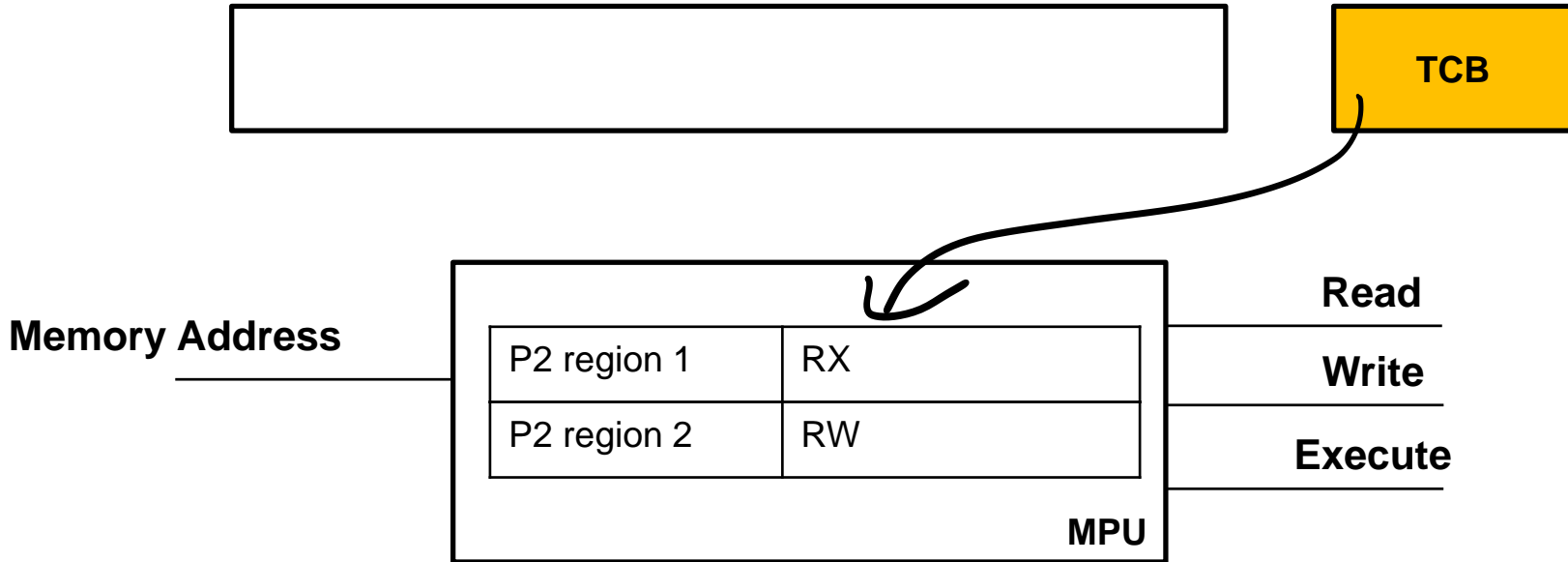
- P1 exécute son code.
- Un appel pour une commutation de contexte se déclenche.

Isolation



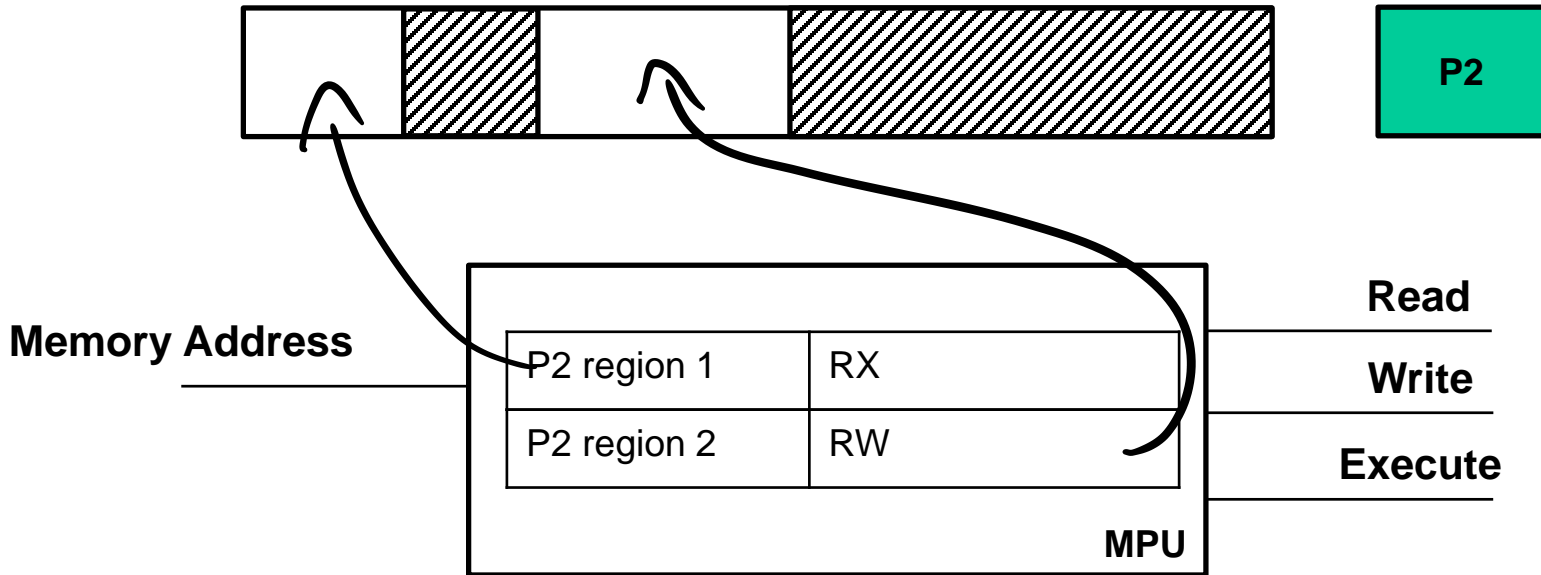
- P1 exécute son code.
- Un appel pour une commutation de contexte se déclenche.
- Reconfiguration de la MPU.

Isolation



- P1 exécute son code.
- Un appel pour une commutation de contexte se déclenche.
- Reconfiguration de la MPU en mode machine.

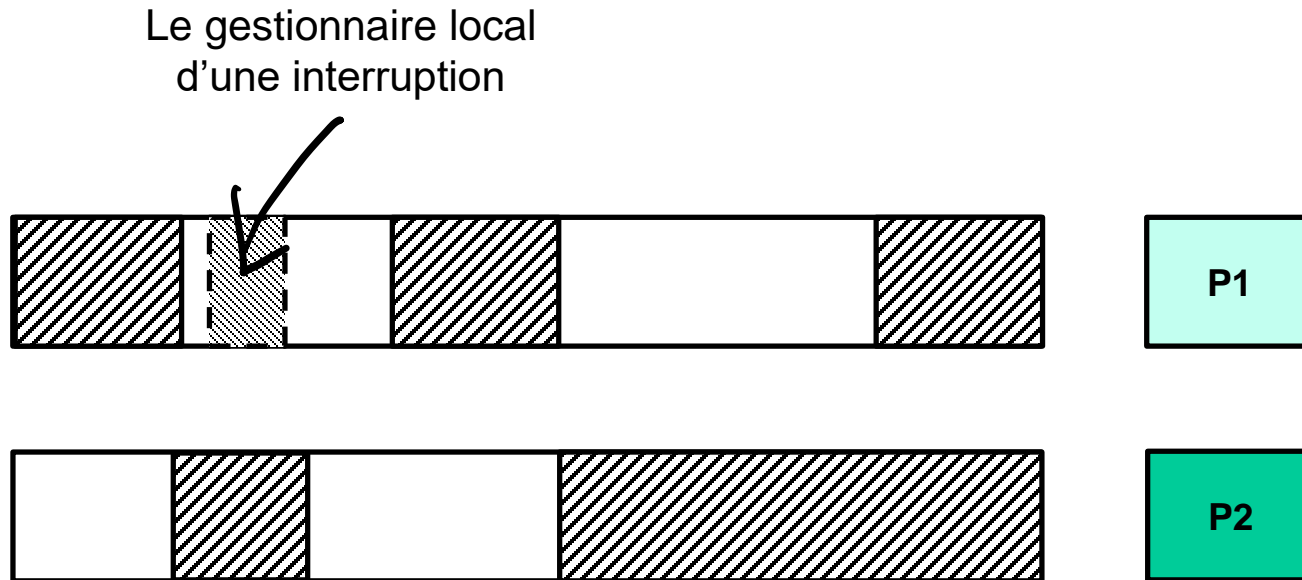
Isolation



- P1 exécute son code.
- Un appel pour une commutation de contexte se déclenche.
- Reconfiguration de la MPU en mode machine.
- P2 Commence l'exécution de son code.

Isolation

- **La reconfiguration de la MPU**
 - Commutation d'un processus à un autre.
 - Déclenchement d'une interruption.



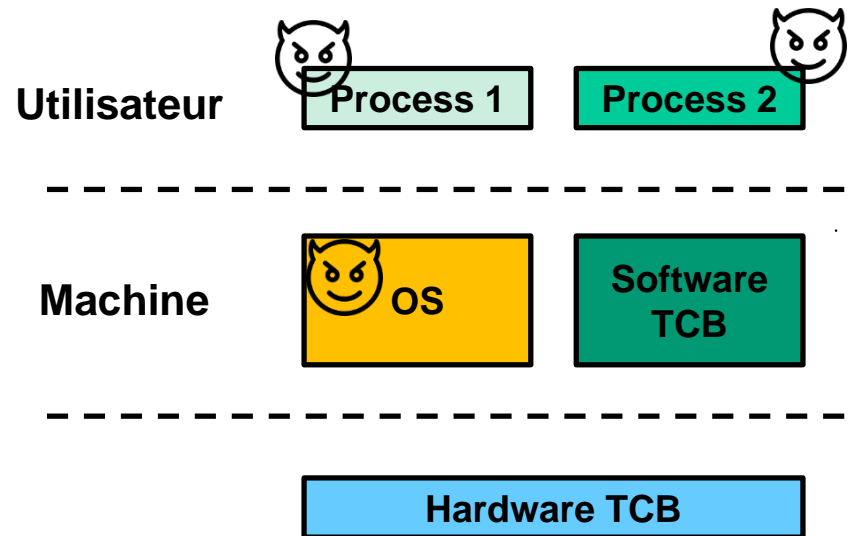
P2 en cours d'exécution

Isolation

- **La reconfiguration de la MPU**
 - Commutation d'un processus à un autre.
 - Déclenchement d'une interruption.
 - Appel d'un point d'entrée vers une autre application.
- **Communication inter-process**
 - Mémoires partagées.
 - Passerelles sécurisées.

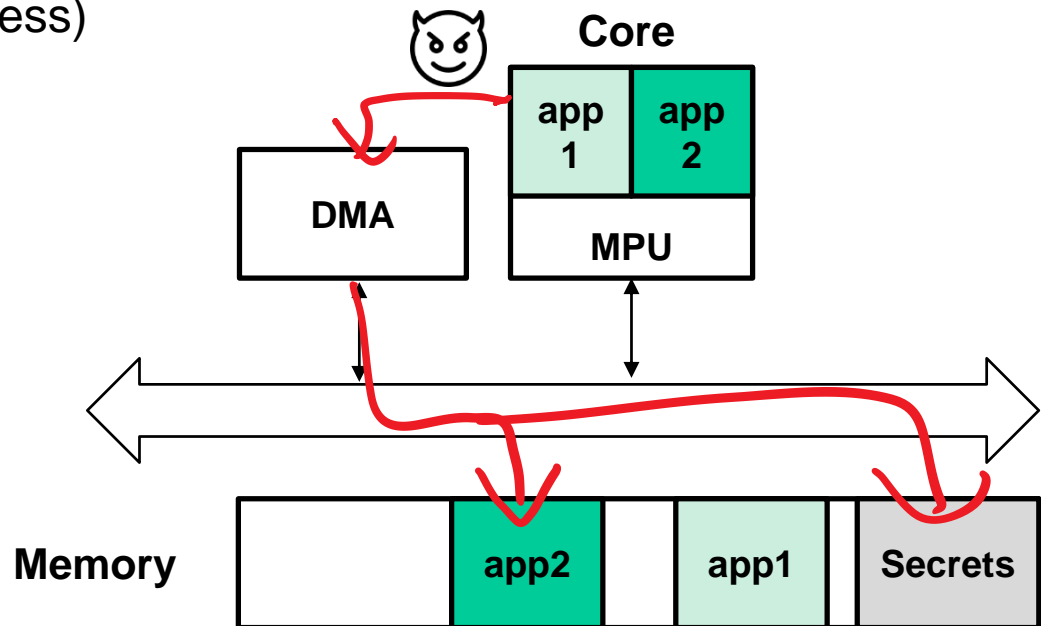
Modèle d'Attaque

- L'attaquant peut contrôler tous les composants logiciels en dehors du TCB.



Modèle d'Attaque

- L'attaquant peut contrôler tous les composants logiciels en dehors du TCB.
- L'attaquant peut contrôler un périphérique comme le DMA (Direct Memory Access)



Agenda

- I. Prérequis
- II. Evaluation des solutions existantes**
- III. Toubkal : Architecture hybride d'isolation et d'attestation
- IV. Perspectives et Conclusion

Architectures

- Etude de 15 architectures d'isolation et d'attestation.

	Isolation	Inter-Process Communication	Root of Trust	Dynamic loading	Exception Handling	Application reboot	Attestation	Code confidentiality	IO Peripherals protection	Lightweight	Memory Protection Module	Hardware-Only TCB	Software adaptation	Open Source	Academic Deployed	ISA
Mondrix	x	x	o	o	x	o	o	o	o	o	Mondrian	o	x	o	x	Multiple
SMART	o	o	x	o	o	o	x	o	o	x	-	o	x	o	x	AVR/MSP430
Sancus	x	x	x	x	o	?	x	o	o	x	-	x	x	x	o	MSP430
SGX	x	x	x	x	x	x	x	x	xo	o	MMU	o	x	o	o	x86_64
TyTan	x	x	x	x	x	o	x	o	o	x	EA-MPU	o	x	o	x	Siskiyou Peak
TrustLite	x	x	o	o	x	o	x	o	o	x	EA-MPU	o	x	o	x	Siskiyou Peak
uVisor	x	x	o	o	x	o	o	o	o	x	MPU	o	x	x	o	Arm
TockOS	x	o	o	x	x	xo	o	o	xo	x	MPU	o	x	x	x	Arm
Sanctum	x	x	x	x	x	x	x	o	xo	o	MMU	o	x	x	x	RISC-V
TrustZone-M	x	o	x	x	o	o	x	o	x	x	-	o	x	xo	o	Arm
Sopris	x	o	x	o	o	?	x	o	o	x	MMU	o	x	o	o	Arm
EPOXY	x	o	o	o	o	o	o	o	o	x	MPU	o	o	x	x	Arm
ACES	x	o	o	o	o	o	o	o	o	x	MPU	o	o	x	x	Arm
MultiZone	x	x	x	o	?	x	xo	o	x	x	PMP	o	o	o	o	RISC-V
VRASED	o	o	x	o	o	o	x	o	xo	x	-	o	x	o	x	AVR/MSP430

x: Yes, o: No, xo: Partial, ?: NA, -: Non-relevant

Architectures

- Etude de 15 architectures d'isolation et d'attestation.
- Etude plus étendue des architectures avec une MPU.

	Isolation	Inter-Process Communication	Root of Trust	Dynamic loading	Exception Handling	Application reboot	Attestation	Code confidentiality	IO Peripherals protection	Lightweight	Memory Protection Module	Hardware-Only TCB	Software adaptation	Open Source	Academic Deployed	ISA
Mondrix	x	x	o	o	x	o	o	o	o	o	Mondrian	o	x	o	x	Multiple
SMART	o	o	x	o	o	o	x	o	o	x	-	o	x	o	x	AVR/MSP430
Sancus	x	x	x	x	o	?	x	o	o	x	-	x	x	x	x	MSP430
SGX	x	x	x	x	x	x	x	x	xo	o	MMU	o	x	o	o	x86_64
TyTan	x	x	x	x	x	o	x	o	o	x	EA-MPU	o	x	o	x	Siskiyou Peak
TrustZone	x	x	o	o	x	o	x	o	o	x	EA-MPU	o	x	o	x	Siskiyou Peak
uVisor	x	x	o	o	x	o	o	o	o	x	MPU	o	x	x	o	Arm
TockOS	x	o	o	x	x	xo	o	o	xo	x	MPU	o	x	x	x	Arm
Sanctum	x	x	x	x	x	x	x	o	xo	o	MMU	o	x	x	x	RISC-V
TrustZone-M	x	o	x	x	o	o	x	o	x	x	-	o	x	xo	o	Arm
Sapris	x	o	x	o	o	?	x	o	o	x	MMU	o	x	o	o	Arm
EPOXY	x	o	o	o	o	o	o	o	o	x	MPU	o	o	x	x	Arm
ACES	x	o	o	o	o	o	o	o	o	x	MPU	o	o	x	x	Arm
MultiZone	x	x	x	o	?	x	xo	o	x	x	PMP	o	o	o	o	RISC-V
VRASED	o	o	x	o	o	o	x	o	xo	x	-	o	x	o	x	AVR/MSP430

x: Yes, o: No, xo: Partial, ?: NA, -: Non-relevant

Critères de Comparaison

- **Equipements de protection**
 - Communication entre processus
 - Protection DMA
 - Existence d'un "Root-of-Trust"
 - Sûreté de la mémoire

- **Evaluation du temps d'exécution**
 - Création d'un Processus
 - Configuration de la MPU
 - Changement de contexte
 - Gestion des interruptions

Etude Expérimentale

- **Cible:**
STM32F429I Discovery
Cortex-M4, Armv7 MPU
48MHz, 2MB FLASH, 192KB RAM

- **Etapes de l'évaluation**
 - Portage des différentes architectures.
 - Implémentation des différents tests.
 - Exécution des différents tests.

Equipements de protection

	Gestion d'interruption	Root-of- Trust	Protection DMA	Sûreté mémoire	Communication inter-process
uVisor	✓	✗	✗	✗	✓
TockOS	✓	✗	✓ ✗	✓ ✗	✓
ACES	✓ ✗	✗	✗	✗	✗

- Aucune des solutions propose un Root-of-Trust.
- Protection DMA négligée.
- TockOS utilise le langage Rust mais juste pour certaines parties du TCB.

Equipements de protection

	Gestion d'interruption	Root-of-Trust	Protection DMA	Sûreté mémoire	Communication inter-process
uVisor	✓	✗	✗	✗	✓
TockOS	✓	✗	✓ ✗	✓ ✗	✓
ACES	✓ ✗	✗	✗	✗	✗

- Aucune des solutions propose un Root-of-Trust.
- Protection DMA négligée.
- TockOS utilise le langage Rust mais juste pour certaines parties du TCB.

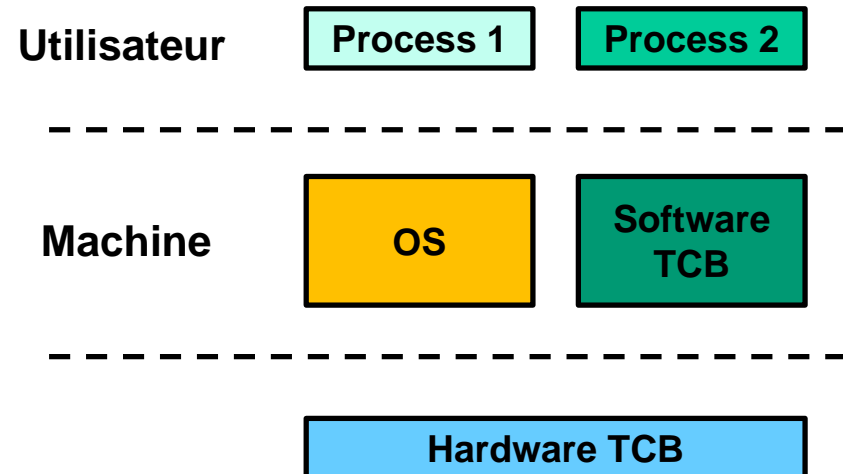
Performance

	Creation d'un process	Changement de contexte	Accès à un register de ctrl	Gestion d'interruption
uVisor	5,500	2,040 – 6,100	6	290 – 4,740
TockOS	2,290	810	340	3,160
ACES	-	675	6	-

- ACES est une architecture destinée aux applications bare-metal.
- La reconfiguration de la MPU prend plus de temps sur uVisor.
- TockOS offre des points d'entrée vers le monde privilégié pour accéder aux périphériques.

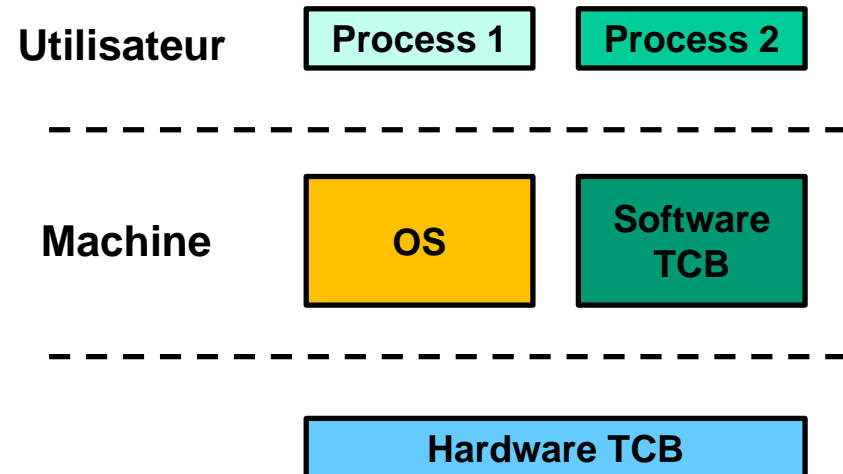
Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine



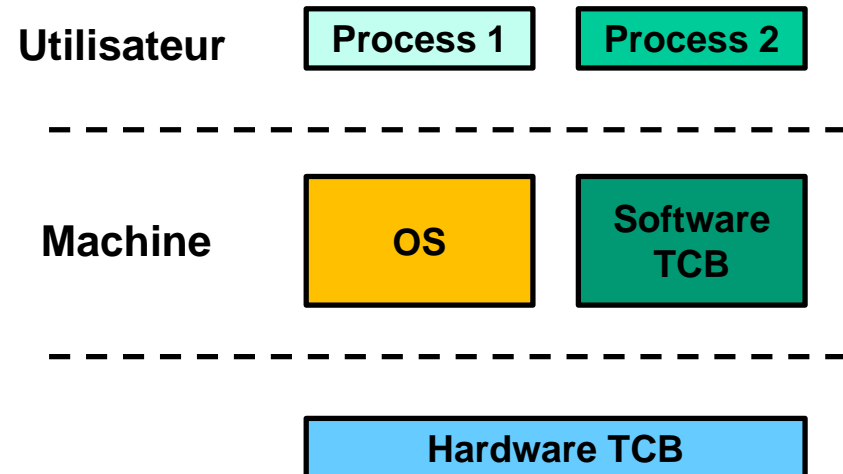
Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine
 - L'OS et le TCB partagent le même privilège.



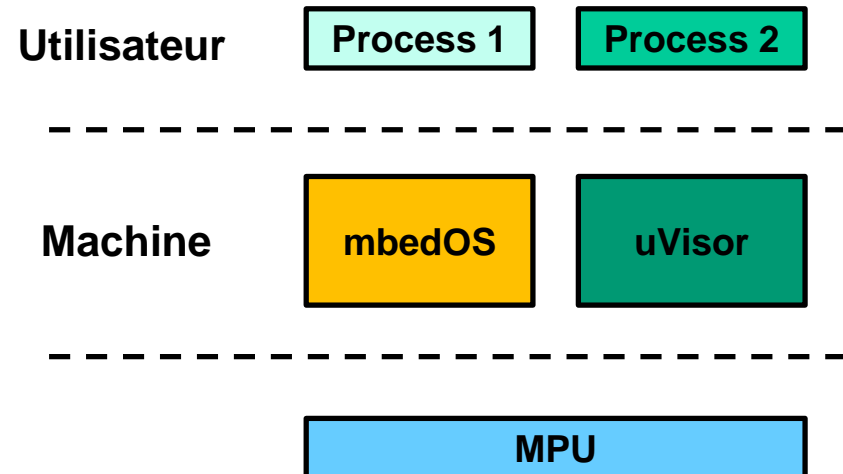
Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine
 - L'OS et le TCB partagent le même privilège.
 - L'OS augmente la surface d'attaque.



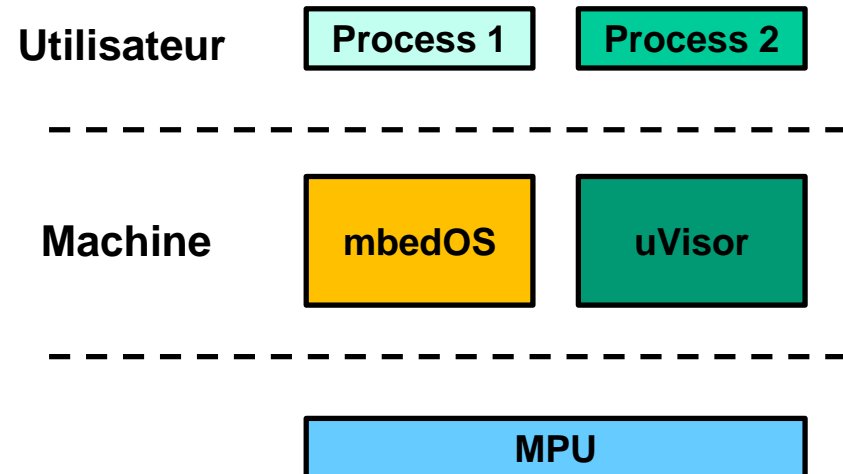
Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine
 - L'OS et le TCB partagent le même privilège.
 - L'OS augmente la surface d'attaque.
 - Exemple:



Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine
 - L'OS et le TCB partagent le même privilège.
 - L'OS augmente la surface d'attaque.
 - Exemple:
mbedOS fait confiance au R12
R12 est accessible depuis le
mode utilisateur



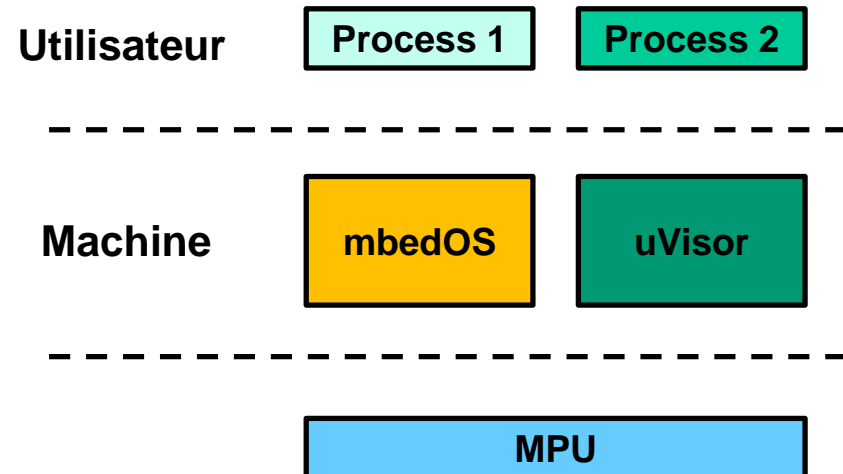
Limitations

- **Seulement deux modes d'exécution**
 - Utilisateur et machine
 - L'OS et le TCB partagent le même privilège.
 - L'OS augmente la surface d'attaque.
 - Exemple:
mbedOS fait confiance au R12
R12 est accessible depuis le
mode utilisateur
Le handler de l'OS :

...

BLX R12

...



Limitations

▪ Seulement deux modes d'exécutions

- Utilisateur et machine
- L'OS et le TCB partagent le même privilège.
- L'OS augmente la surface d'attaque.
- Exemple:

mbedOS fait confiance au R12

R12 est accessible depuis le
mode utilisateur

Le handler de l'OS :

...

```
BLX R12
```

...

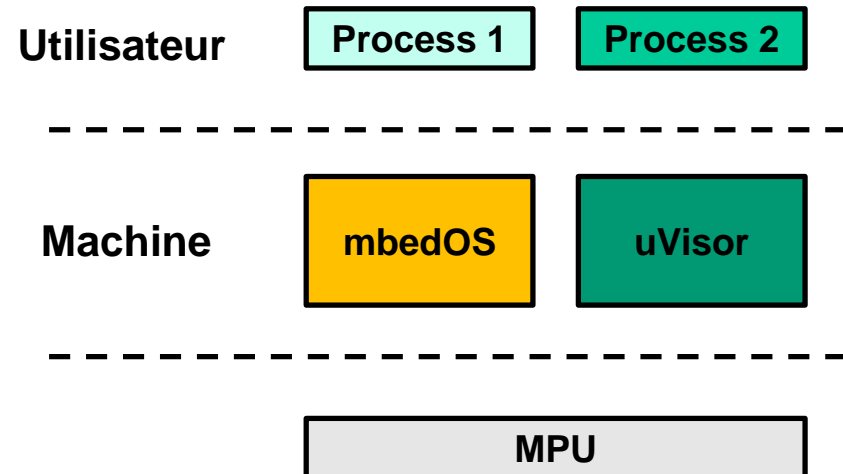
L'attaquant :

```
LDR R12, exploit
```

```
SVC 0
```

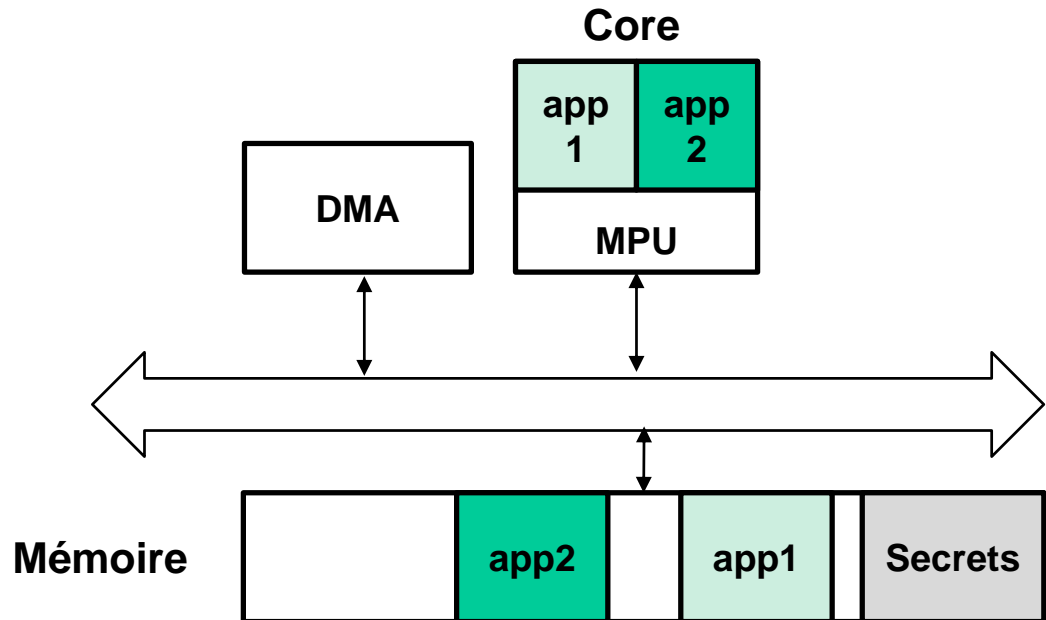
...

=> '*exploit*' est exécuté en mode machine



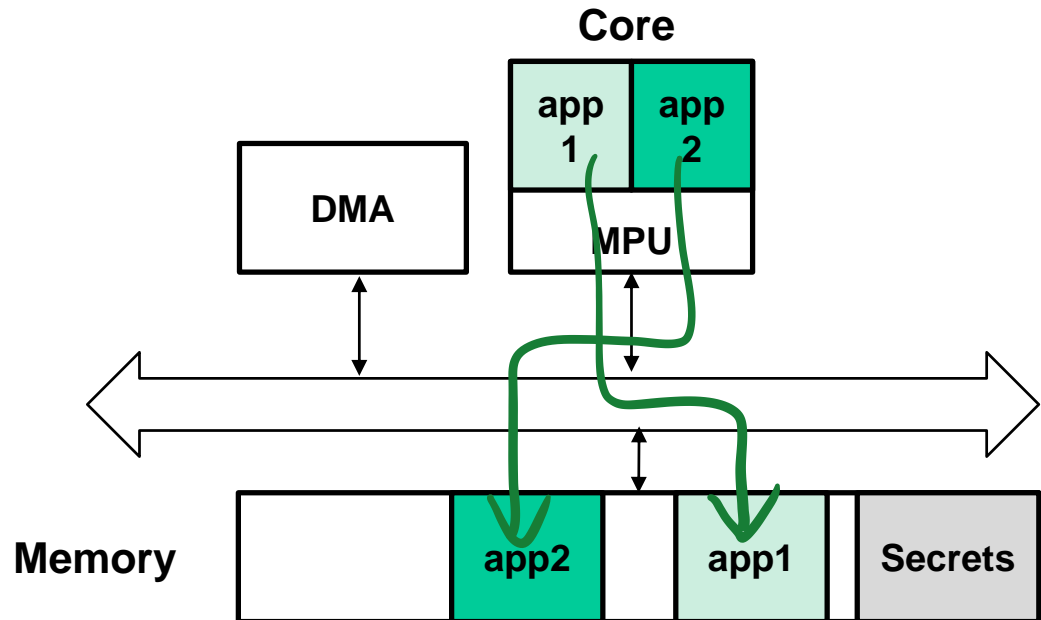
Limitations

- Seulement deux modes d'exécution
- Les autres périphériques avec un accès direct à la mémoire ne sont pas contrôlés par la MPU



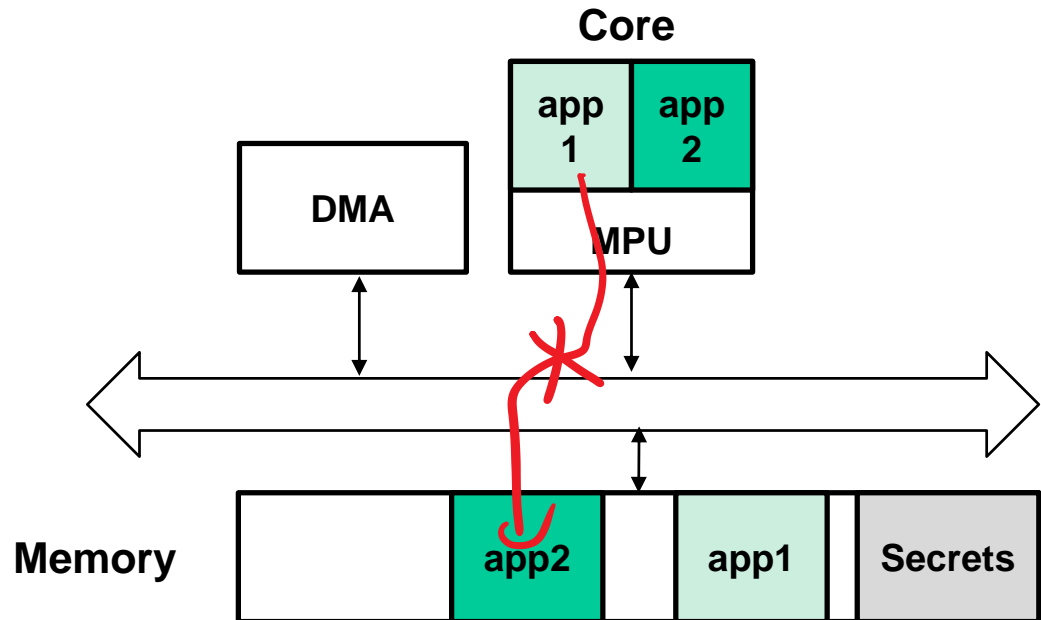
Limitations

- Seulement deux modes d'exécution
- Les autres périphériques avec un accès direct à la mémoire ne sont pas contrôlés par la MPU



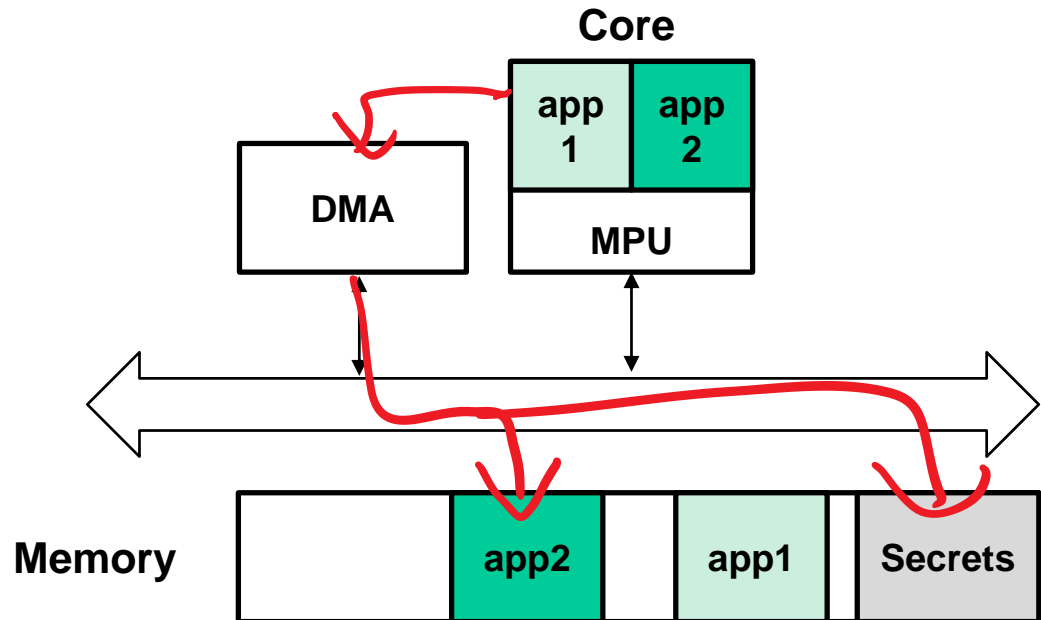
Limitations

- Seulement deux modes d'exécution
- Les autres périphériques avec un accès direct à la mémoire ne sont pas contrôlés par la MPU



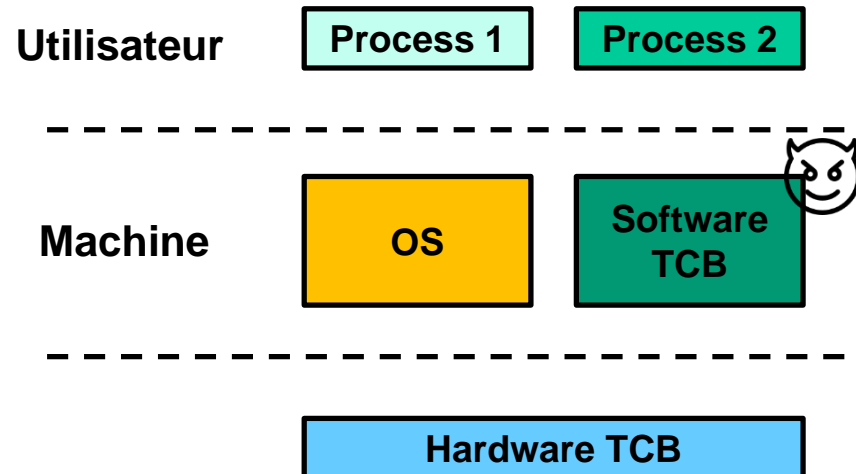
Limitations

- Seulement deux modes d'exécution
- Les autres périphériques avec un accès direct à la mémoire ne sont pas contrôlés par la MPU



Limitations

- Seulement deux modes d'exécution
- Les autres périphériques avec un accès direct à la mémoire ne sont pas contrôlés par la MPU
- Manque d'un Root-of-Trust



Bilan

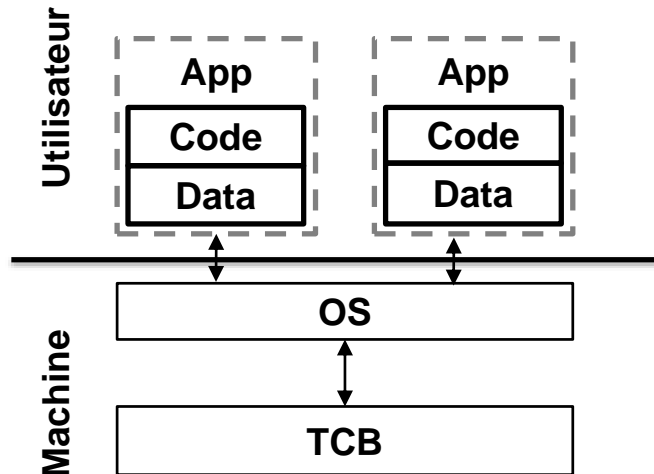
- **Etude de 15 architectures d'isolation et d'attestation**
 - **Etude expérimentale des architectures avec une MPU**
 - Portage des architectures
 - Evaluation de l'apport en protection et des performances.
 - **Identification des limitations des ces architectures**
 - 2 modes d'exécution.
 - Aucun contrôle des accès mémoire pour les périphériques.
 - Manque d'un Root-of-Trust.
- **Publication dans une revue:**
A.Sensaoui, O.Aktouf, D.Hély, S.Di Vito. **An In-depth Study of MPU-based Isolation Architecture**, Journal of Hardware and Systems Security, 2019

Agenda

- I. Prérequis
- II. Evaluation des solutions existantes
- III. Toubkal : Architecture hybride d'isolation et d'attestation**
- IV. Perspectives et Conclusion

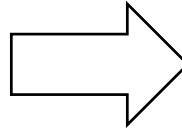
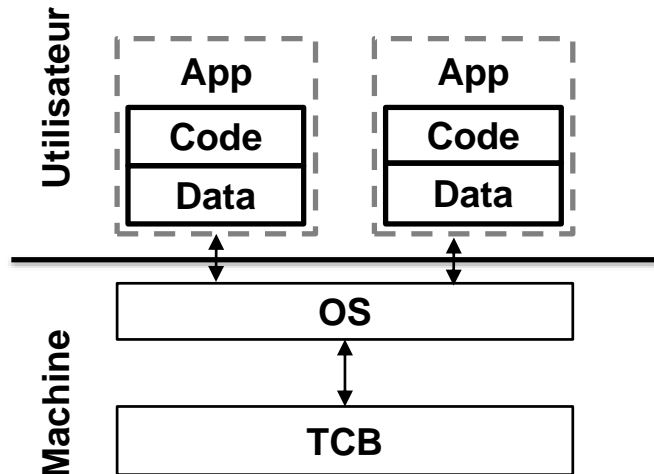
Modèle de Programmation

Le modèle actuel

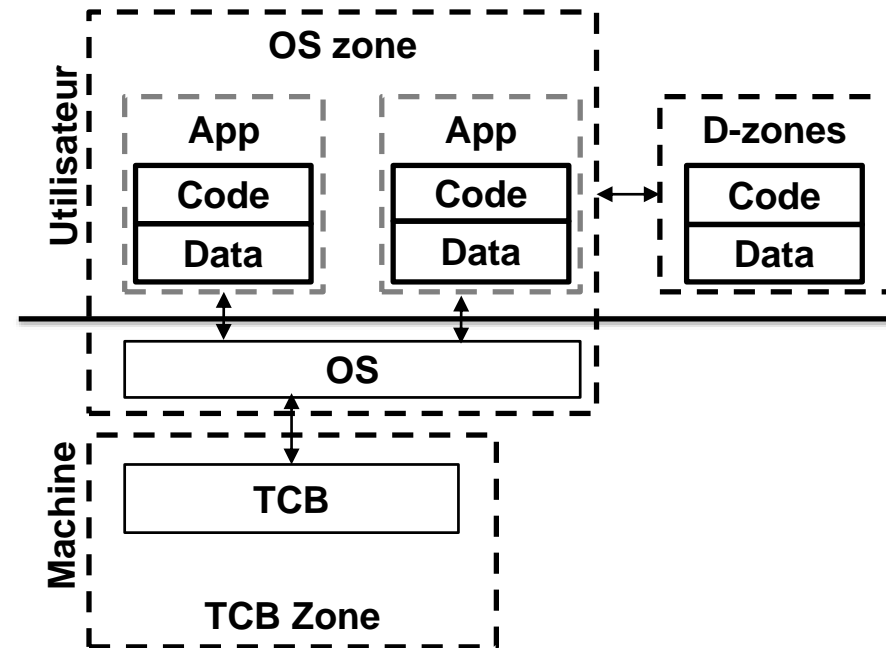


Modèle de Programmation

Le modèle actuel

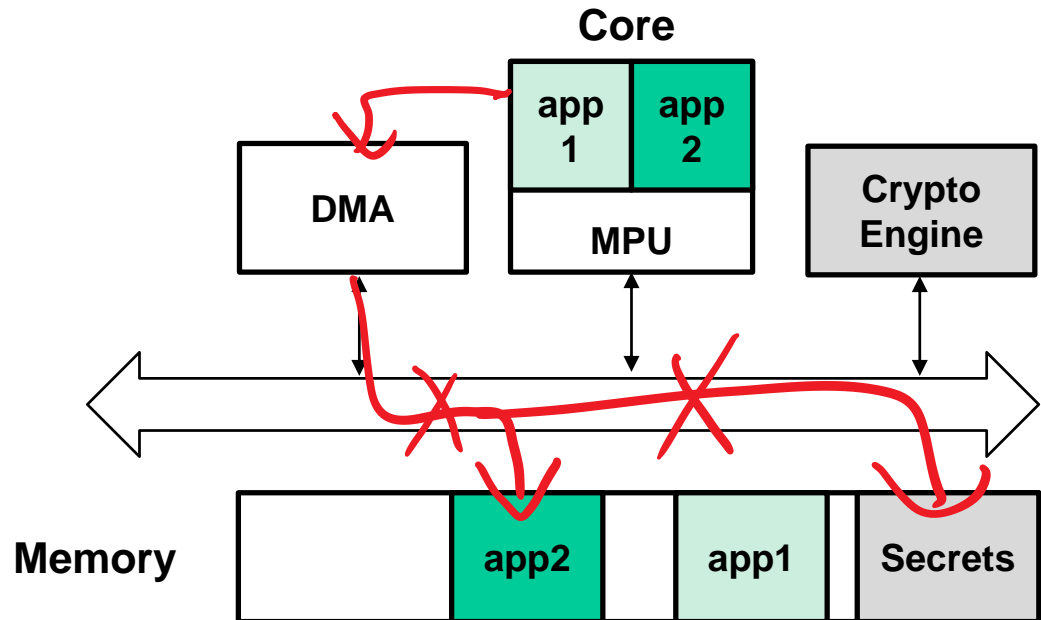


Le modèle proposé



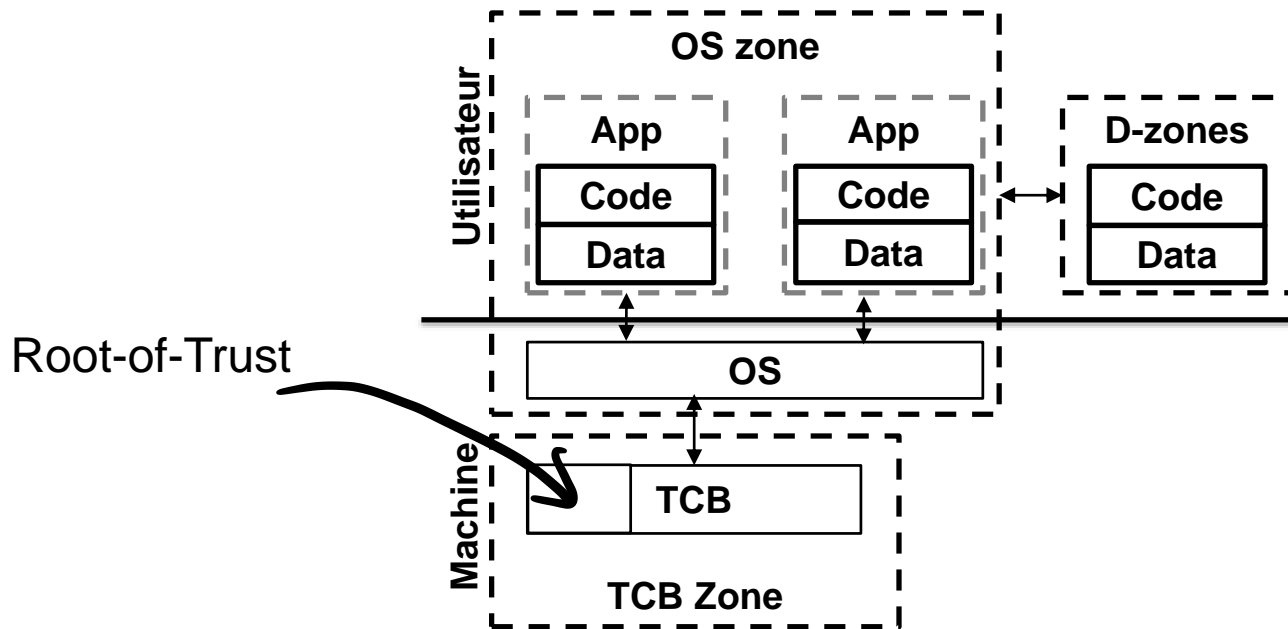
Contrôle des Accès Mémoires

- Pouvoir contrôler les accès mémoire des différents périphériques.

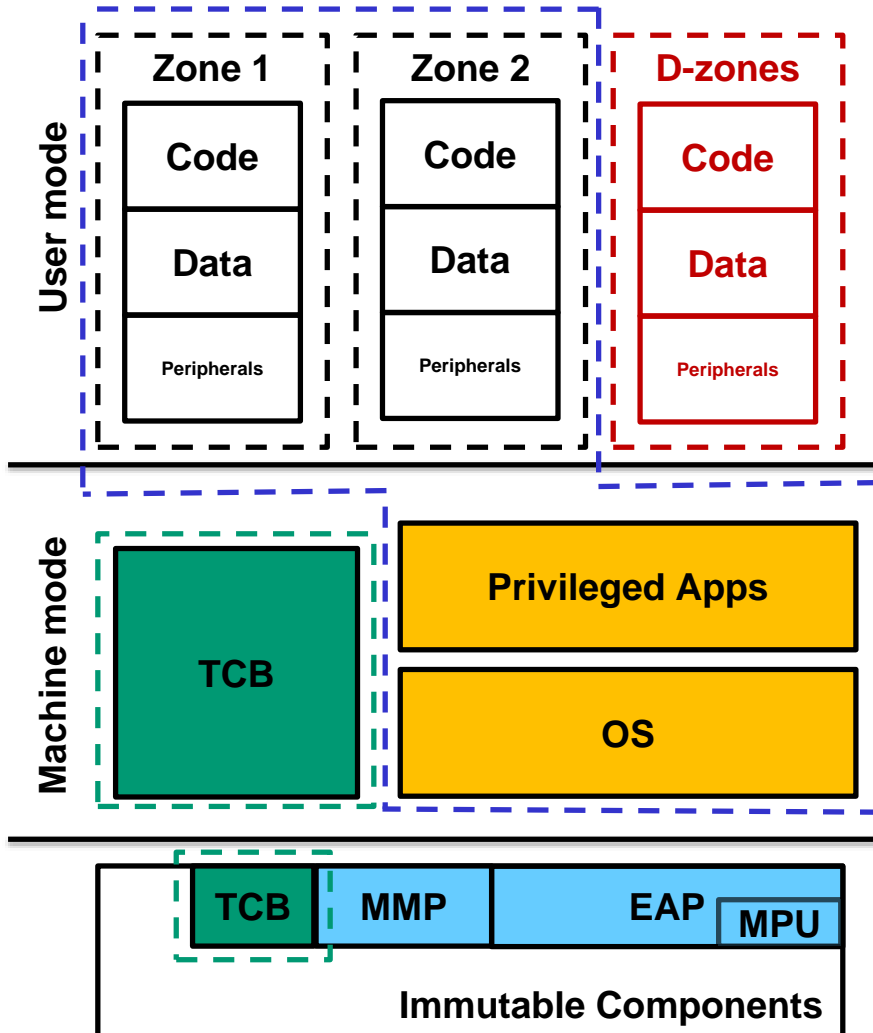


Root-of-Trust

- Avoir un ensemble matériel/logiciel comme base de confiance.



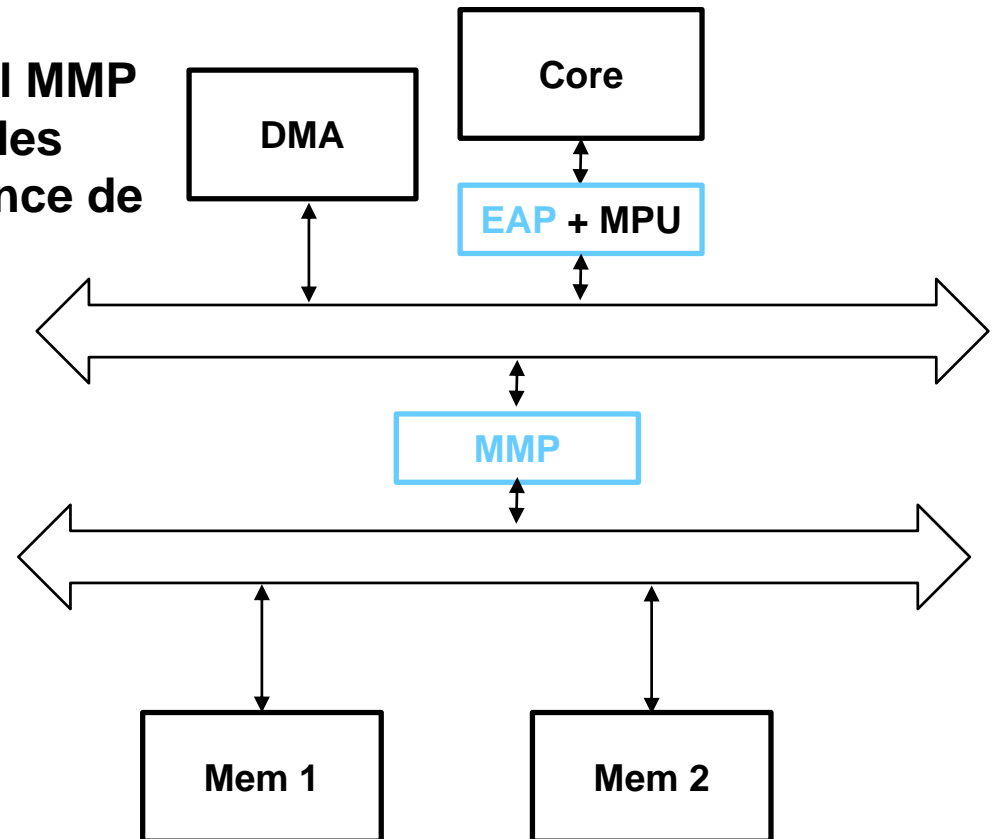
Toubkal : Vue d'Ensemble



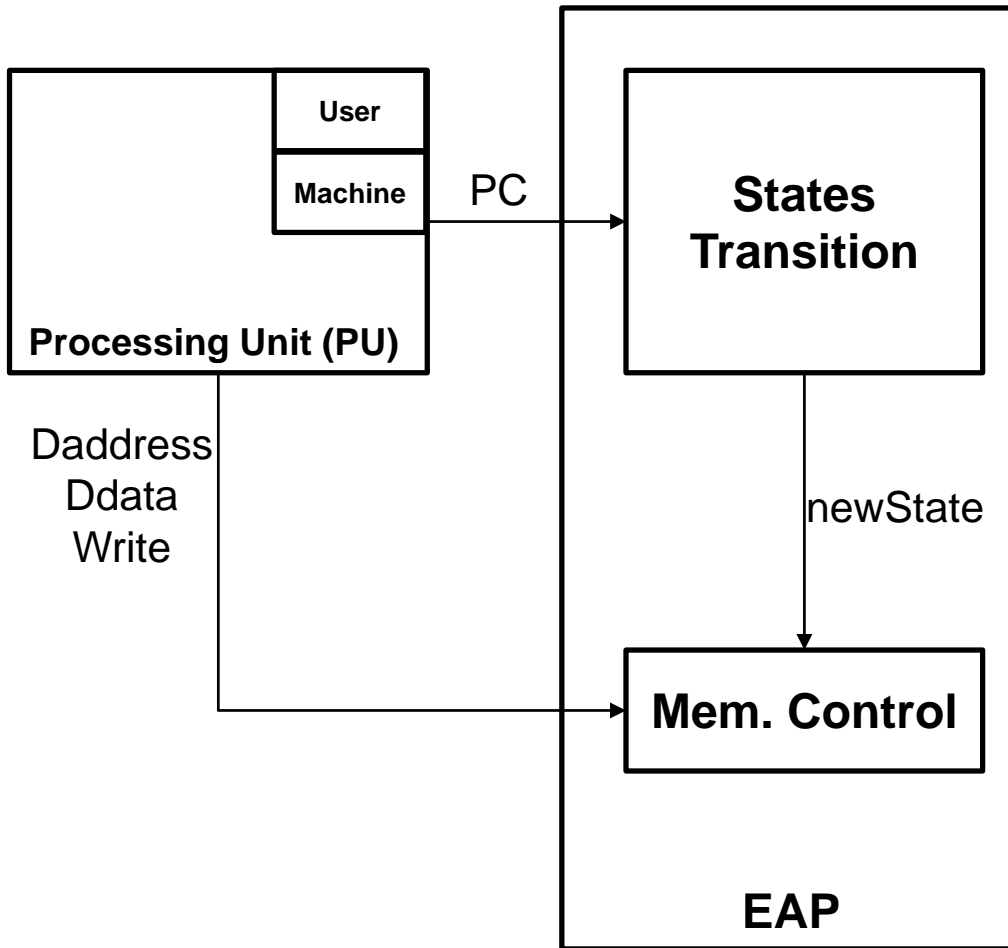
- Co-design matériel/logiciel.
- Composé de deux modules matériel: Execution Aware Protection (EAP), Master Memory Protection (MMP)
- .. Et un TCB logiciel

Toubkal: Vue d'Ensemble

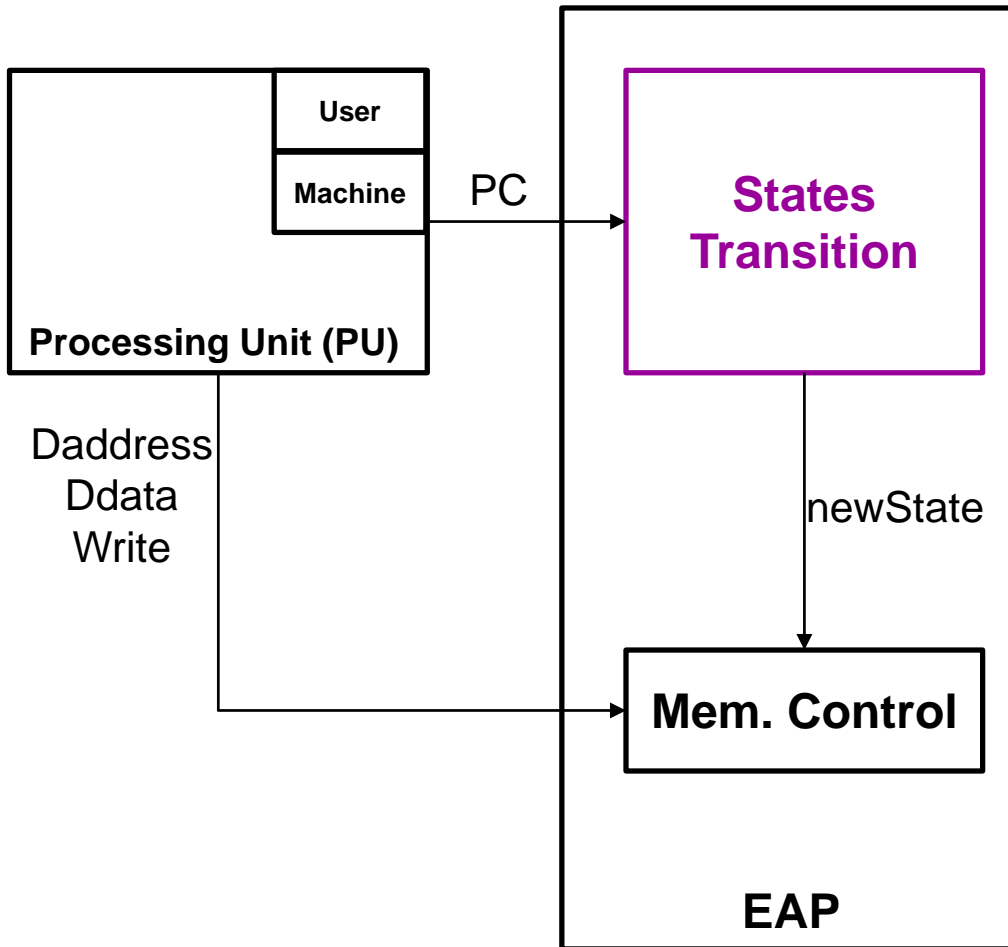
- Un premier module matériel EAP, qui permet de contrôler l'exécution au sein du Cœur.
- Un deuxième module matériel MMP qui permet de contrôler tous les accès mémoires (en provenance de tous les périphériques).



Execution Aware Protection

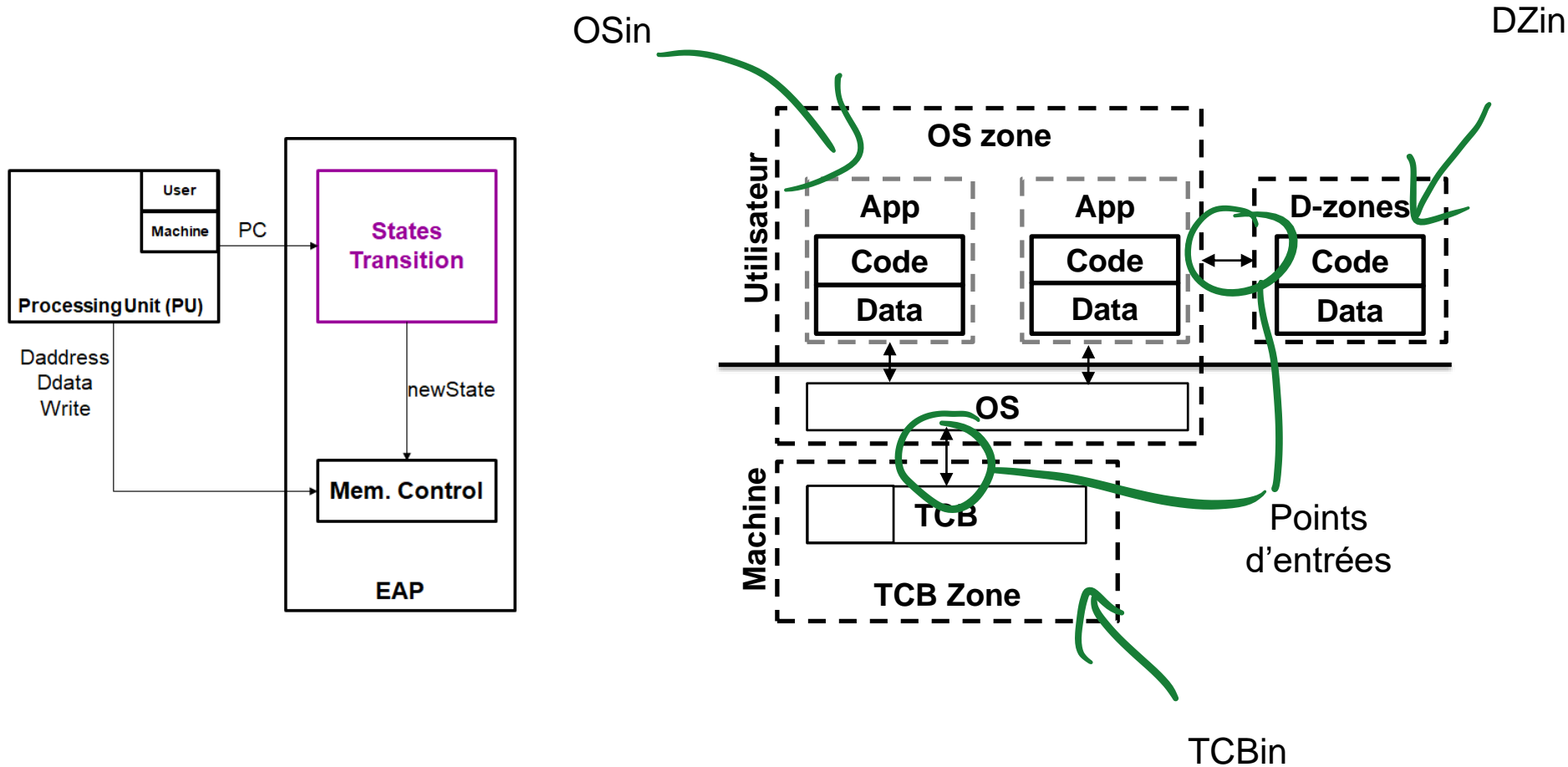


Execution Aware Protection

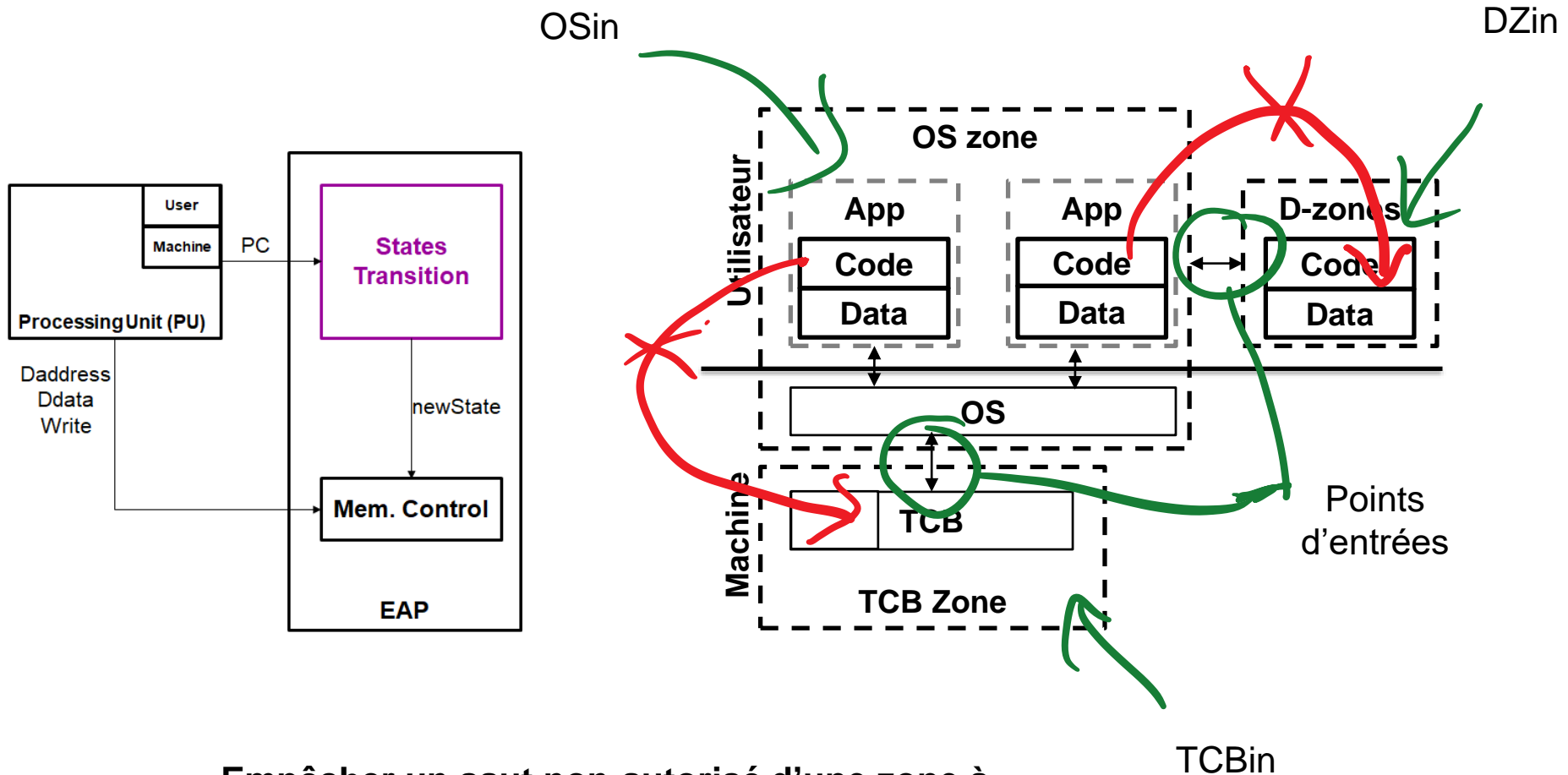


- Le module “**States Transition**” : machine à états qui contrôle toutes les valeurs du compteur ordinal.

Execution Aware Protection

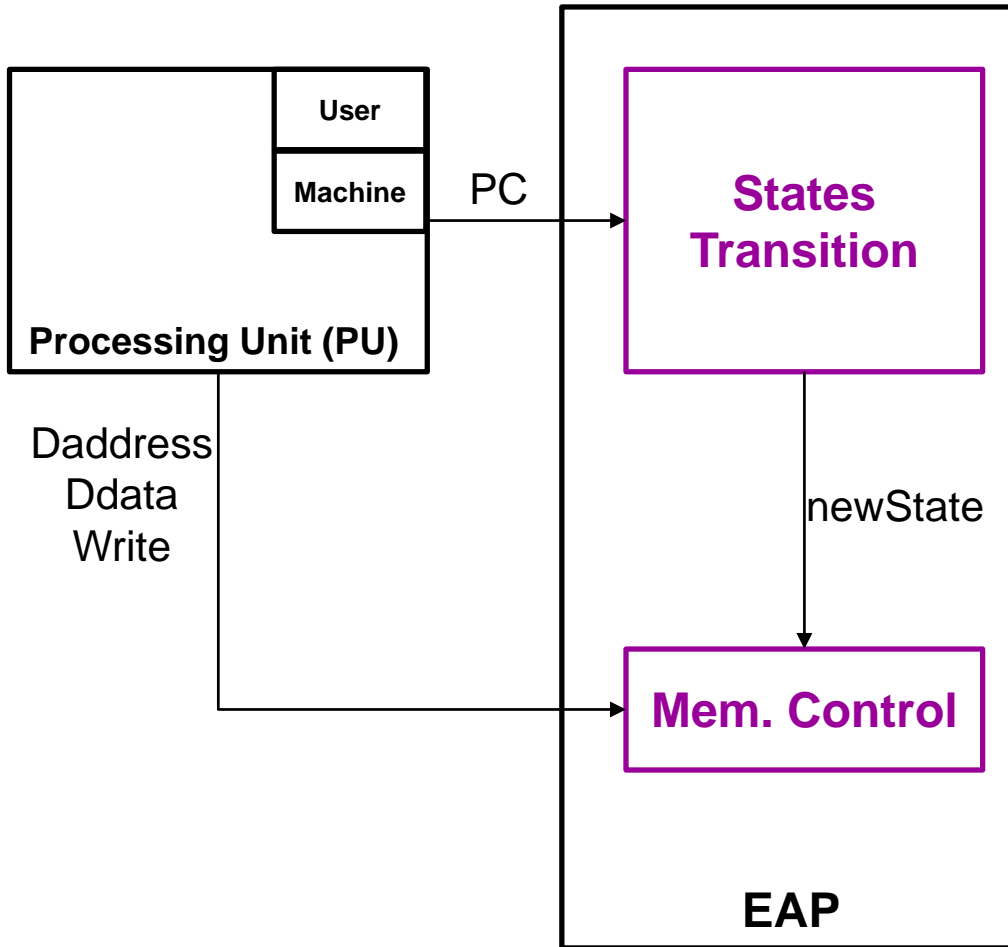


Execution Aware Protection



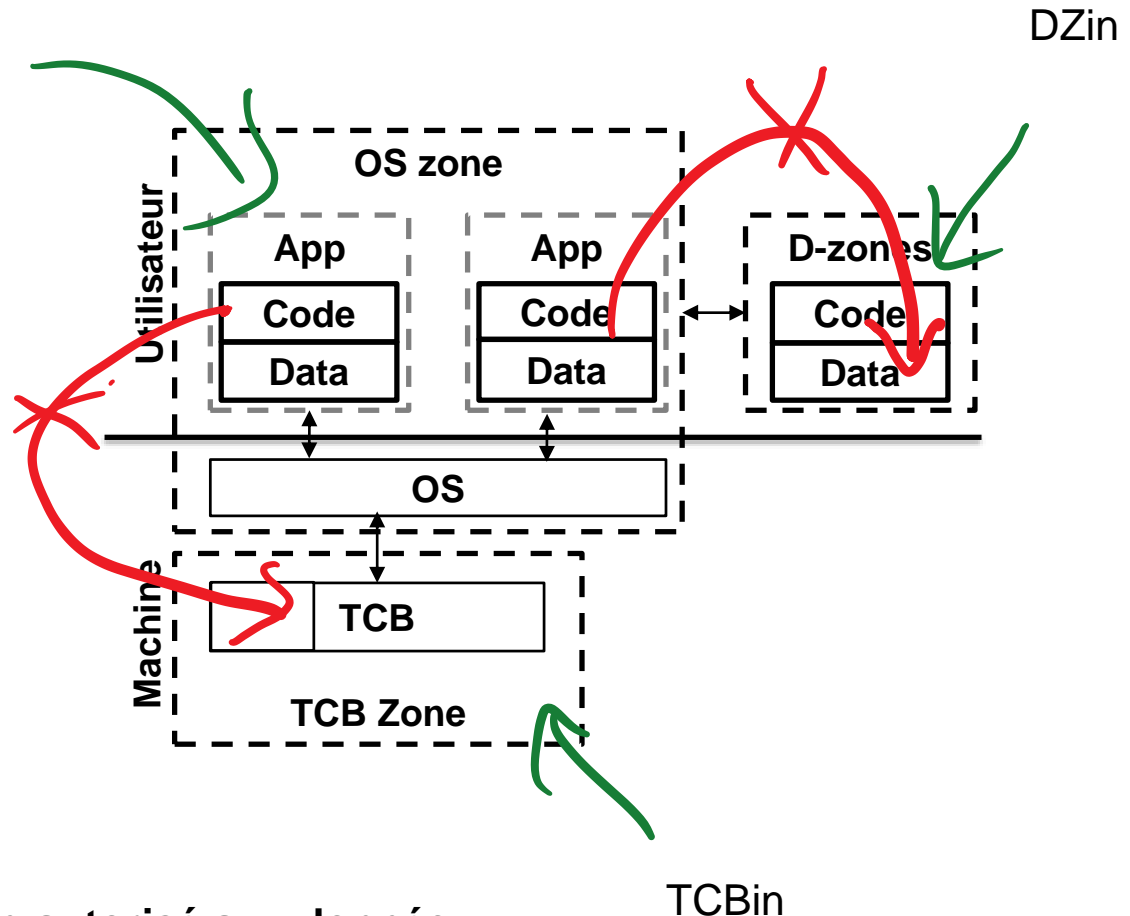
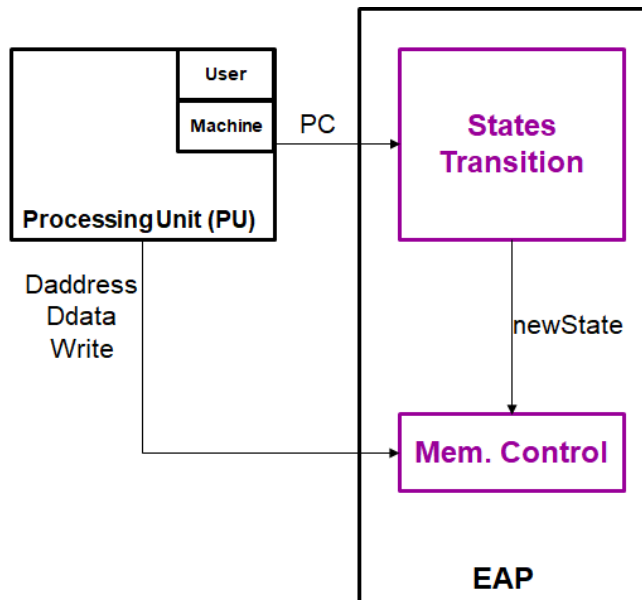
Empêcher un saut non-autorisé d'une zone à une autre

Execution Aware Protection



- **Le module “States Transition”** : machine à états qui contrôle toutes les valeurs du compteur ordinal.
- **Le module “Mem. Control”** : Contrôle les accès mémoire en fonction de l’état du cœur.

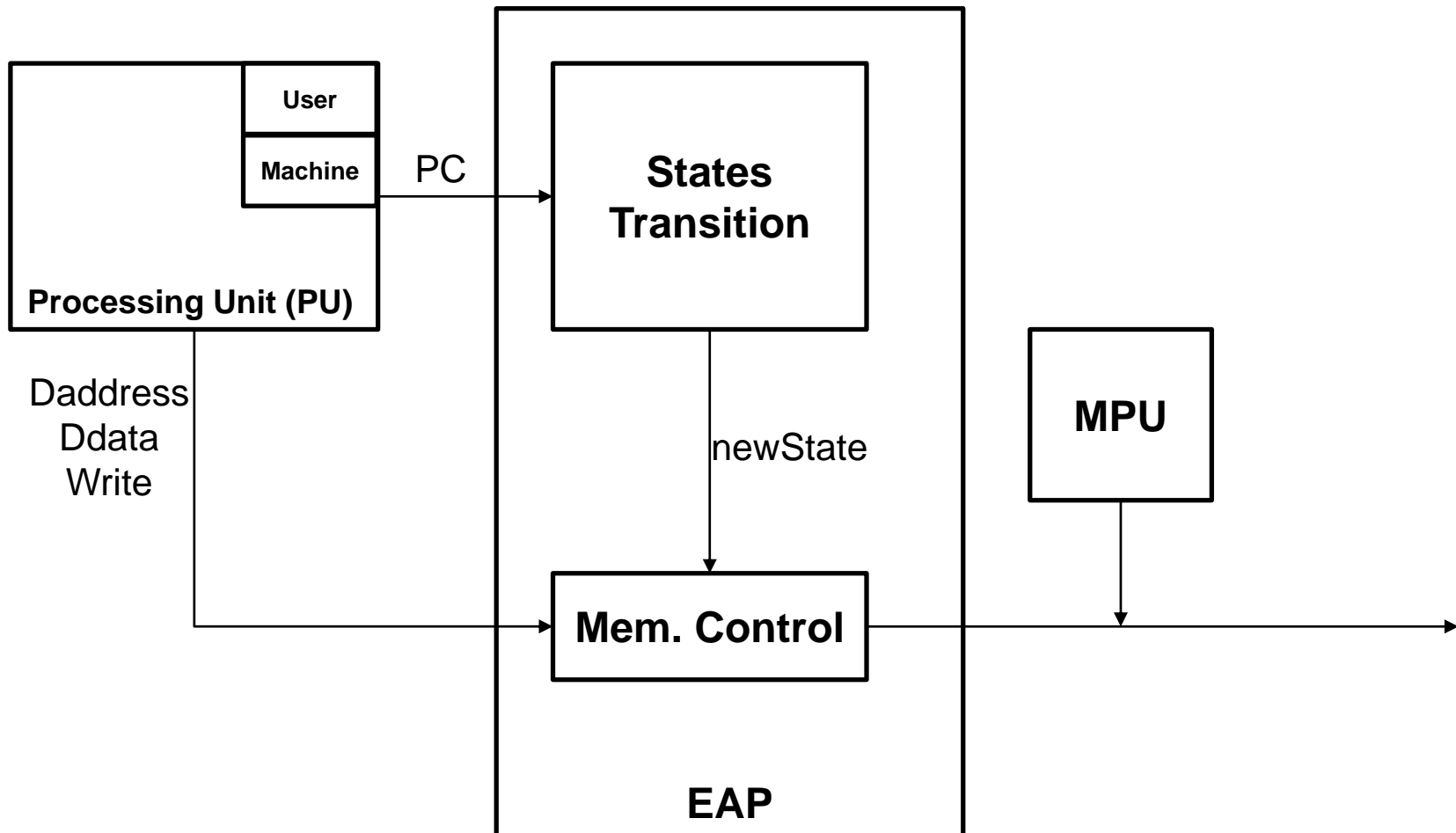
Execution Aware Protection



Empêcher un accès non-autorisé aux données

Délégation MPU

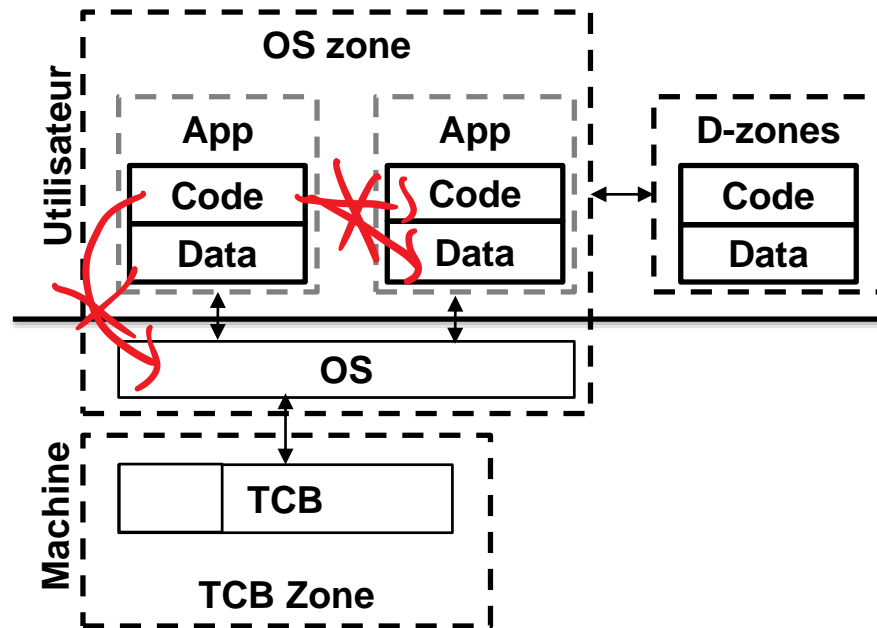
L'EAP peut déléguer le contrôle d'accès mémoire à la MPU



Execution Aware Protection

L'EAP peut déléguer le contrôle d'accès mémoire à la MPU

Exemple: la séparation entre les tâches de l'OS est gérée par la MPU



Execution Aware Protection

- **En utilisant l'EAP avec la MPU, la mémoire est divisée en quatre niveaux :**

niveau 0 contient la zone TCB.

niveau 1 contient les D-Zones.

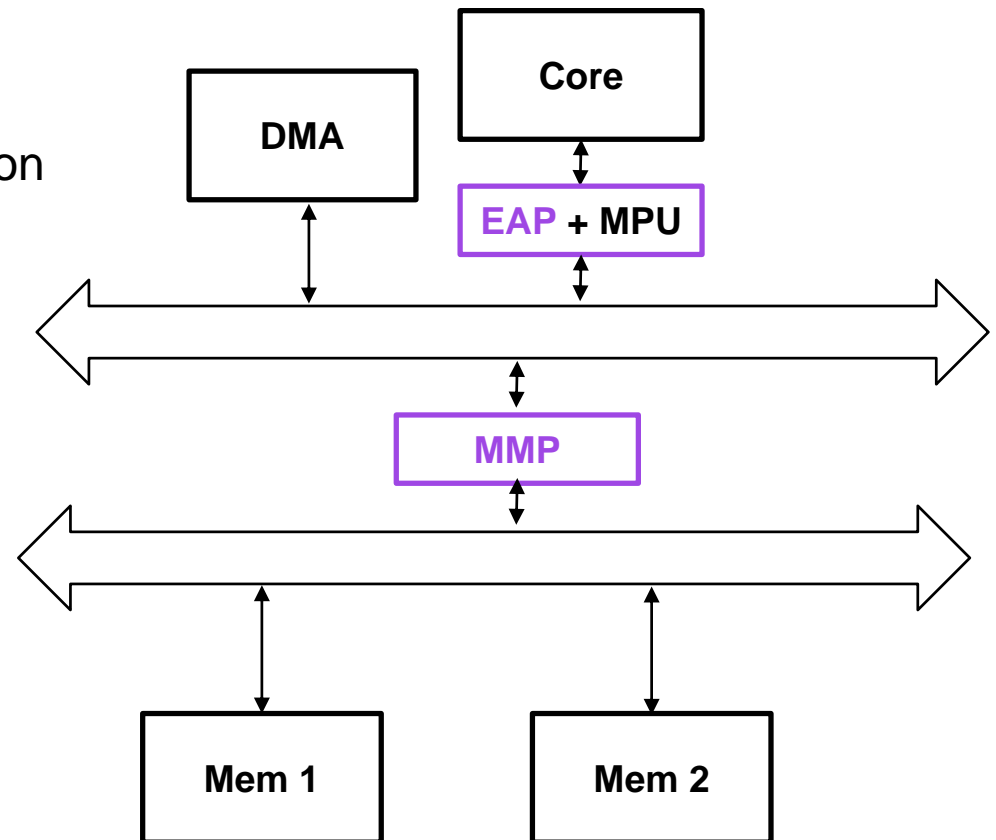
niveau 2 contient la zone OS.

niveau 3 contient les tâches OS

From/To	level 0	level 1	level 2	level 3
level 0	rwX	rw	rw	rw
level 1	-	rwX	-	-
level 2	-	-	rwX	rw
level 3	-	-	-	rwX

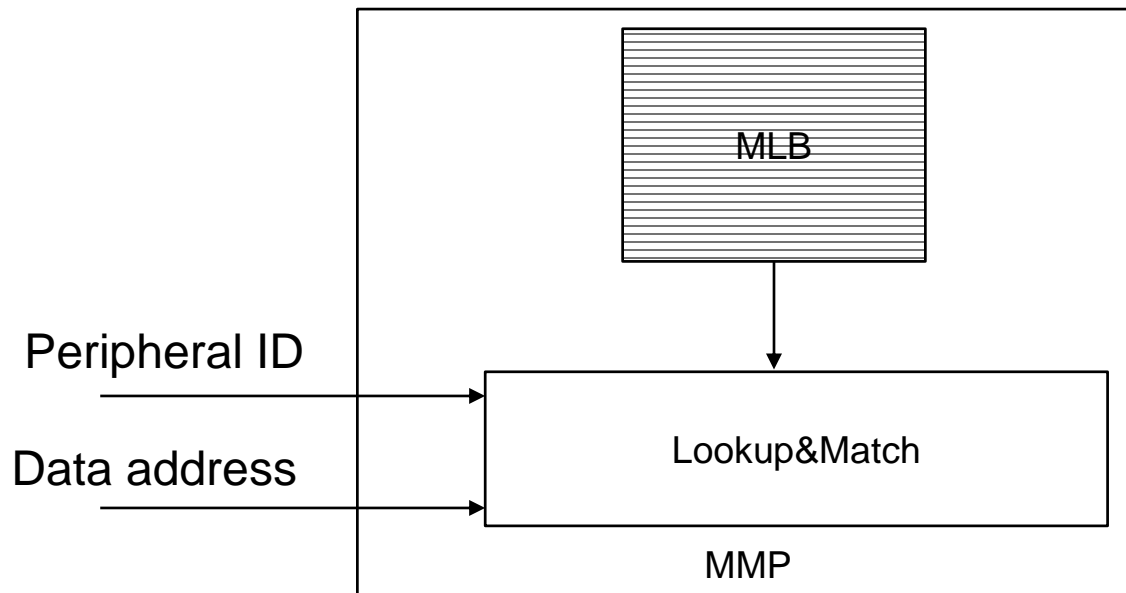
Master Memory Protection

- Contrôler les accès mémoire des différents périphériques.
- Elle se configure de la même façon que la MPU.
Contrairement à la MPU :
 - Elle prend en compte l'identifiant du périphérique qui fait l'accès à la mémoire.
 - On configure les régions de mémoire interdites aux périphériques.



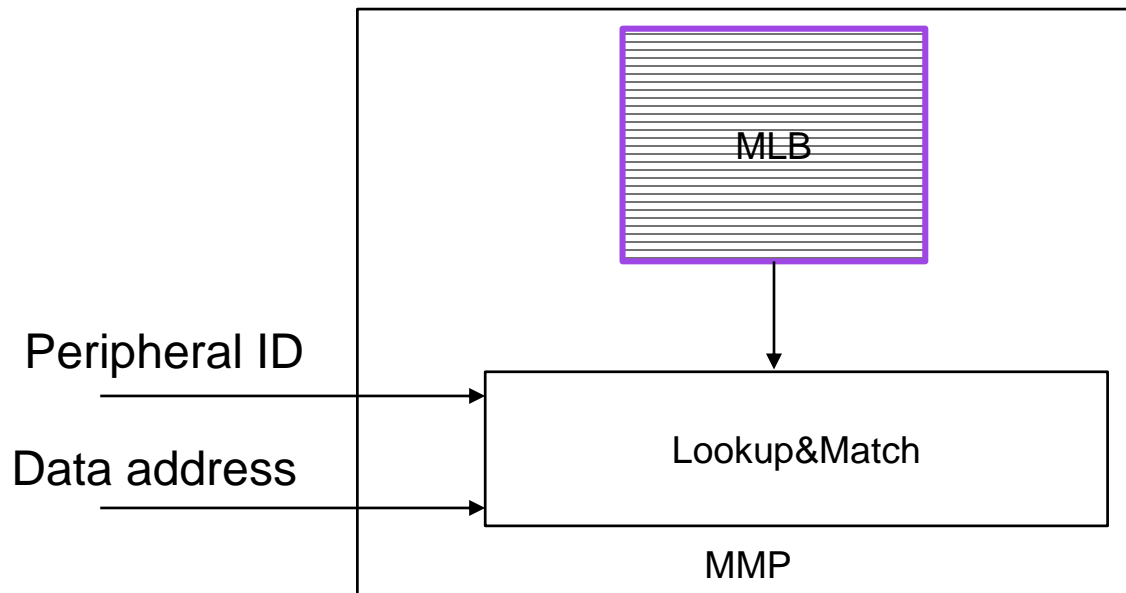
Master Memory Protection

- Composé de :

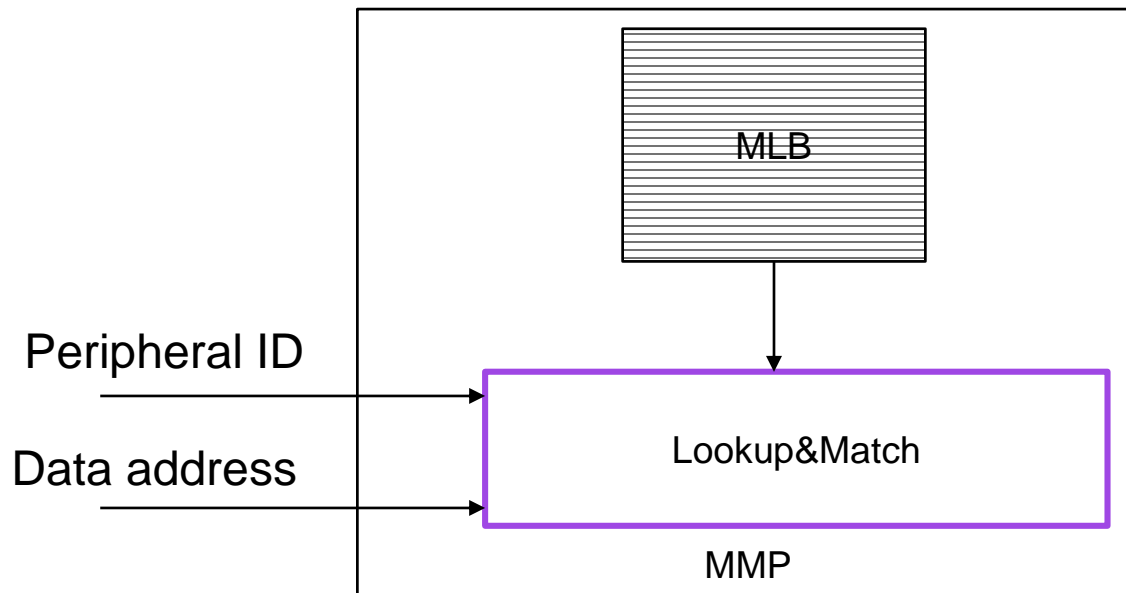


Master Memory Protection

- **Composé de:**
 - **MLB (Master Look-aside Buffer)** : contient la configuration des permissions des regions mémoire.



Master Memory Protection



- **Composé de :**
 - **MLB (Master Look-aside Buffer)** : contient la configuration des permissions des régions mémoire.
 - **Lookup&Match** : Vérifie les accès mémoire.

Evaluation

- **Cible FPGA :**
Nexys4 DDR.

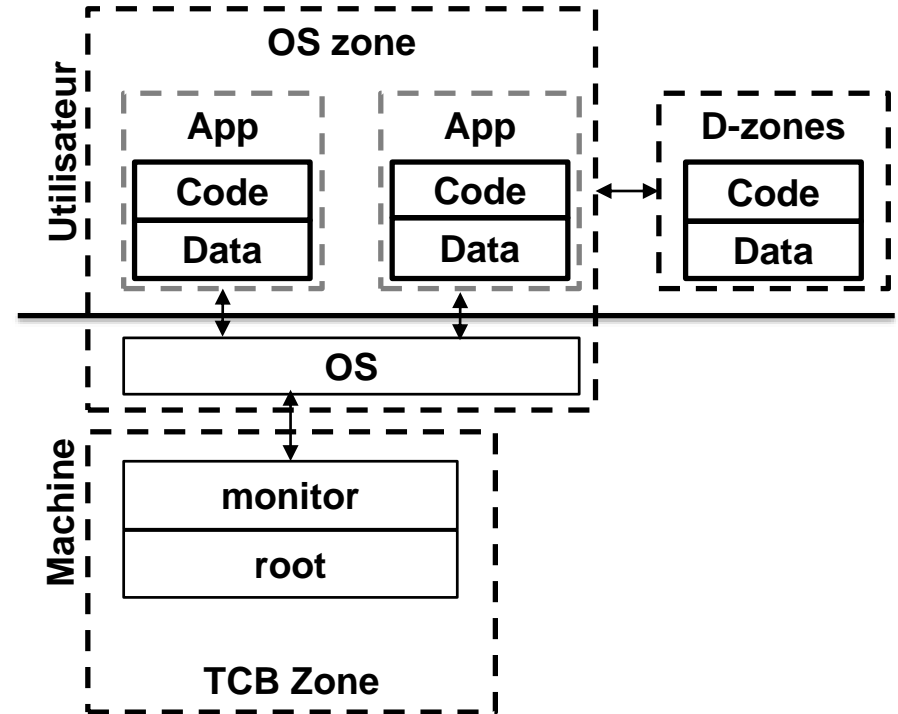
- **Configuration :**
Basé sur un Rocket Chip 32-bit mono-coeur.
64KB de RAM-like, une fréquence d'horloge de 36MHz.
Une EAP avec une seule D-zone et une MMP avec 4 configurations.
Ces deux modules ont été conçus en CHISEL3/Scala

- **Coût matériel de Toubkal :**

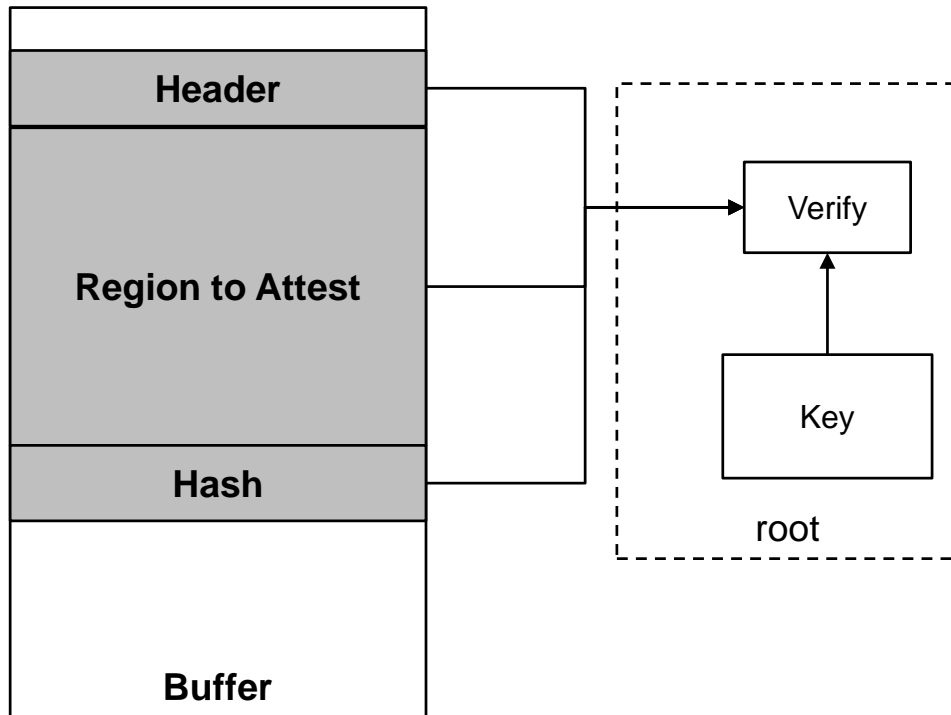
	Look-up Tables	Flip-Flops
Configuration de base	23,100	8,100
Coût de Toubkal	1,200 (5.1%)	800 (10%)

TCB Logiciel

- Il offre des mécanismes pour renforcer la sécurité du système.
- Composé de deux parties : le root qui est non-modifiable (**root-of-trust**) et le monitor .
- La responsabilité du root est la configuration de l'EAP et la verification de l'authenticité du monitor avant son exécution.

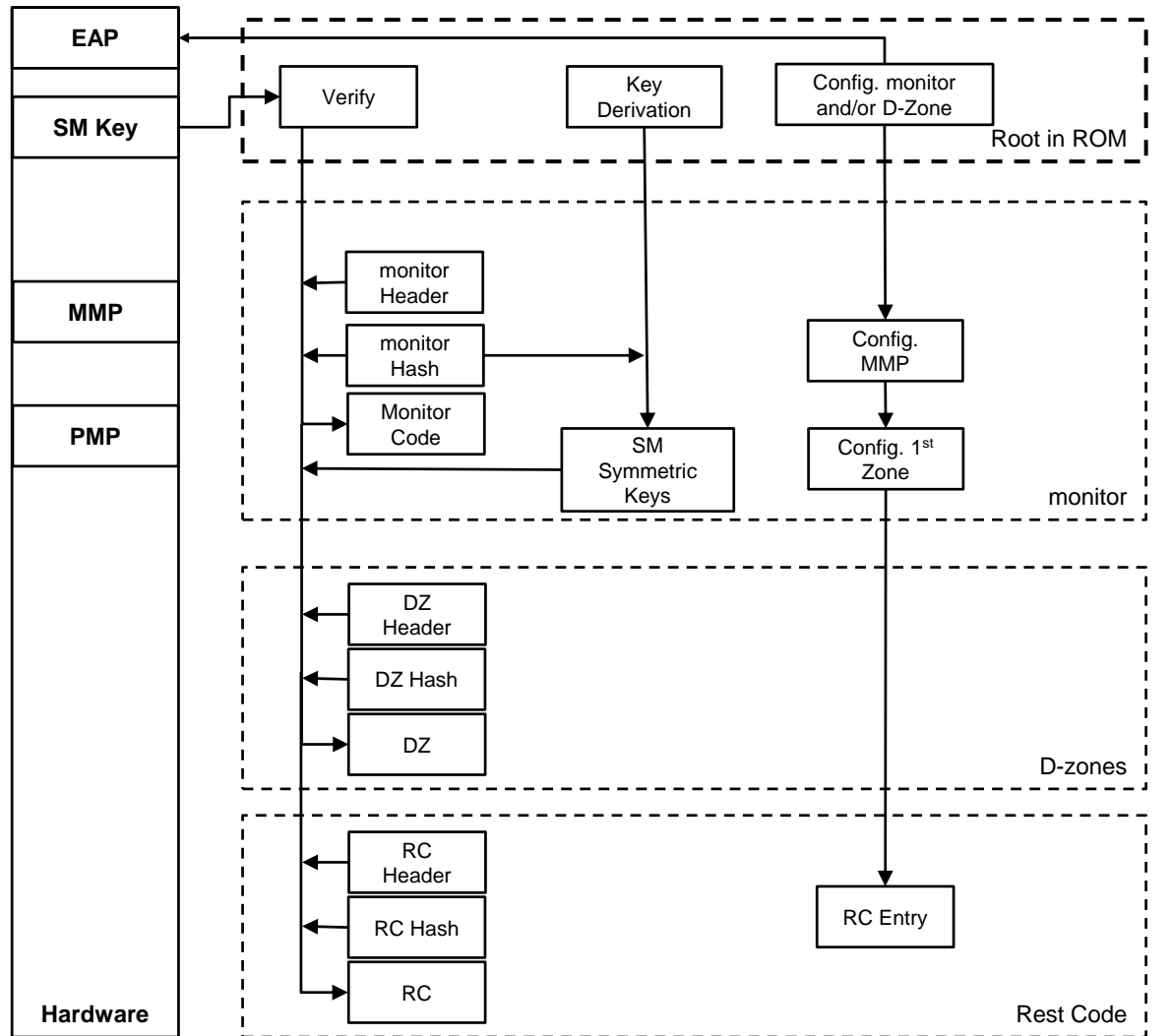


La vérification d'authenticité



- Être capable de vérifier l'authenticité de toute section mémoire.
- La fonction « Verify » récupère le début et la fin de la section, une clé symétrique, et le cash qui a été pré-calculé.
- Cette fonction calcule le hash en utilisant la clé et un algorithme HMAC, et ensuite compare le hash résultant avec celui fourni.

Démarrage du système



Vérification du monitor.

Le root est non-modifiable et il est responsable.

Génération des clés symétriques.

Après vérification du hash du monitor, ce dernier est utilisé pour générer deux clés symétriques.

Configuration de l'Execution Aware Protection.

Configuration du Master Memory Protection.



Vérification des autres applications

Les autres applications sont vérifiées avec les clés générées.

Evaluation

- **Cible FPGA :**
Nexys4 DDR.
- **Configuration :**
Basé sur un RISC-V, 32-bit mono-coeur.
64KB de RAM-like, fréquence d'horloge de 36MHz.
- **Coût logiciel de Toubkal :**
Environ 5kB en ROM
- **Coût démarrage :**
Hypothèse : un monitor de 8kB
Environ 80ms

Evaluation

- **Cible FPGA :**
Nexys4 DDR.
- **Configuration :**
Basé sur un RISC-V, 32-bit mono-coeur.
64KB de RAM-like, fréquence d'horloge de 36MHz.
- **Coût logiciel de Toubkal :**
Environ 5kB en ROM 
- **Coût démarrage :**
Hypothèse: un monitor de 8kB
Environ 80ms 

Evaluation

- Que se passe-t-il si uVisor est utilisé comme monitor avec mbedOS?

mbedOS fait confiance au R12
 R12 est accessible depuis le
 mode utilisateur
 Le handler de l'OS :

...

BLX R12

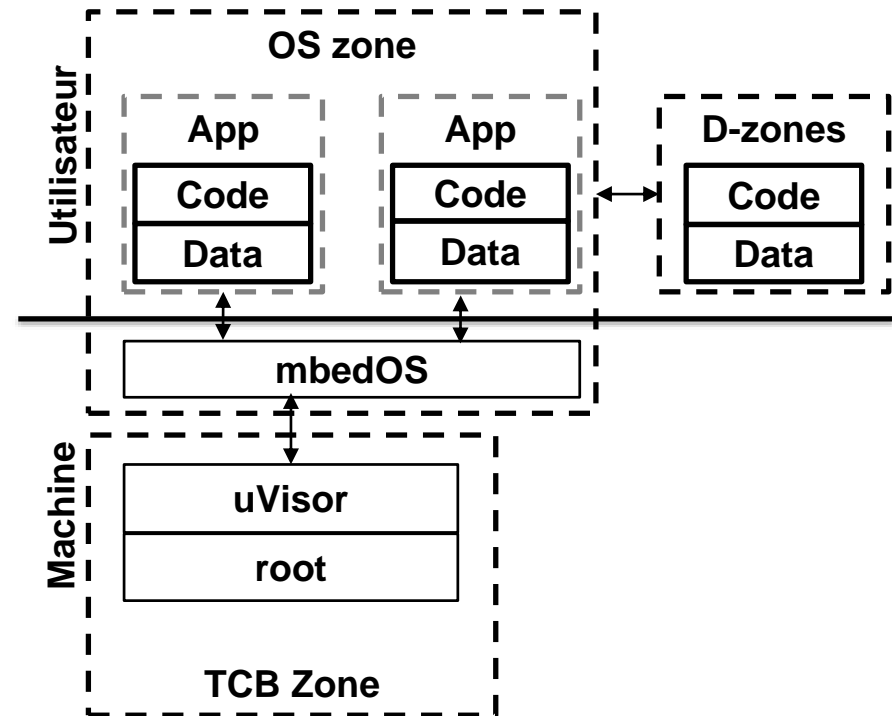
...

L'attaquant :

LDR R12, exploit

SVC 0

...



Evaluation

- Que se passe-t-il si uVisor est utilisé comme monitor avec mbedOS?

mbedOS fait confiance au R12
 R12 est accessible depuis le
 mode utilisateur
 Le handler de l'OS :

...

BLX R12

...

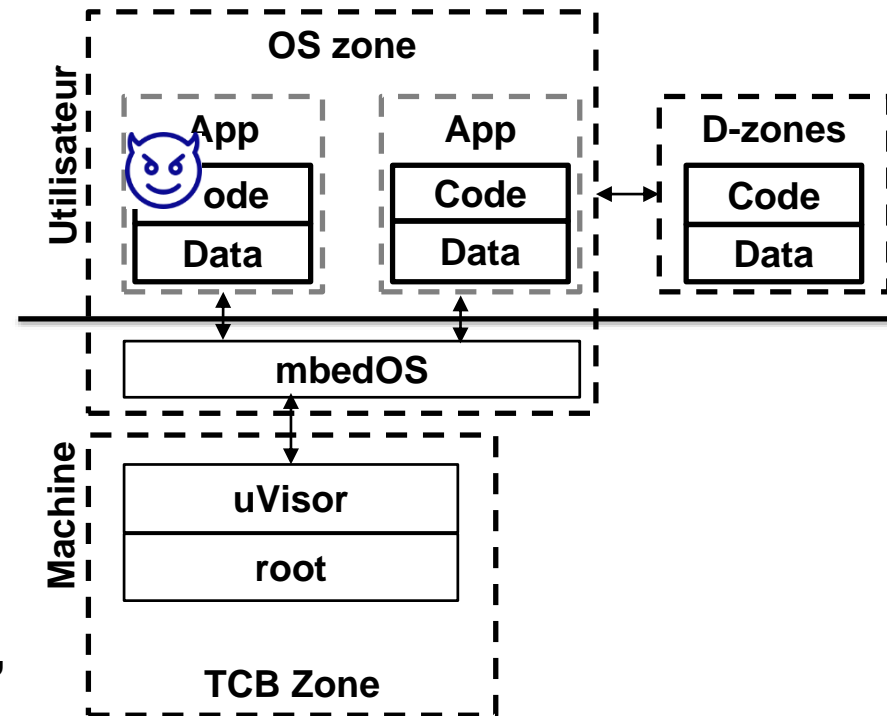
L'attaquant :

LDR R12, exploit

SVC 0

...

=> **"exploit"** exécuté en mode machine,
 mais, il reste coincé dans l'OS zone.



Analyse et vérification des propriétés de protection

Comment être sûr que Toubkal nous garantie le niveau de sécurité souhaité?

- La vérification de modèle peut promettre des garanties sur le niveau de sécurité.

Analyse et vérification des propriétés de protection

Vérification de modèle:

- Vérification du logiciel
 - Utilisation d'une librairie vérifiée formellement : Hacl*.
 - Elle couvre la majorité du root.
- Vérification du matériel.
 - On se concentre sur la vérification de l'EAP.

Analyse et vérification des propriétés de protection

La vérification de l'EAP consiste en 3 étapes :

- Modéliser l'EAP en machine à états.
- Spécifier les propriétés de sécurité que l'EAP doit garantir.
- Vérifier que le modèle respecte ces propriétés.

Analyse et vérification des propriétés de protection

Notre approche:

- Définir les propriétés de sécurité et les traduire en spécification LTL (Logique Temporelle Linéaire). => ~18 spécifications LTL.
- Utilisation de l'outil NuSMV pour vérifier si notre modèle satisfait ces spécifications.

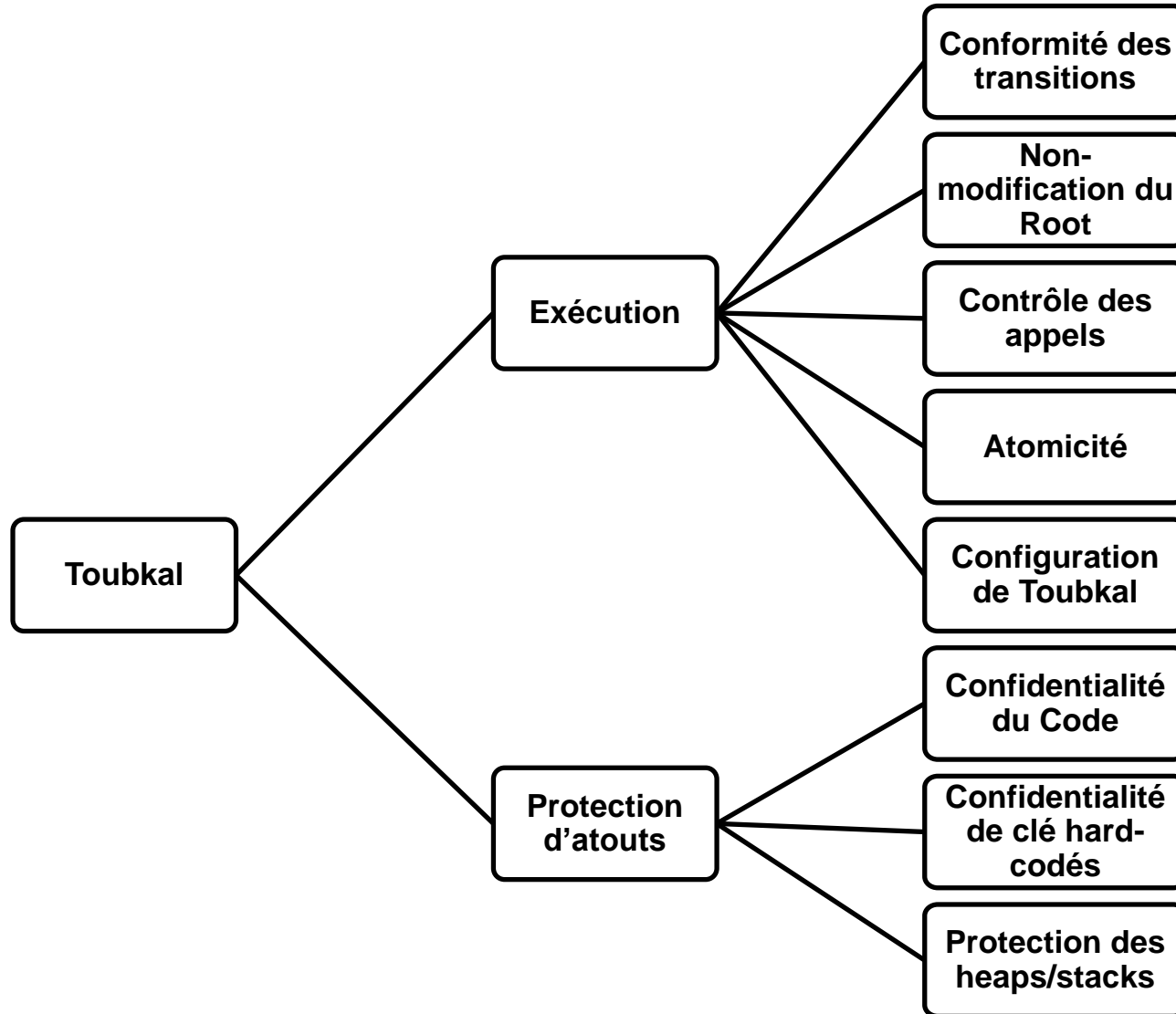
Logique Temporelle Linéaire (LTL)

Future "F": Fp est vrai, si p est vrai dans un future état.

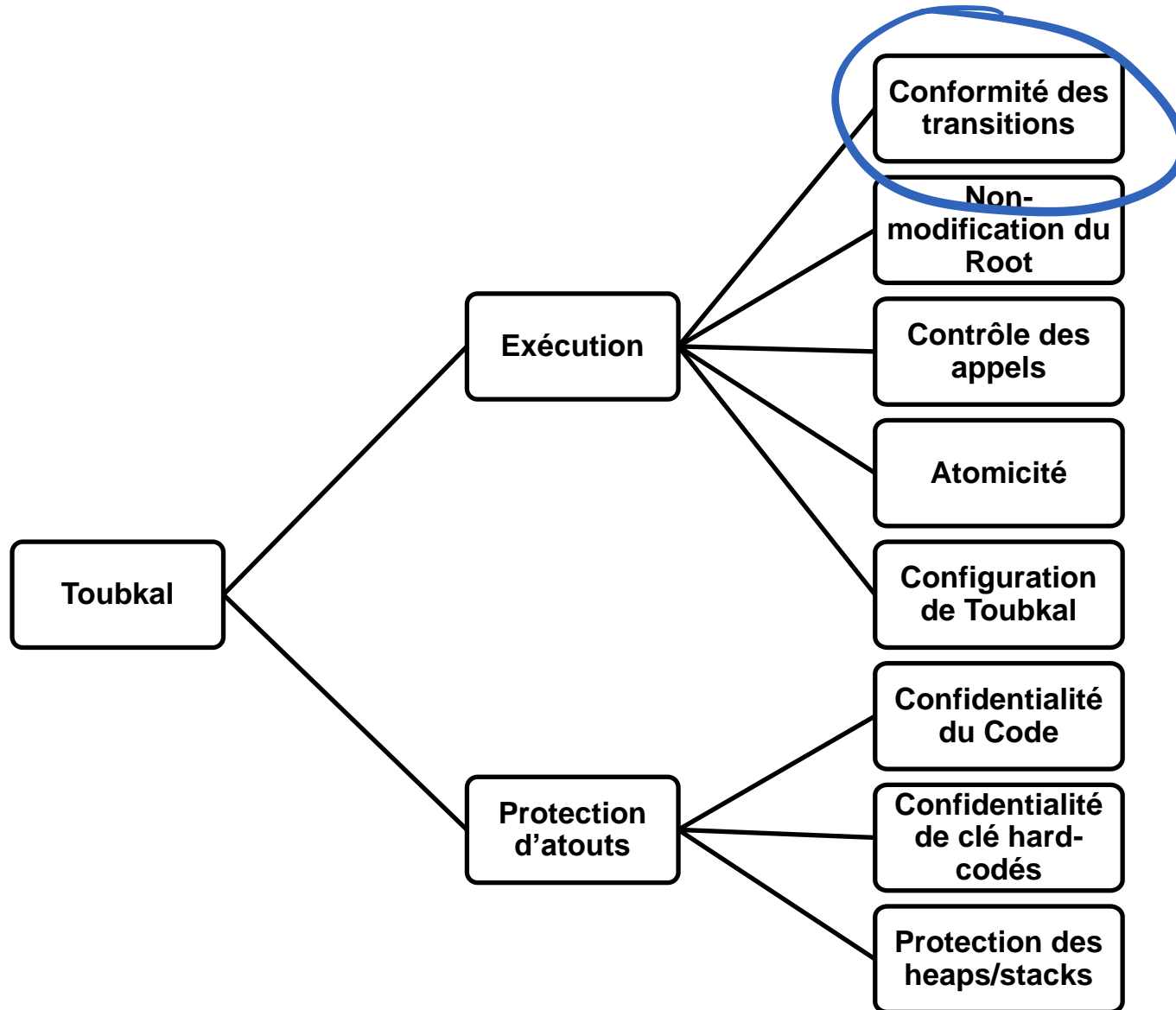
Global "G": Gp est vrai, si p est toujours vrai.

neXt "X": Xp est vrai, si p est vrai à l'état suivant.

Propriétés de sécurité

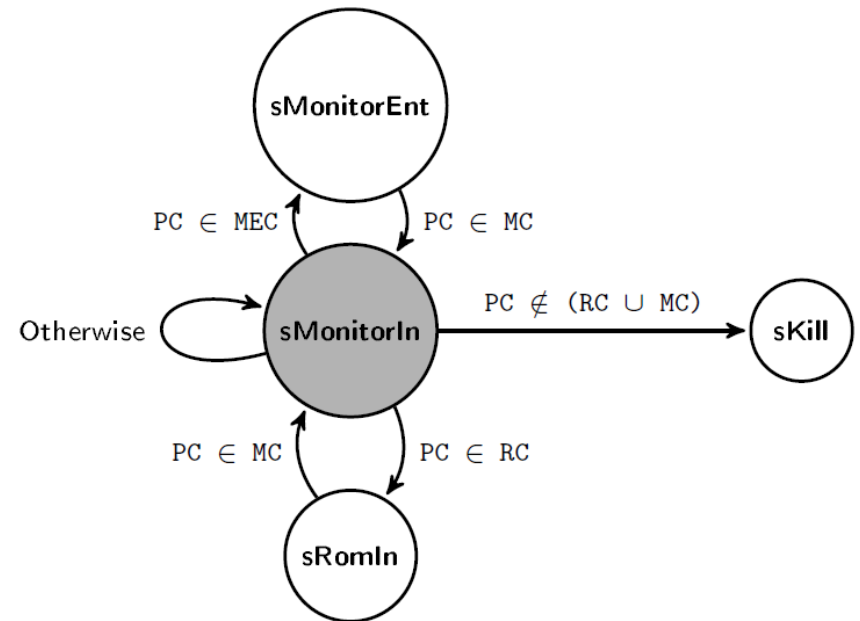


Propriétés de sécurité



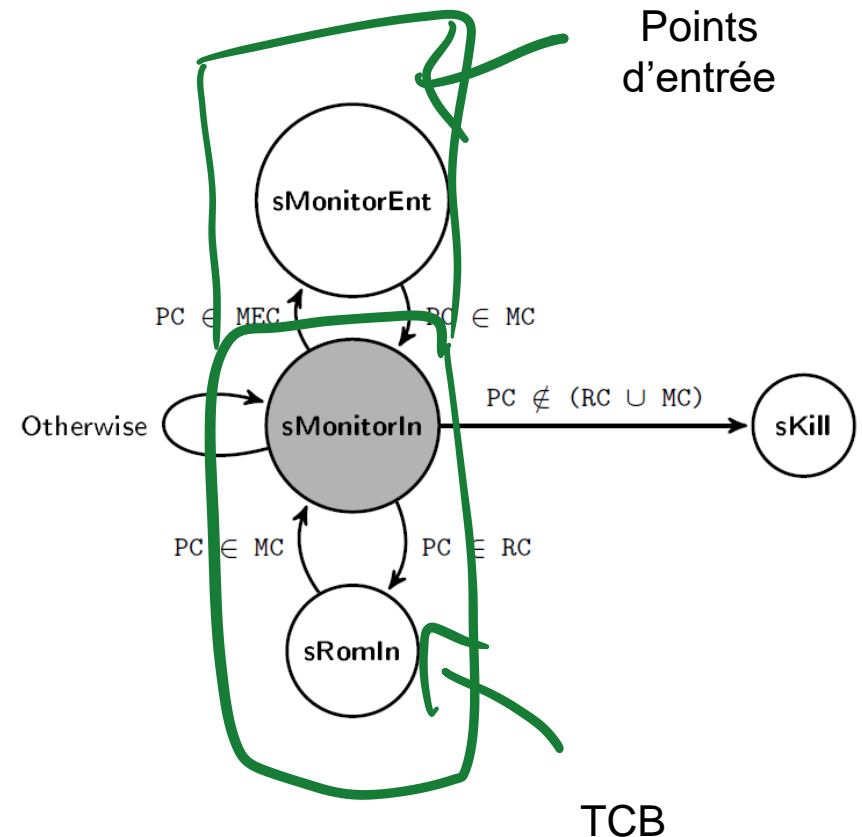
Exemple : Vérification des spécifications

- Le monitor fait partie du TCB et peut proposer différents services (ex. configuration de la MPU).



Exemple : Vérification des spécifications

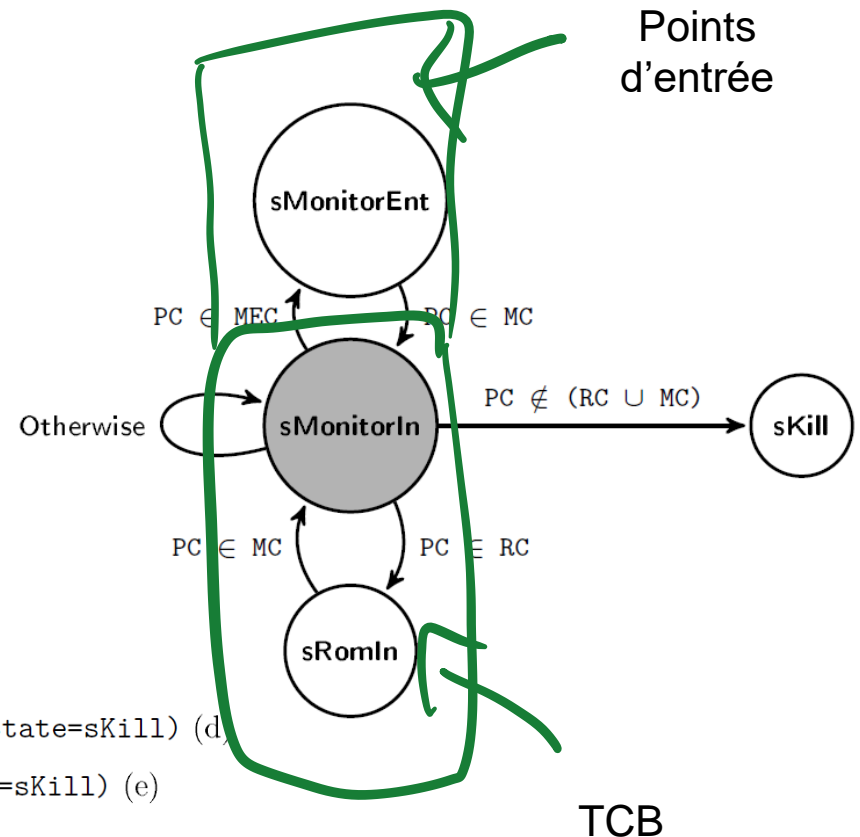
- Le monitor fait partie du TCB et peut proposer différents services (ex. configuration de la MPU).
- Le monitor ne doit pas être appelé directement en dehors du TCB.



Exemple : Vérification des spécifications

- Le monitor fait partie du TCB et peut proposer différents services (ex. configuration de la MPU).
- Le monitor ne doit pas être appelé directement en dehors du TCB.
- Traduction des propriétés de sécurité en spécification LTL.

LTLSPEC $G(\text{oldState}=\text{sMonitorIn}) \wedge (\text{PC} \notin (\text{RC} \cup \text{MC})) \rightarrow X(\text{newState}=\text{sKill})$ (d)
LTLSPEC $G(\text{oldState}=\text{sMonitorIn}) \wedge (\text{PC} \notin \text{MEC}) \rightarrow X(\text{newState}=\text{sKill})$ (e)



- **Conception et implémentation de deux modules matériels :**
 - L'EAP pour créer une forte séparation entre le TCB et le reste.
 - La MMP pour contrôler les accès mémoire des différents périphériques.

- **Développement d'un root-of-trust :**
 - La partie root du TCB logiciel permet de configurer l'EAP et de vérifier l'authenticité du monitor avec un algorithme HMAC.

- **Vérification formelle des propriétés de sécurité :**
 - Présentation d'une vérification formelle des propriétés de sécurité qui nous garantissent un bon niveau de sécurité.

Bilan

- **Articles** (Conférences et Workshops internationaux, et communications) :

(Conférence) A.Sensaoui, D.Hély, O.Aktouf. **Toubkal: A Flexible and Efficient Hardware Isolation Module for Secure Lightweight Devices.** In 15th European Dependable Computing Conference, Naples, 2019

(Workshop) A.Sensaoui, D.Hély, O.Aktouf. **Hardware-Based Isolation and Attestation Architecture for a RISC-V Core.** In SiFive Technical Symposium, Grenoble, 2019

(Workshop) A.Sensaoui, O.Aktouf, D.Hély. **SHCoT: Secure (and Verified) Hybrid Chain of Trust to protect from malicious software in lightweight devices.** In The 1st Annual International Workshop on Software Hardware Interaction Faults, co-located with ISSRE, Berlin, 2019

(Communication) A.Sensaoui, D.Hély, O.Aktouf. **Hardware-Based Isolation and Attestation Architecture for a RISC-V Core.** In CySep and EuroS&P, Stockholm, 2019 And In 15th European Dependable Computing Conference, Naples, 2019

- **Un article de revue est en cours de soumission au journal "IEEE Transactions on Computers"**

Agenda

- I. Prérequis
- II. Evaluation des solutions existantes
- III. Toubkal : Architecture hybride d'isolation et d'attestation
- IV. Perspectives et Conclusion**

Perspectives

- **Gestion sécurisée des interruptions :**
 - Introduction des interruptions à Toubkal et contrôler leurs points d'entrée.

- **Prendre en considération d'autres chemins d'attaques :**
 - Que faut il prendre en considération pour contourner les attaques matérielles?

- **Vérification hybride :**
 - Être capable de vérifier l'interaction logicielle matérielle

- **Développements d'outils qui vont permettre aux développeurs d'automatiser l'intégration de Toubkal.**

- **2 contributions majeures :**

- Une étude expérimentale de l'existant.
- Implémentation de deux briques matérielles en Chisel3/scala et développement d'une partie logicielle en C/Assembleur.
Utilisation d'outils de vérification formelle pour vérifier les propriétés de sécurité de Toubkal.

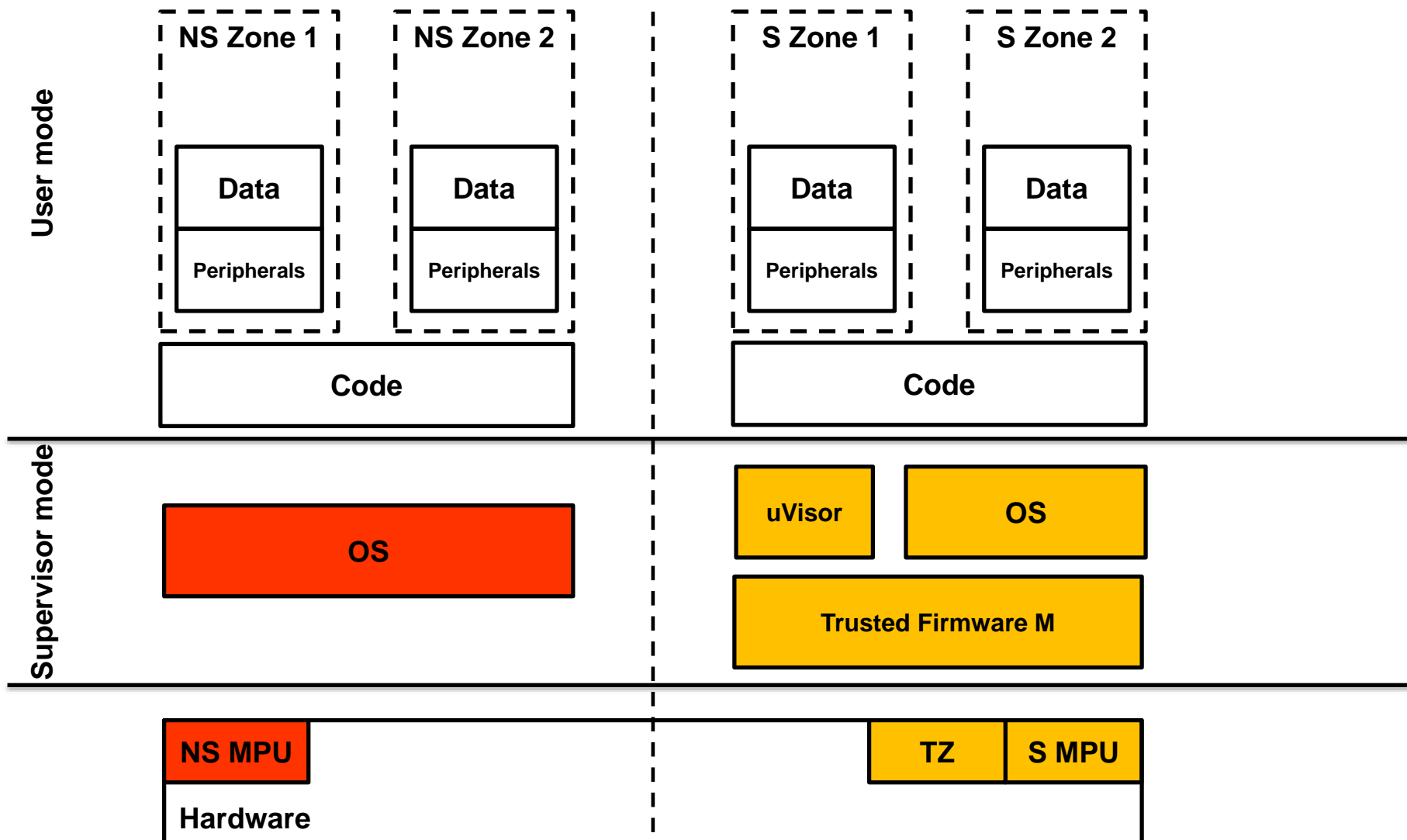
- **Publications :**

- Un article de revue publié, et un autre en cours de soumission.
- Une conférence et deux workshops internationaux.
- Deux communications dans des journées scientifiques.

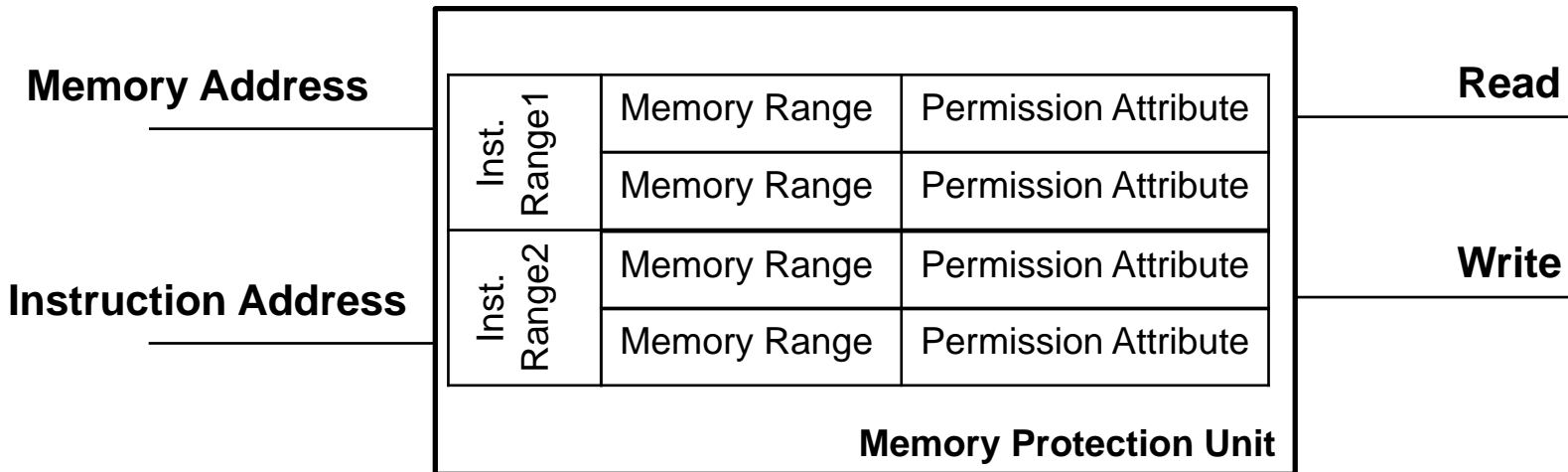
Merci

Q & R

Arm TrustZone for Armv8-M

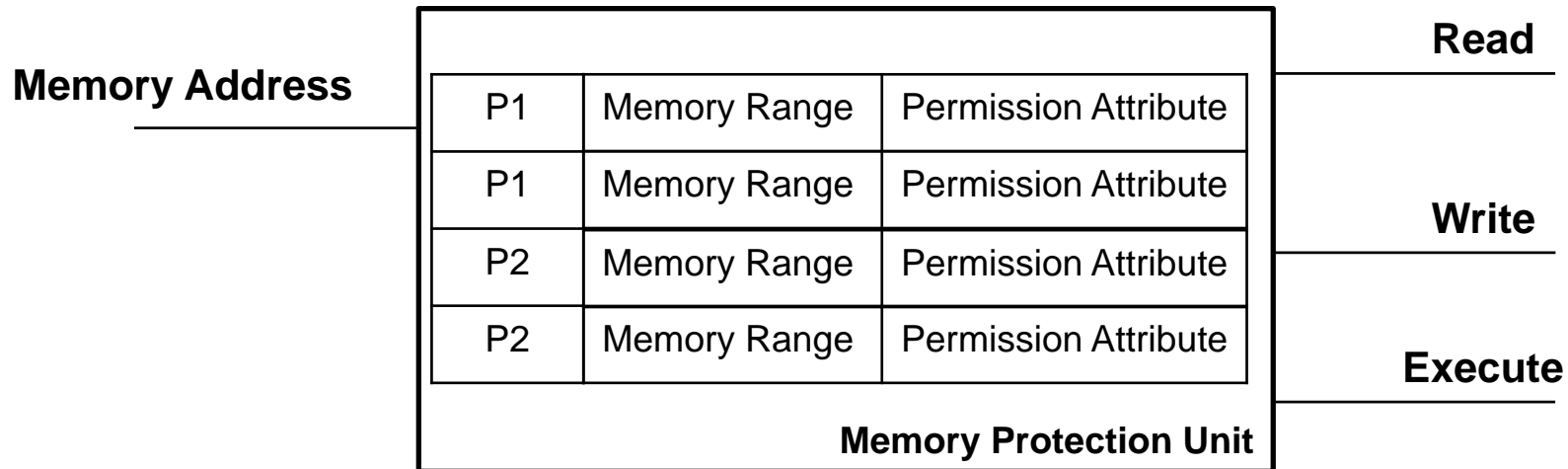


Memory Protection Units



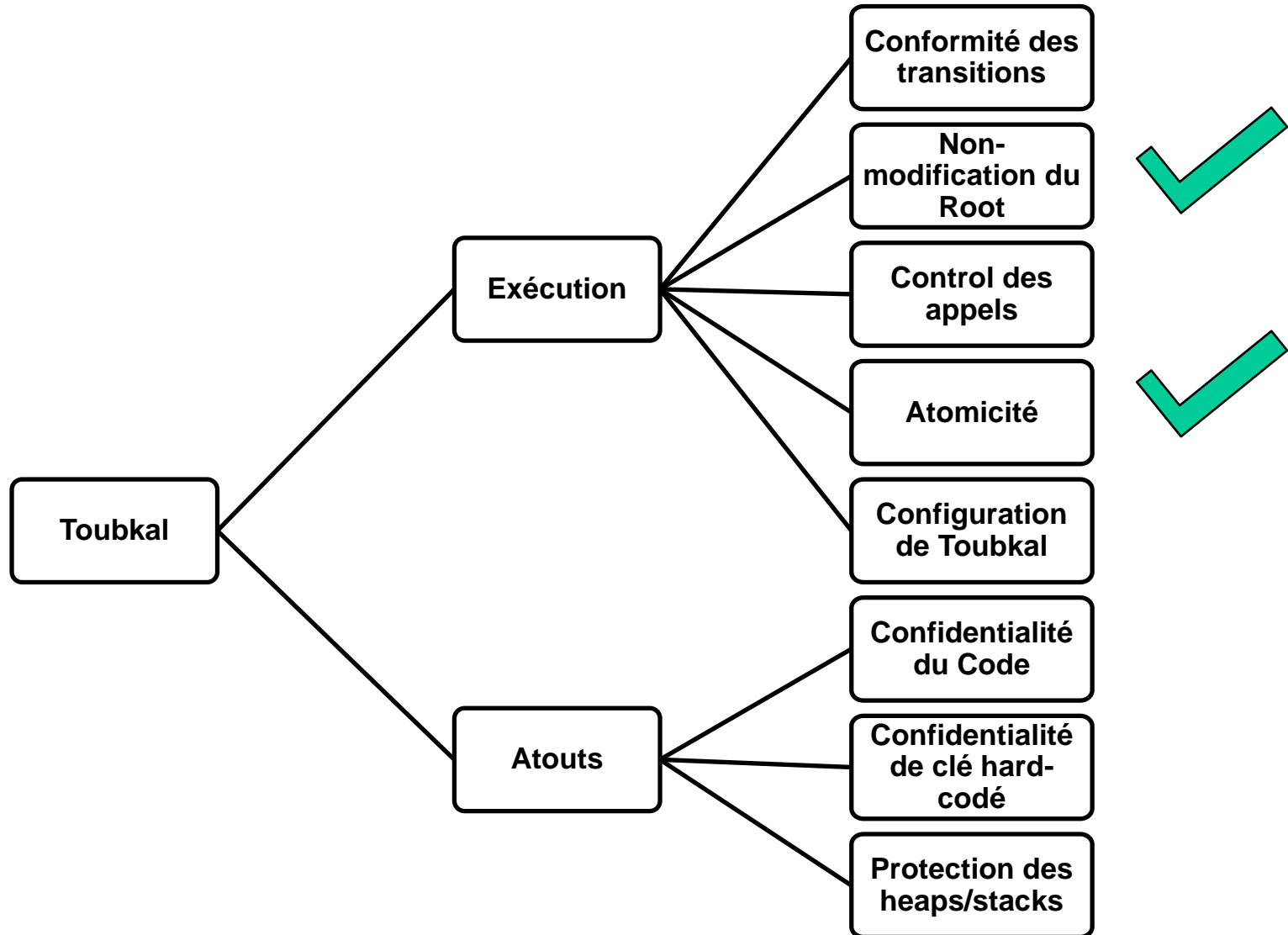
Examples: Sancus, Intel Execution Aware-MPU (EA-MPU)

Protection de la Mémoire

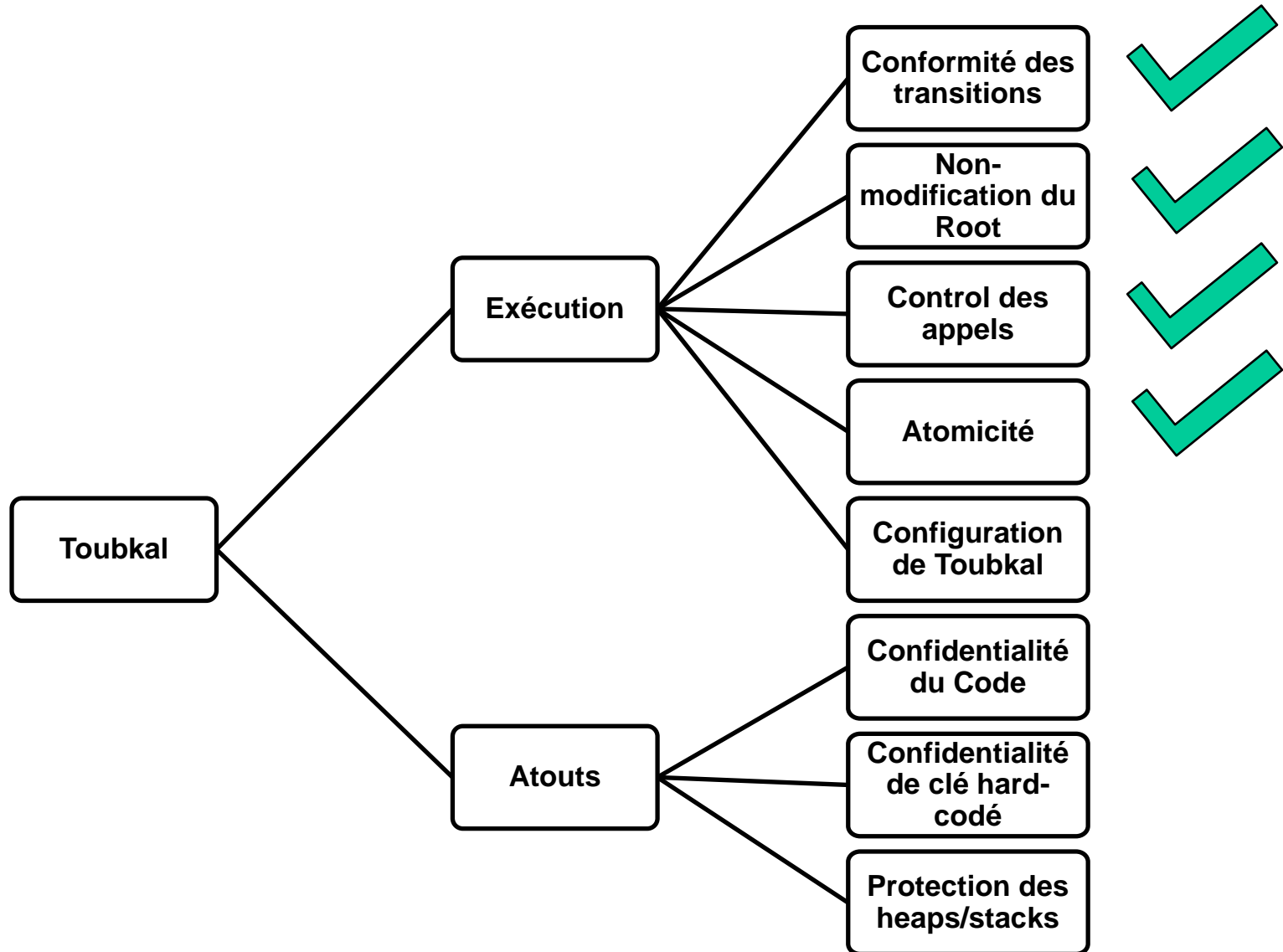


Examples: Mondrian

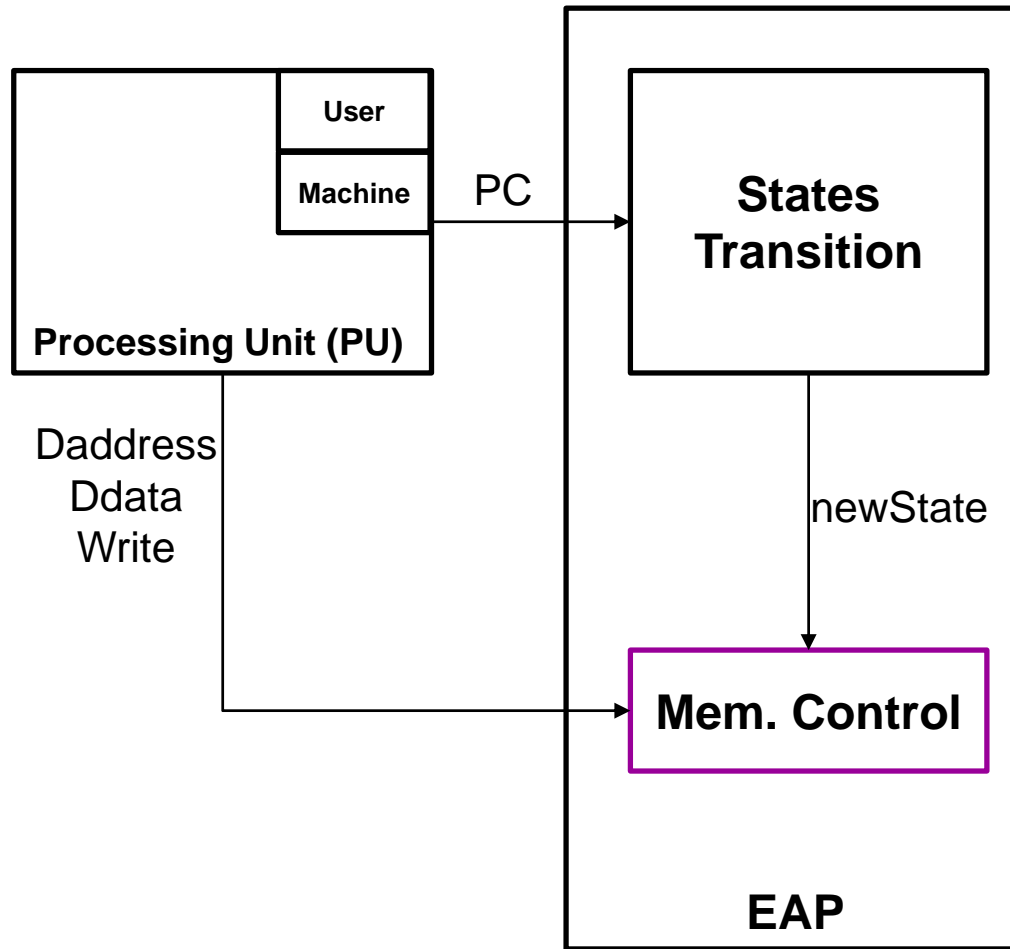
Propriétés de sécurité



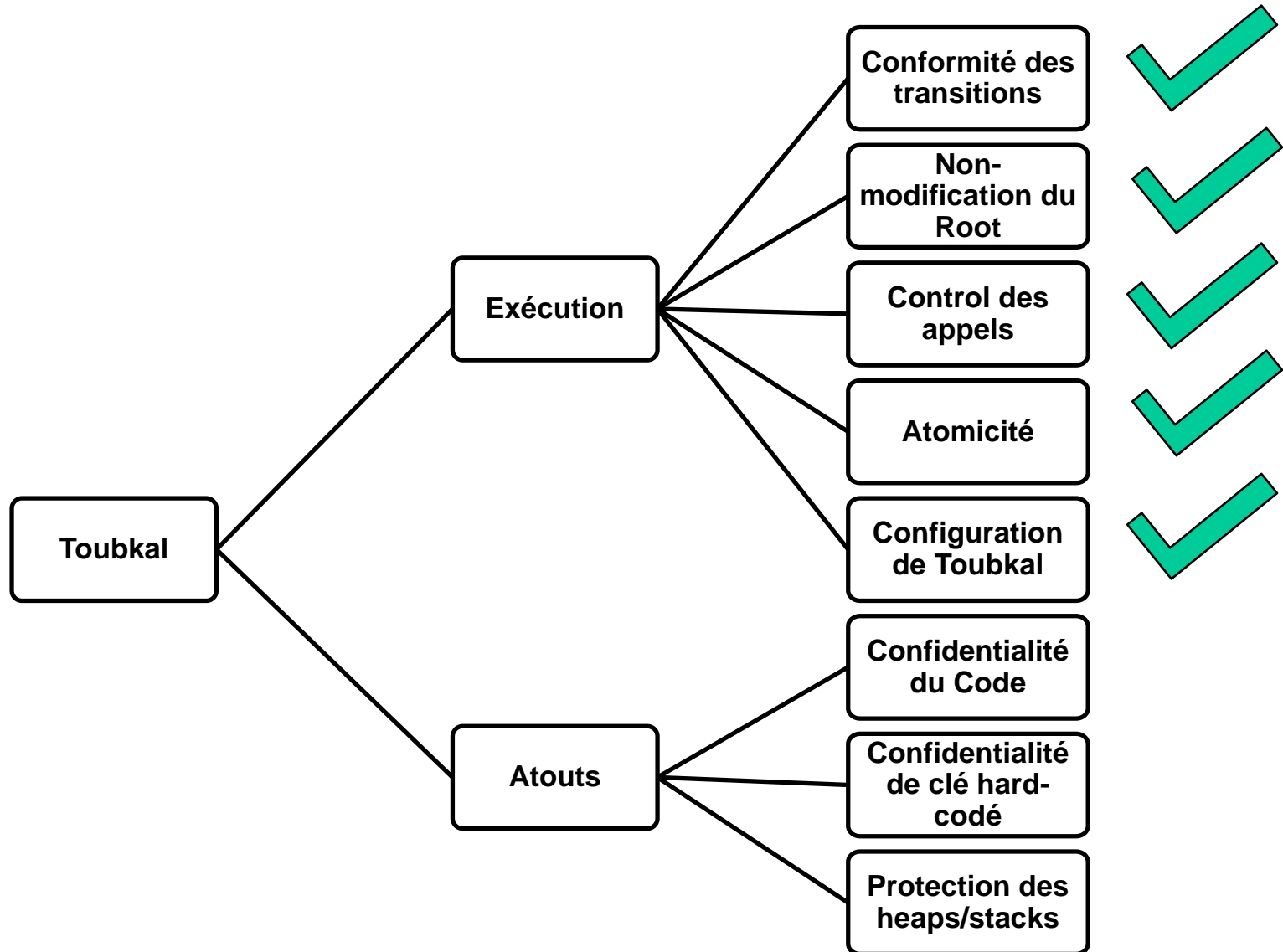
Propriétés de sécurité



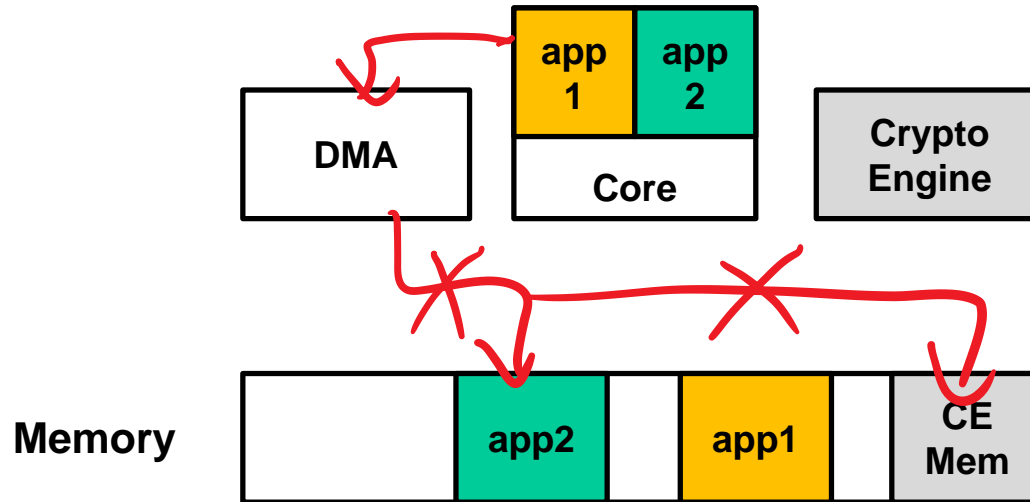
Propriétés de sécurité



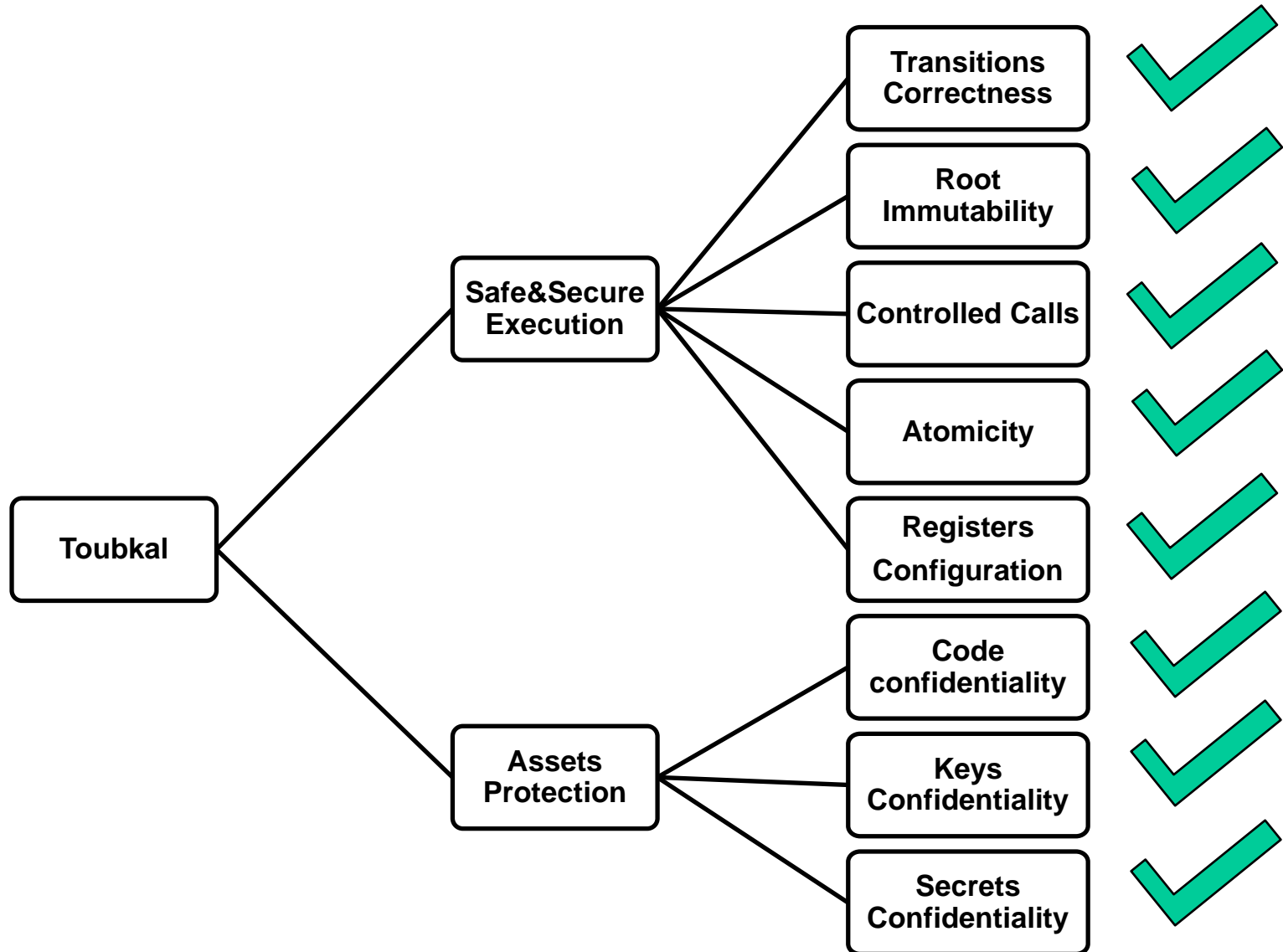
Propriétés de sécurité



Propriétés de sécurité



Propriétés de sécurité



Proving the correctness of these security properties

Notation	Description
PC	Program Counter
RC	Root Code
REC	Root Entry Code
MC	Monitor Code
MEC	Monitor Entry Code
DZC	D-zones Code
DZE	D-zones Entry Code
SMM	Security Monitor Data Memory
DZM	D-zones Data Memory
KM	Memory area for key storage
sRomEnt	State when PC is in the Root code entry point
sRomIn	State when PC is legitimately inside the root
sMonitorEnt	State when PC is in the monitor entry points
sMonitorIn	State when PC is legitimately inside the monitor
sDzoneEnt	State when PC is in the D-zones entry points
sDzoneIn	State when PC is legitimately inside the D-zones
sNone	State when PC is in the rest of code
sKill	State when there is a non-authorized access
oldState	The state just before we check the new PC
newState	The state after checking the transition and memory access
Daddr	Data address for memory access

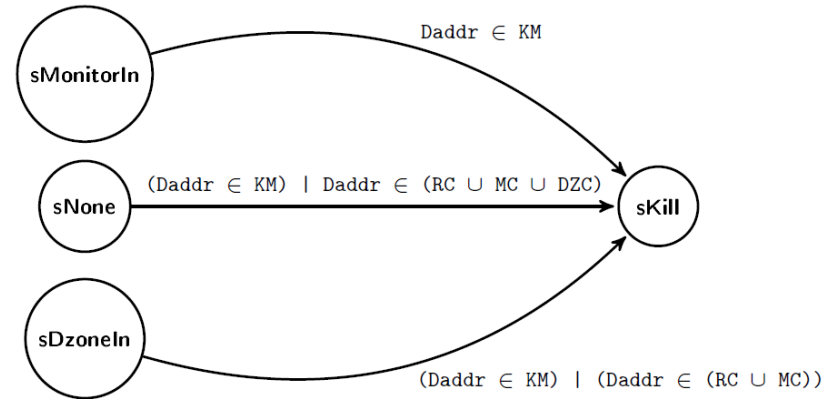
Proving the correctness of these security properties

▪ Goal.

Prove that the hard-coded key is only accessible from the root.

▪ Linear Temporal Logic (LTL).

- **Future "F"**: $F p$ is true if p is true at a future state.
- **Global "G"**: $G p$ is true if p is always true.
- **next "X"**: $X p$ is true if p is true at the next state.
- **Until "U"**: $p U q$ is true if q is true and p was true at all the state before.



▪ Translate into LTL SPEC.

`LTLSPEC G ((newState!=sRomIn)&(Daddr∈MK) -> X reset=true) (f)`