Attaques par canaux auxiliaires

Panel des attaques physiques

Benoît Gérard
benoit.gerard@irisa.fr
30 septembre 2019

Attaques par canaux auxiliaires

Canal auxiliaire

Canal d'information *auxiliaire* qui contrairement au canal *logique* (entrées/sorties) n'a pas vocation à transporter de l'information.

En général il s'agit d'un canal "physique" (temps, température, consommation électrique ...).

Accès physique local

- Cibles : systèmes embarqués.
- Menace tous les systèmes accessibles physiquement.
- Cours : Benoît

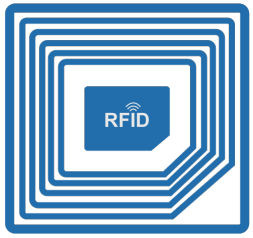
Accès distant (ou logiciel local)

- Cibles : machines non-isolées.
- Menace les systèmes autorisant les logiciels tiers.
- Cours : Clémentine

Sommaire de la session

Les systèmes embarqués

Des applications et des contextes variés

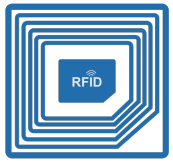


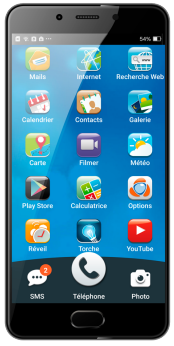
Les systèmes embarqués

Des applications et des contextes variés

Les systèmes embarqués

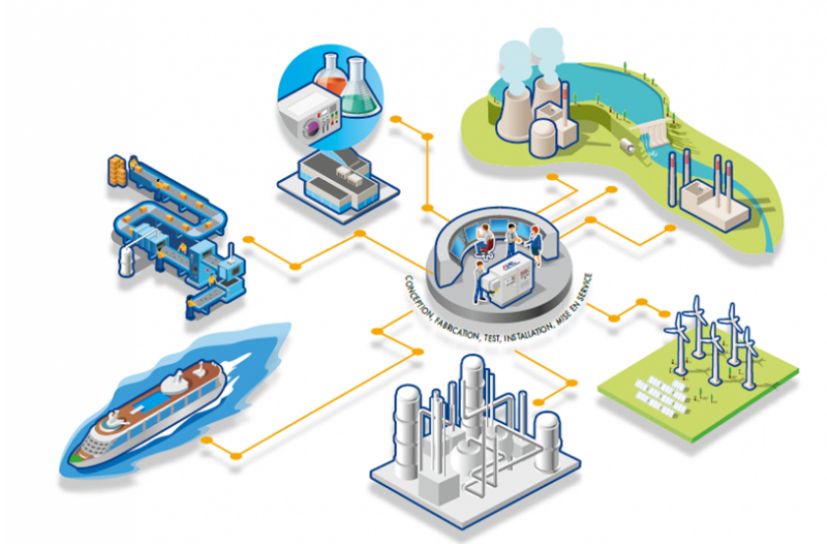
Des applications et des contextes variés

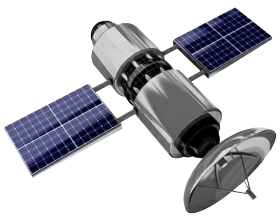




Les systèmes embarqués

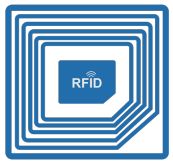
Des applications et des contextes variés

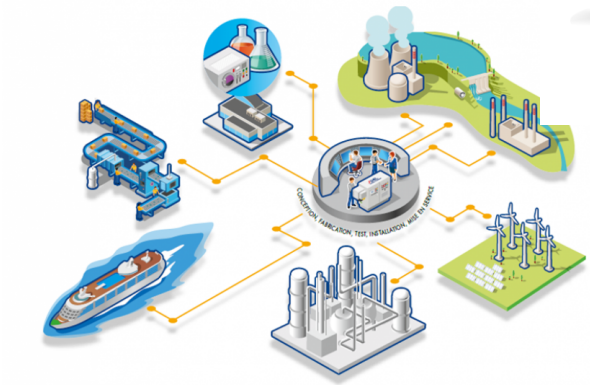




Les systèmes embarqués

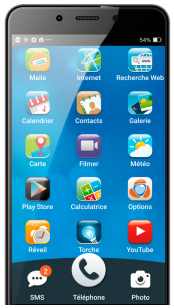
Des applications et des contextes variés

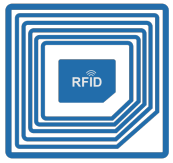


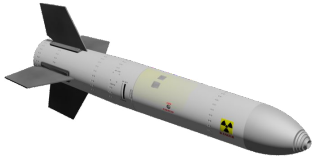


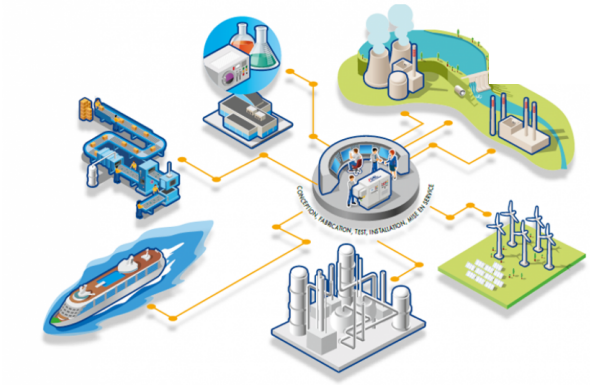
Les systèmes embarqués

Des applications et des contextes variés



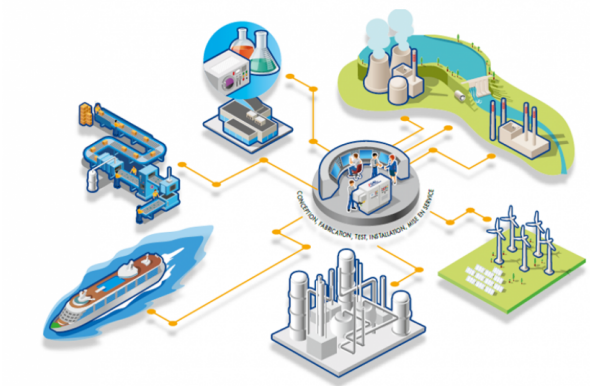






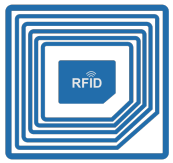
Les systèmes embarqués

Des applications et des contextes variés



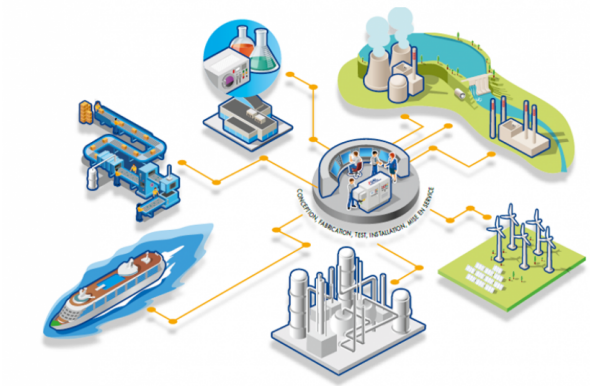
Les systèmes embarqués

Des applications et des contextes variés



système embarqué

système électronique et informatique autonome et spécialisé



Différents attaquants pour différentes menaces.

Attaquants

- Délinquants
  - un geek, un garage et un PC de gamer.
- Mafias
  - une équipe complète et un réseau de PC zombies.
- États
  - d'un groupe de hacker à la NSA.

Menaces

- atteinte à la vie privée,
- espionnage,
- vol/fraude,
- rançonnage,
- traçage,
- panne générale,
- accidents,
- actes de guerre.

Quelques spécificités rencontrées dans le monde embarqué.

Ressources limitées

- puissance limitée,
- faible autonomie (énergie).

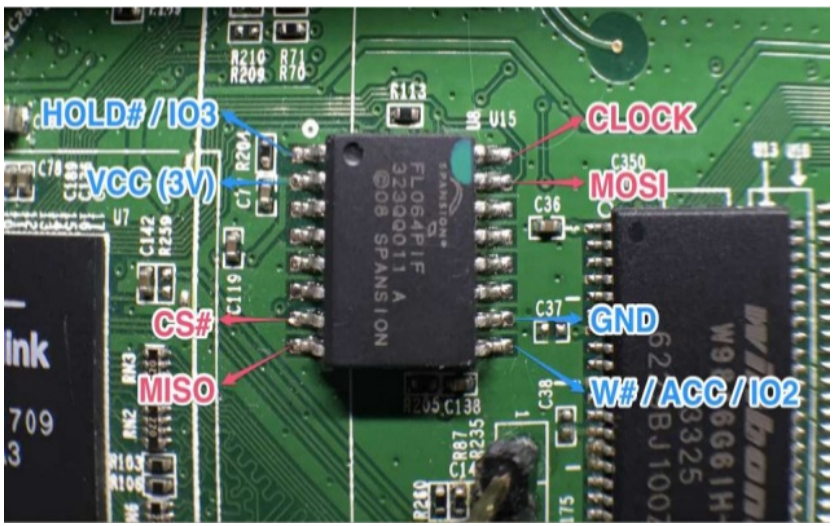
Contraintes

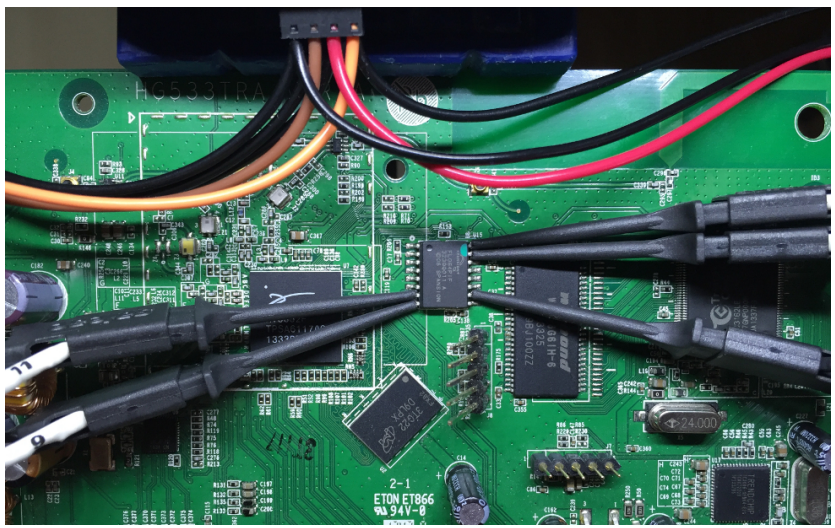
- surface limitée,
- contrainte temps réel,
- conditions environnementales peu clémentes (température, rayons ionisants).

Contexte d'emploi

- embarqué donc accessible à l'attaquant !

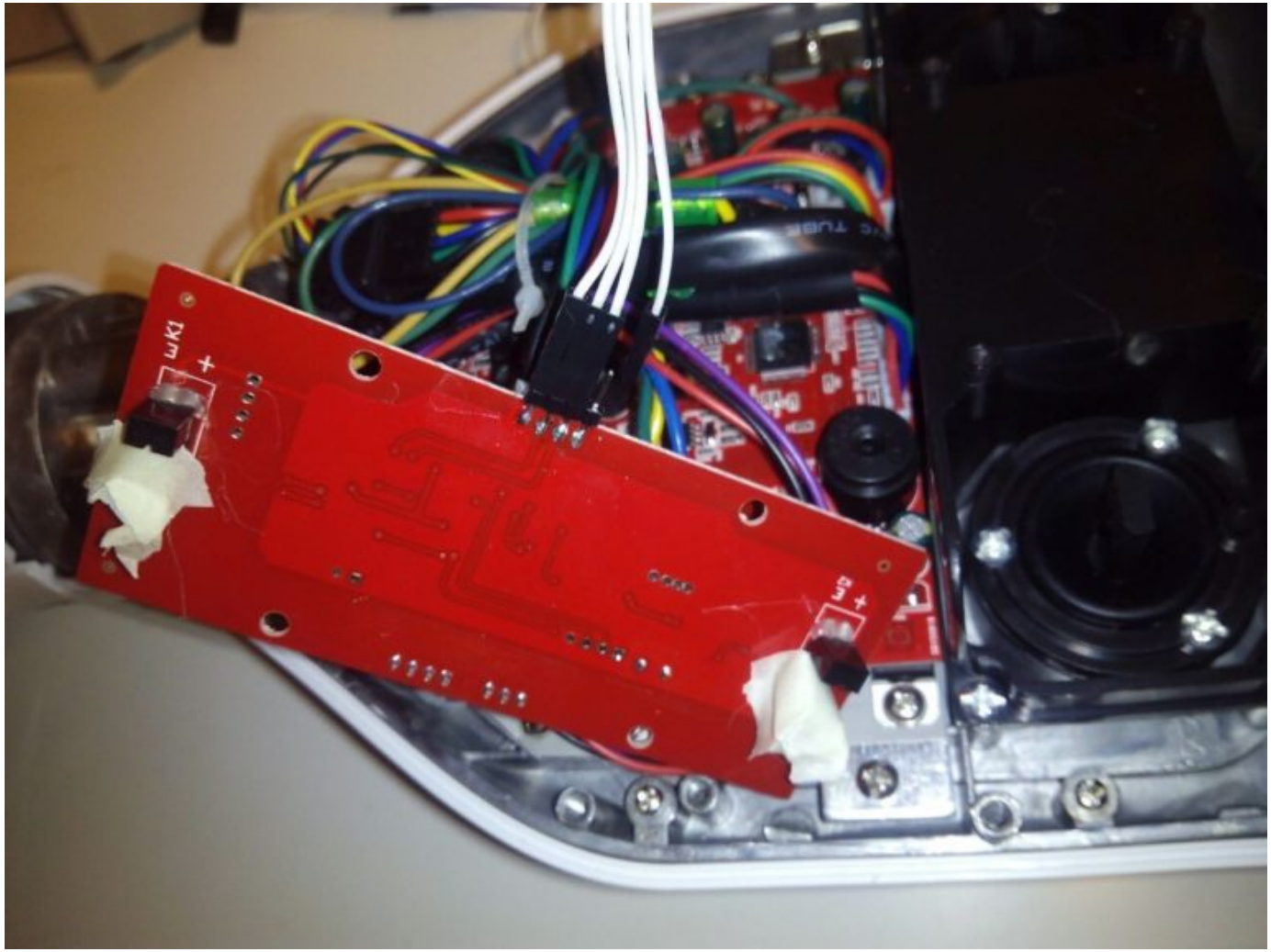
Lecture du firmware d'un routeur UPC



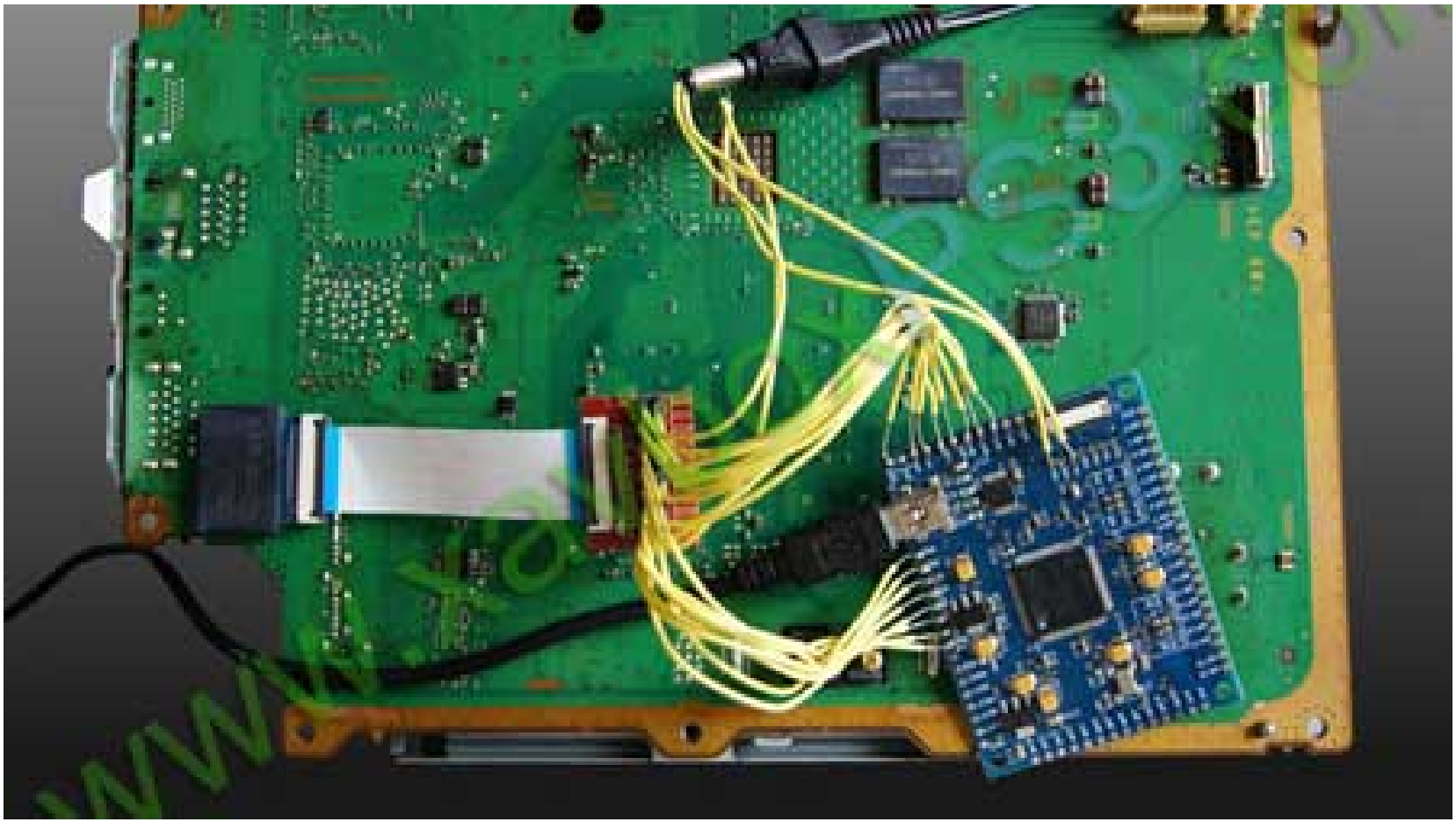


Puis rétro-ingénierie avec des outils classiques type IDA.

Lecture du firmware d'un overboard



Modification du firmware en mémoire d'une PS3



Observation

- Il est facile de lire toute mémoire externe à un composant.
- Des outils puissants existent pour appréhender un code binaire.

Il "suffit" alors de

1. lire le code,
2. le comprendre,
3. le modifier pour supprimer la sécurité
4. le re-charger.

Conclusion

La sécurité d'un produit ne peut donc pas uniquement se reposer sur des tests logiciels !

Il existe de nombreuses techniques complémentaires :

- manque de documentation,
- obfuscation de code,
- brouillage de mémoire (mélange des bits et/ou des adresses),
- enfouissement de pistes sur le PCB,
- utilisation de protocoles "maison" plutôt que de standards,
- ...

Permettent de ralentir les attaquants mais ne suffit pas en général.

Une autre piste est l'utilisation de **cryptographie** !

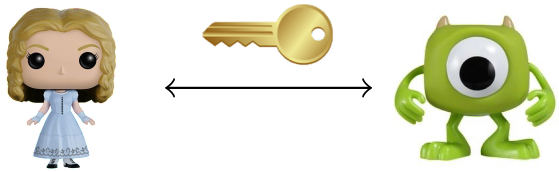
Les trois principaux services fournis par la cryptographie sont :

- la *confidentialité*,
- l'*intégrité*,
- l'*authenticité*.

La confidentialité permet de rendre bénin la lecture d'un bus ou d'une mémoire.

L'authenticité et l'intégrité permettent d'éviter les altérations d'un code critique.

Cryptographie symétrique



Cryptographie symétrique

Cryptographie symétrique

Cryptographie asymétrique

Cryptographie symétrique

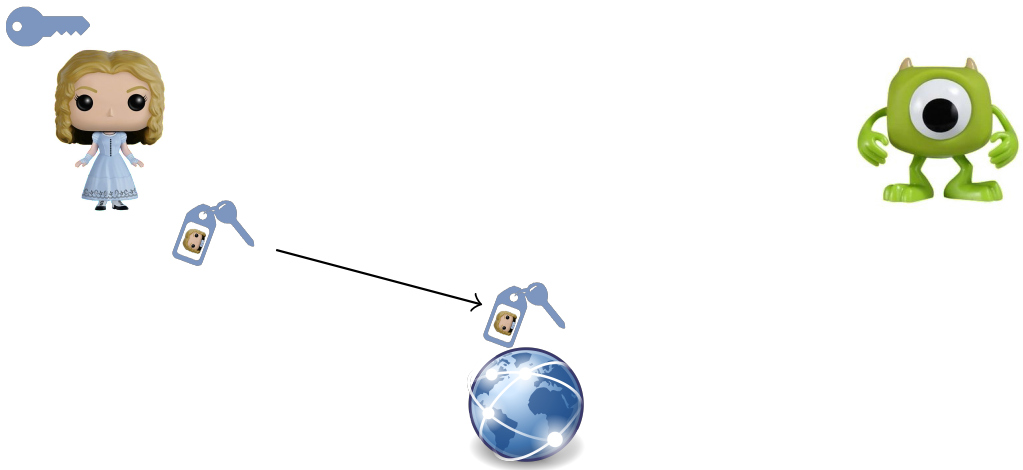
Cryptographie asymétrique

Cryptographie symétrique

Cryptographie asymétrique

Cryptographie symétrique

Cryptographie asymétrique



Cryptographie symétrique

Cryptographie asymétrique

Cryptographie symétrique

Cryptographie asymétrique

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).

MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).

MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).

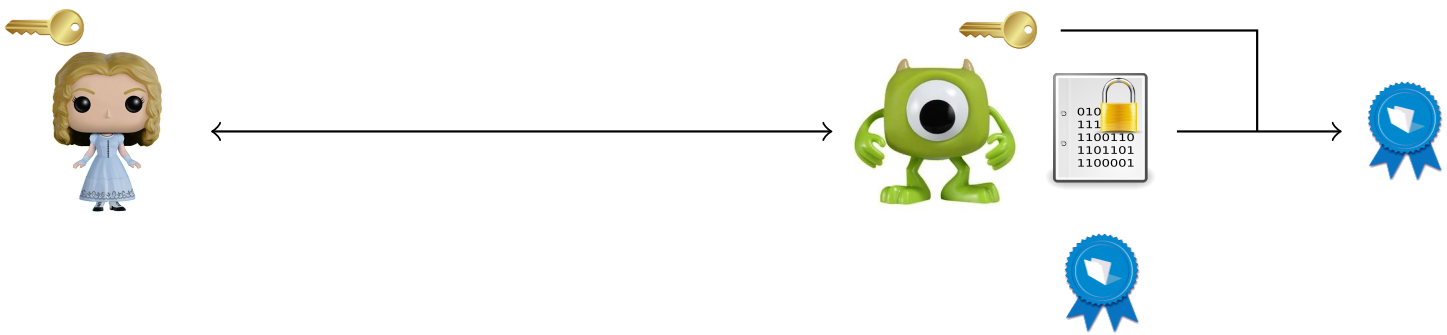
MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).



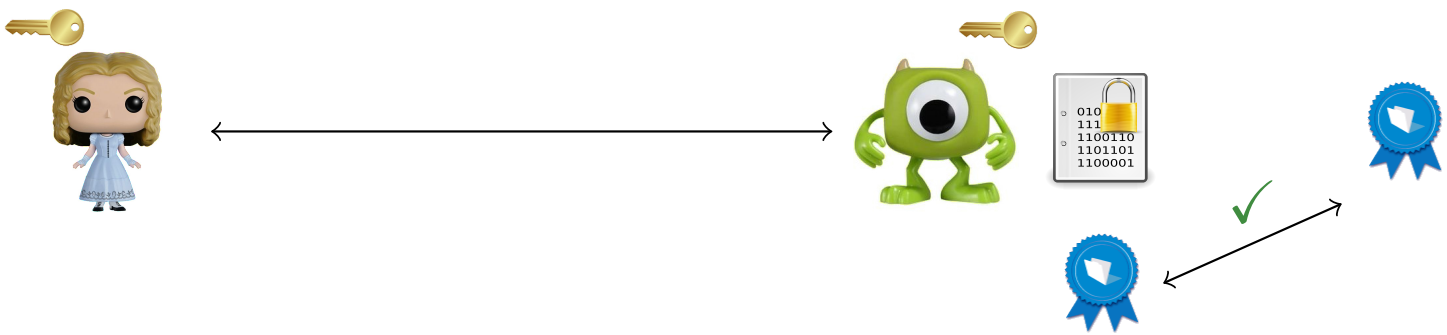
MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).



MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Chiffrement et MAC

Objectifs

- s'assurer de la *confidentialité* d'un contenu,
- s'assurer de l'*intégrité* d'un contenu (contenu non modifié par quelqu'un ne connaissant pas la clef).

MAC = Message Authentication Code

Chiffrement + MAC = chiffrement authentifié.

Signature

Objectifs

- s'assurer de l'*intégrité* d'un contenu,
- s'assurer de l'*authenticité* (preuve de l'origine du contenu).

Signature

Objectifs

- s'assurer de l'*intégrité* d'un contenu,
- s'assurer de l'*authenticité* (preuve de l'origine du contenu).

Signature

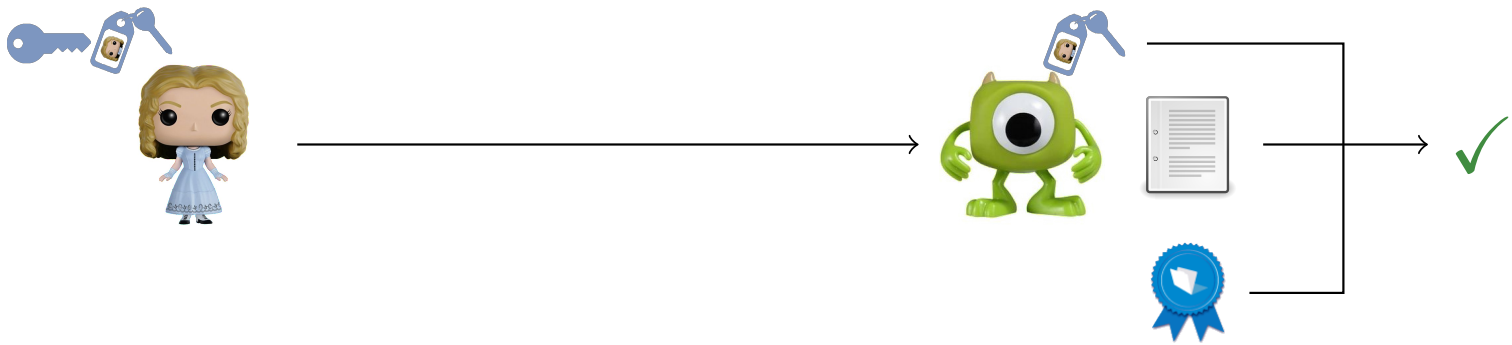
Objectifs

- s'assurer de l'*intégrité* d'un contenu,
- s'assurer de l'*authenticité* (preuve de l'origine du contenu).

Signature

Objectifs

- s'assurer de l'*intégrité* d'un contenu,
- s'assurer de l'*authenticité* (preuve de l'origine du contenu).



Utiliser la cryptographie symétrique implique une clef par correspondant . . .

Chiffrement asymétrique

- utilise le même principe de paire de clefs que la signature,
- le secret est du côté de la personne qui déchiffre,
- les opérations sont coûteuses.

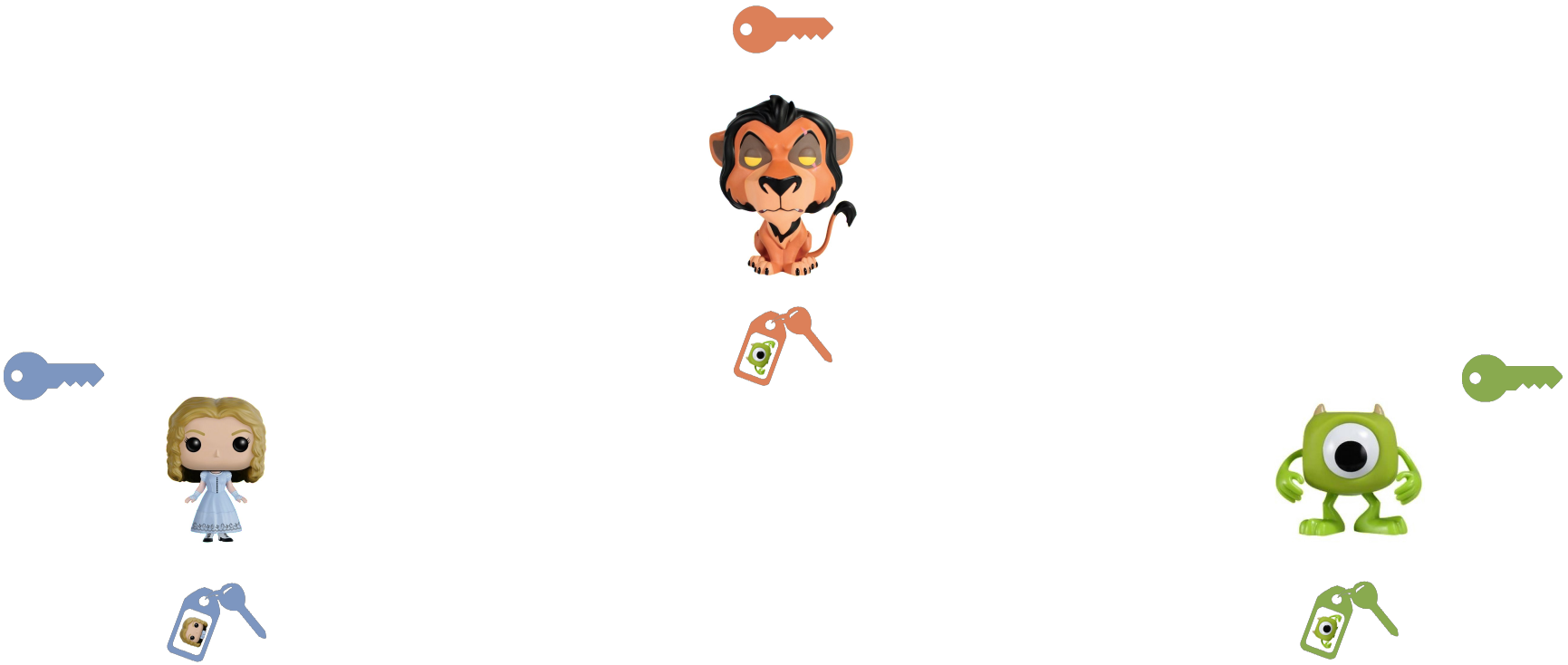
Échange de clef

- établit un secret partagé en se basant sur des opérations coûteuses,
- pour ensuite utiliser la cryptographie symétrique.

Man in the Middle

Man in the Middle

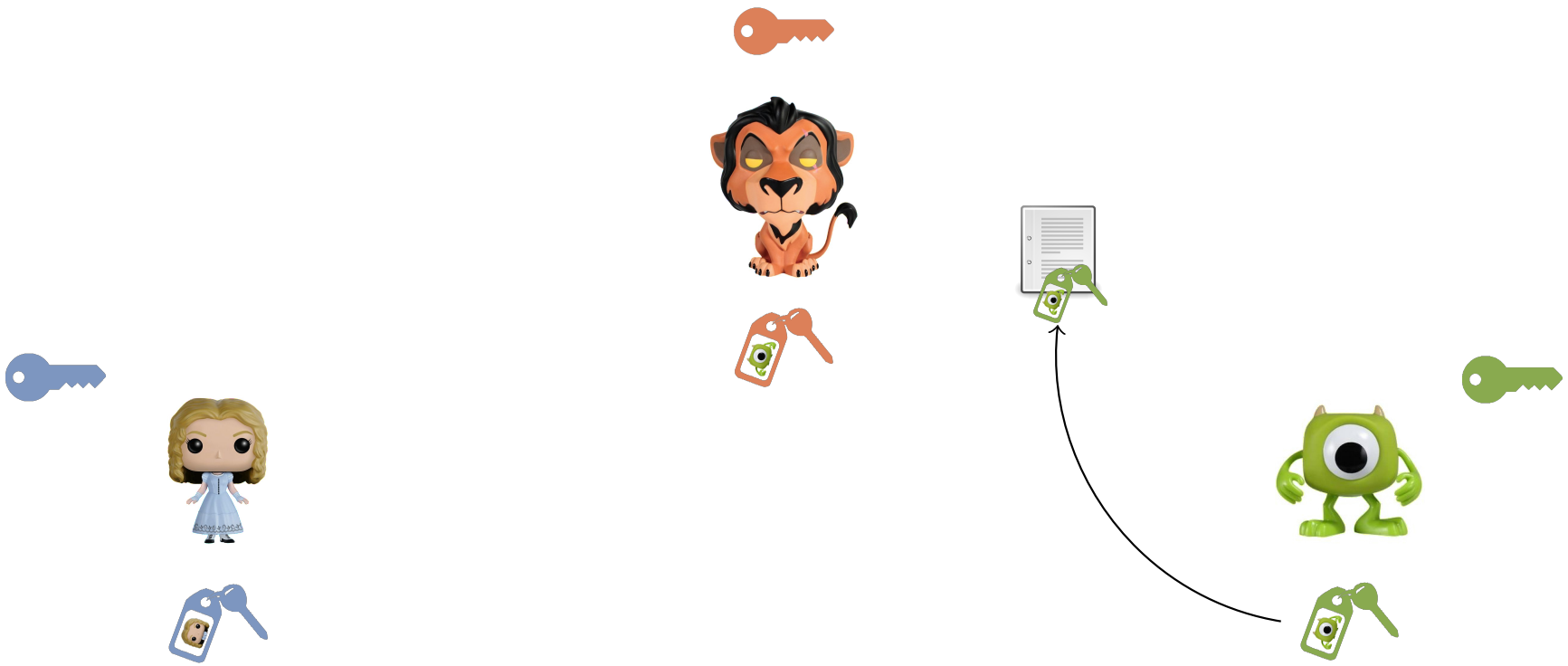
Man in the Middle



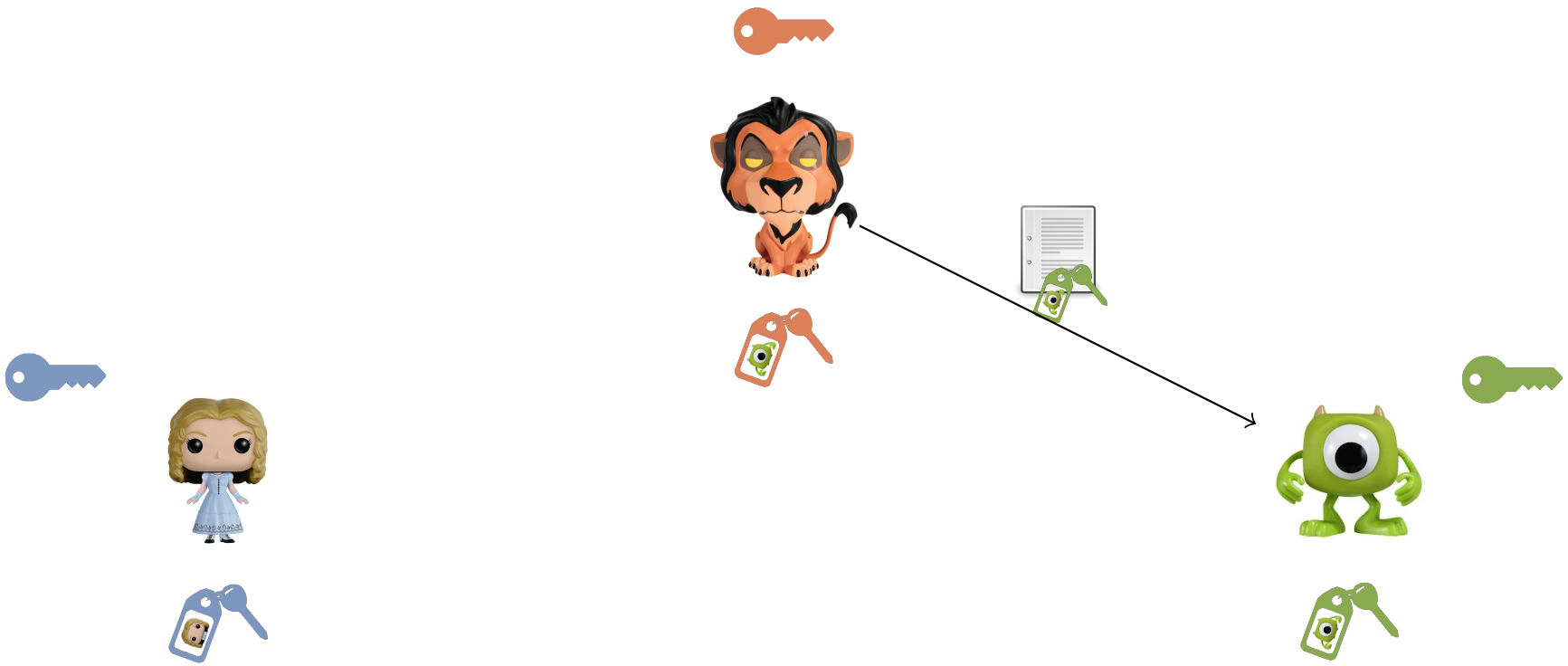
Man in the Middle

Man in the Middle

Man in the Middle



Man in the Middle



Certificats

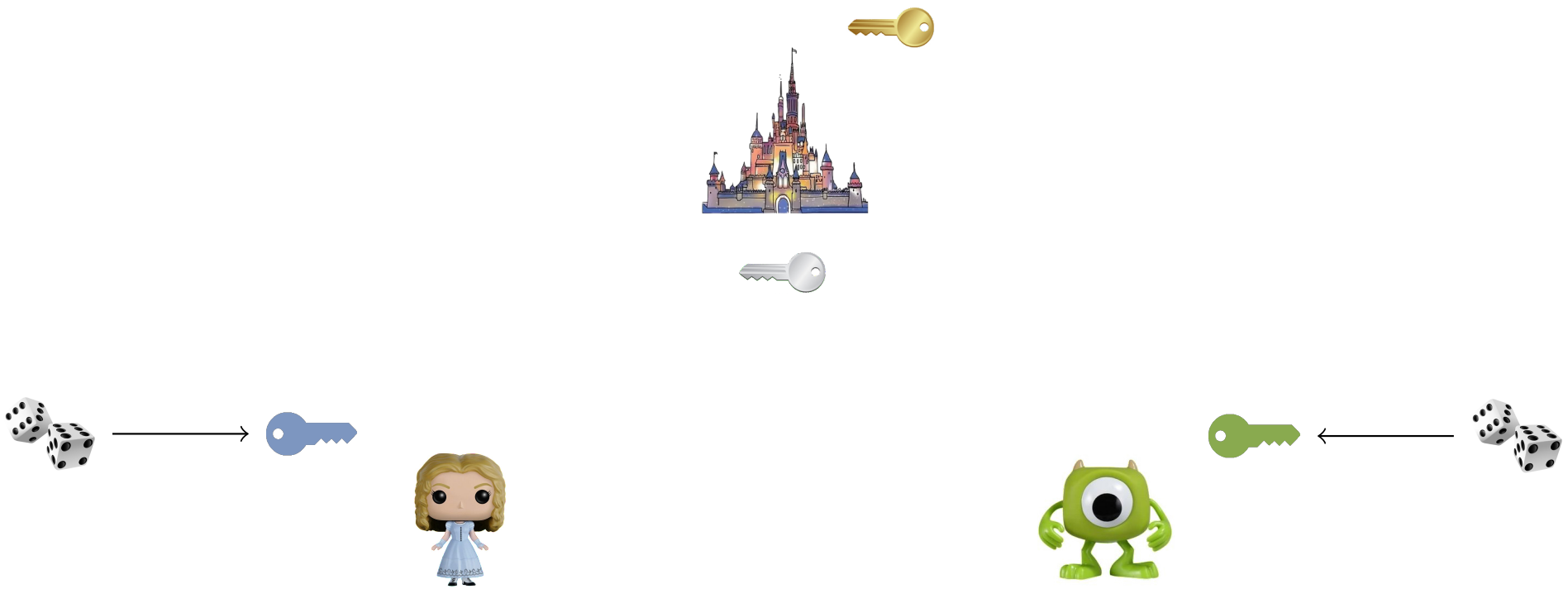
Objectif

- s'assurer de l'*identité* de son correpsondant.

Certificats

Objectif

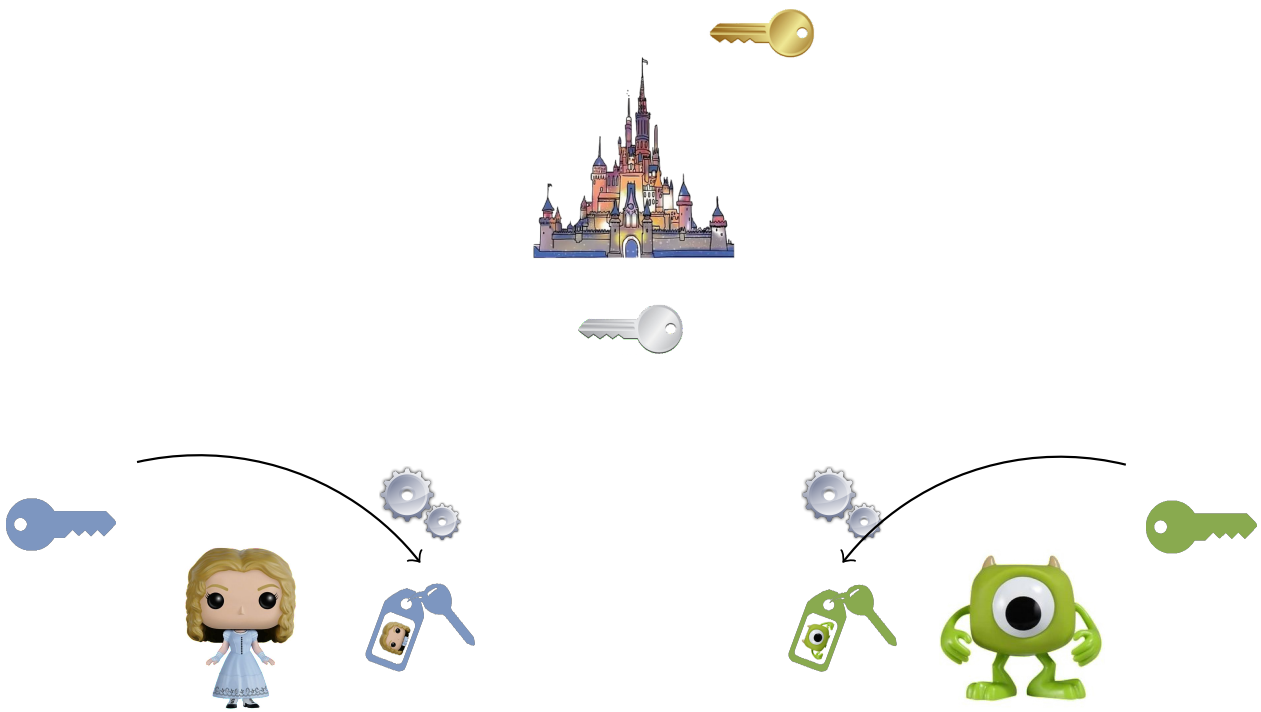
- s'assurer de l'*identité* de son correpsondant.



Certificats

Objectif

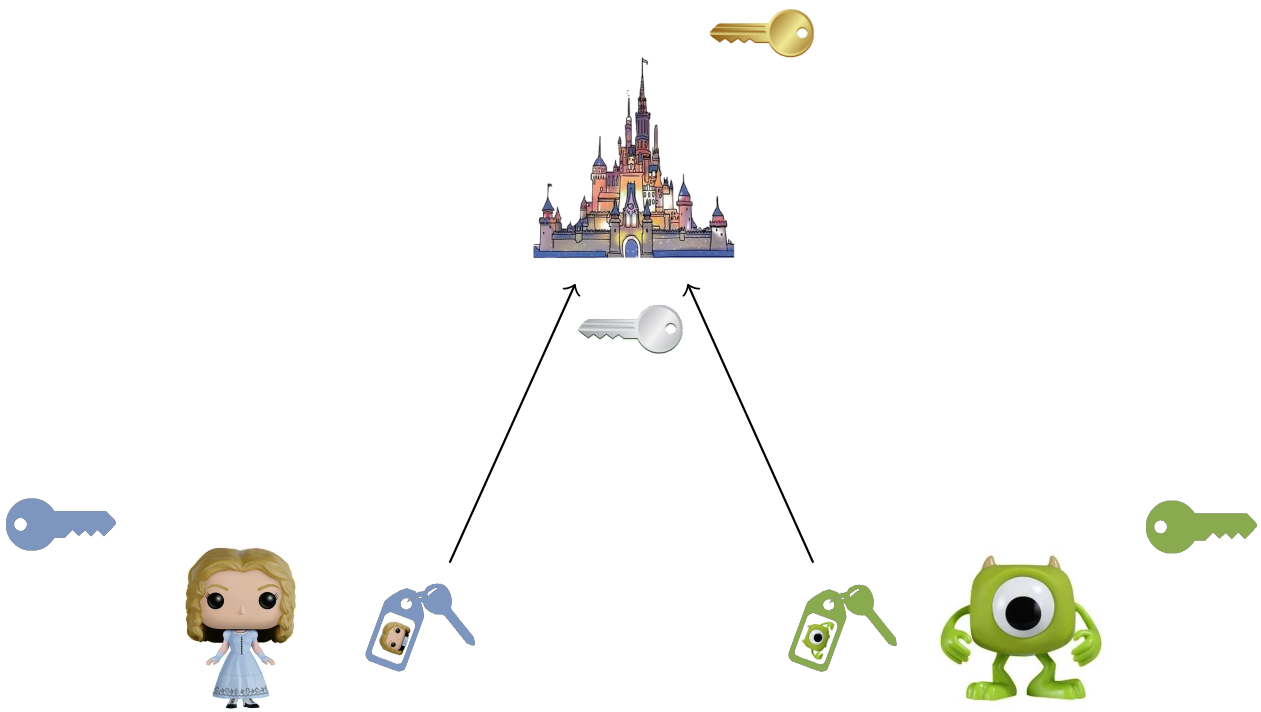
- s'assurer de l'*identité* de son correpsondant.



Certificats

Objectif

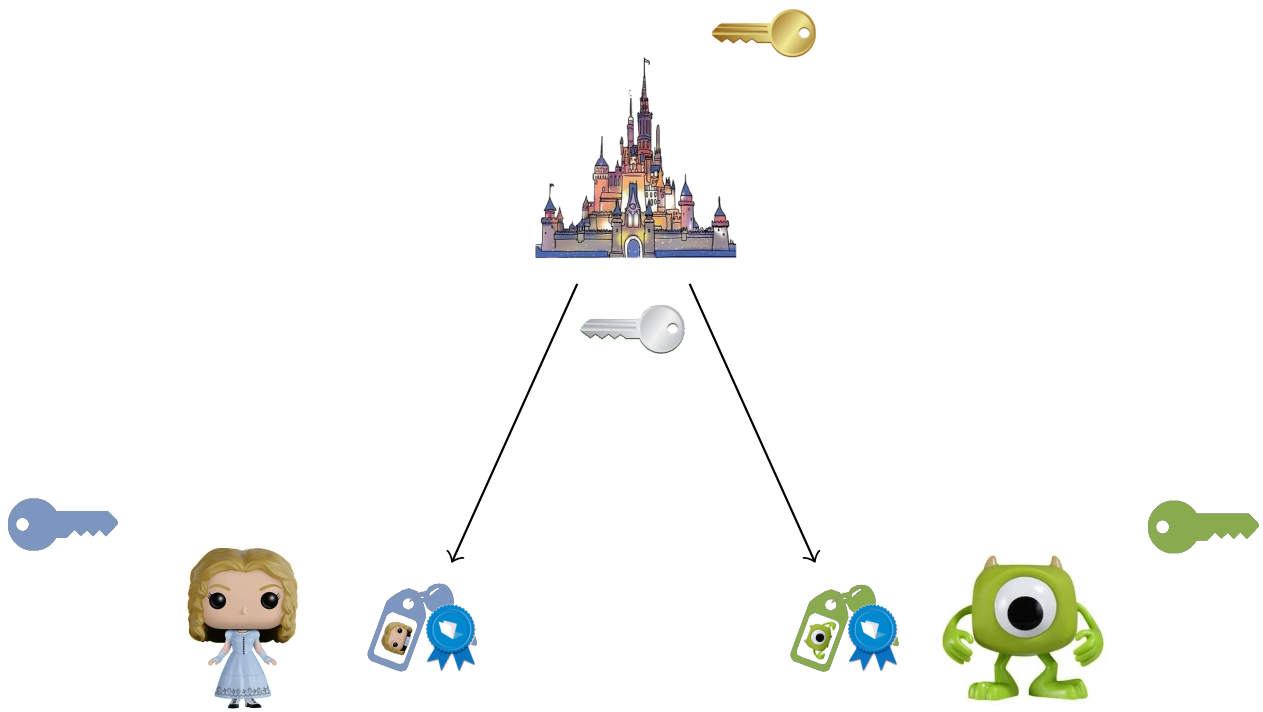
- s'assurer de l'*identité* de son correpsondant.



Certificats

Objectif

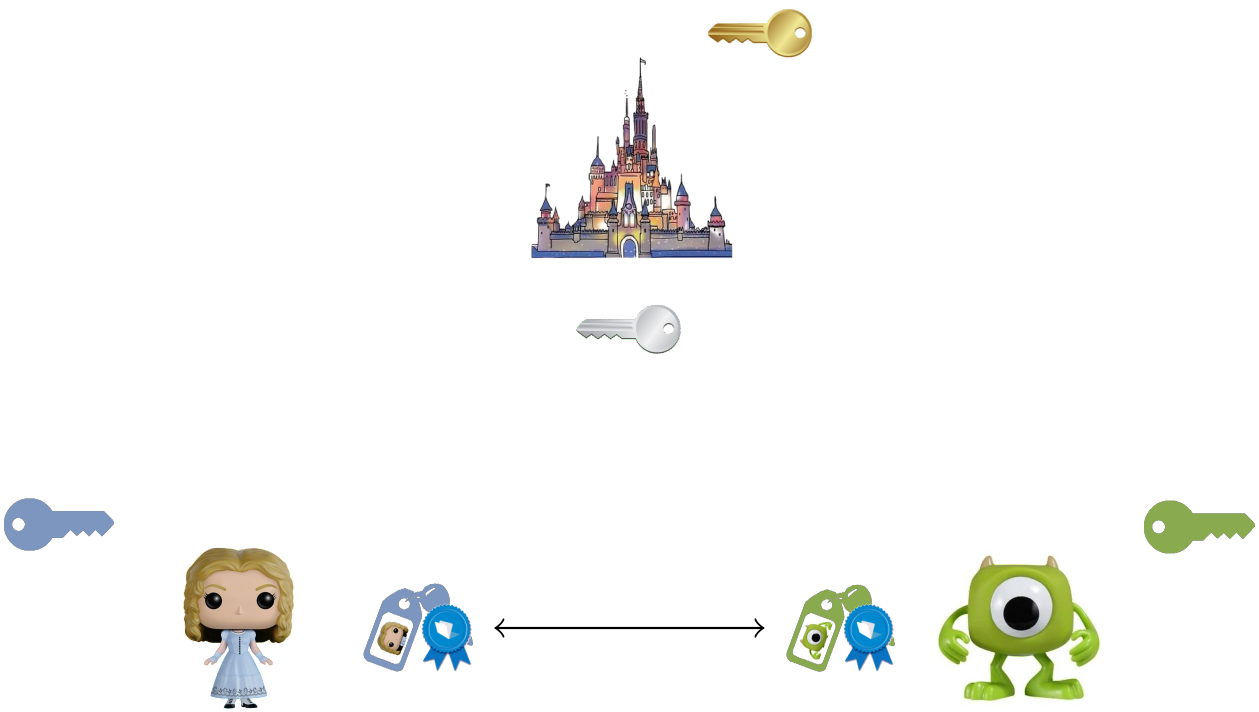
- s'assurer de l'*identité* de son correpsondant.

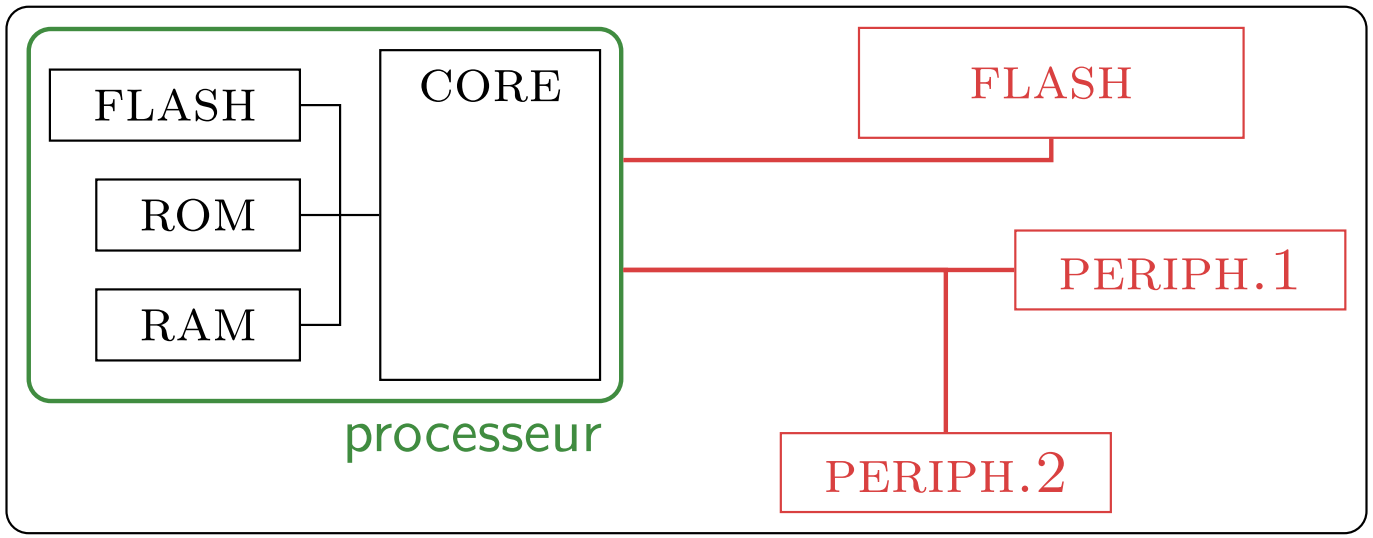


Certificats

Objectif

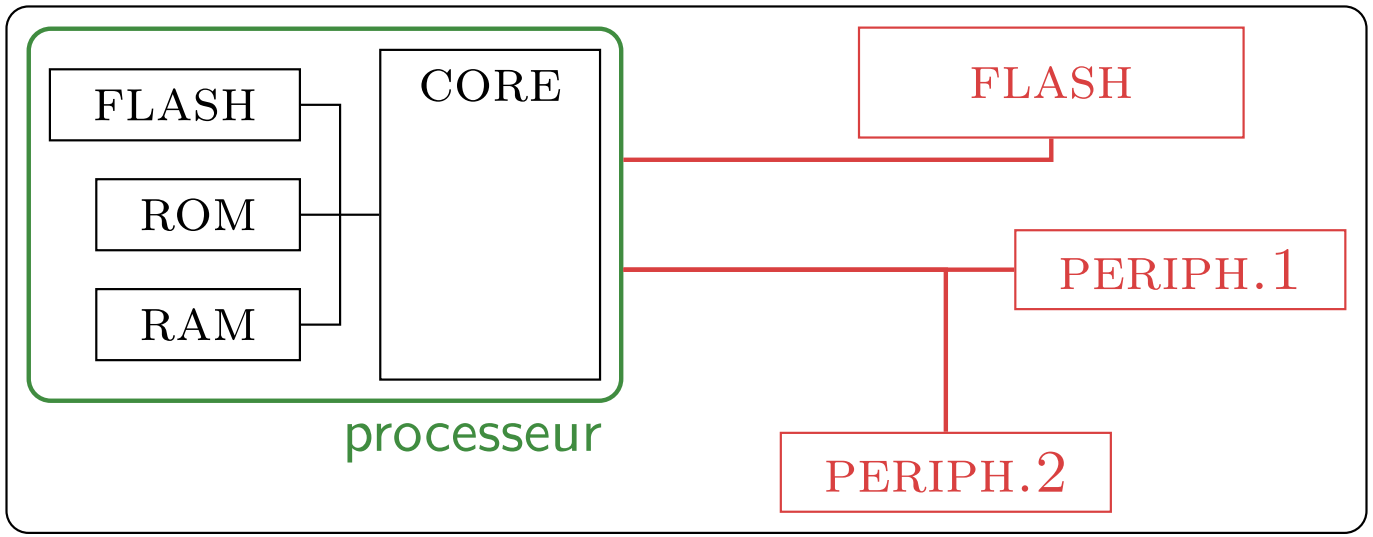
- s'assurer de l'*identité* de son correpsondant.

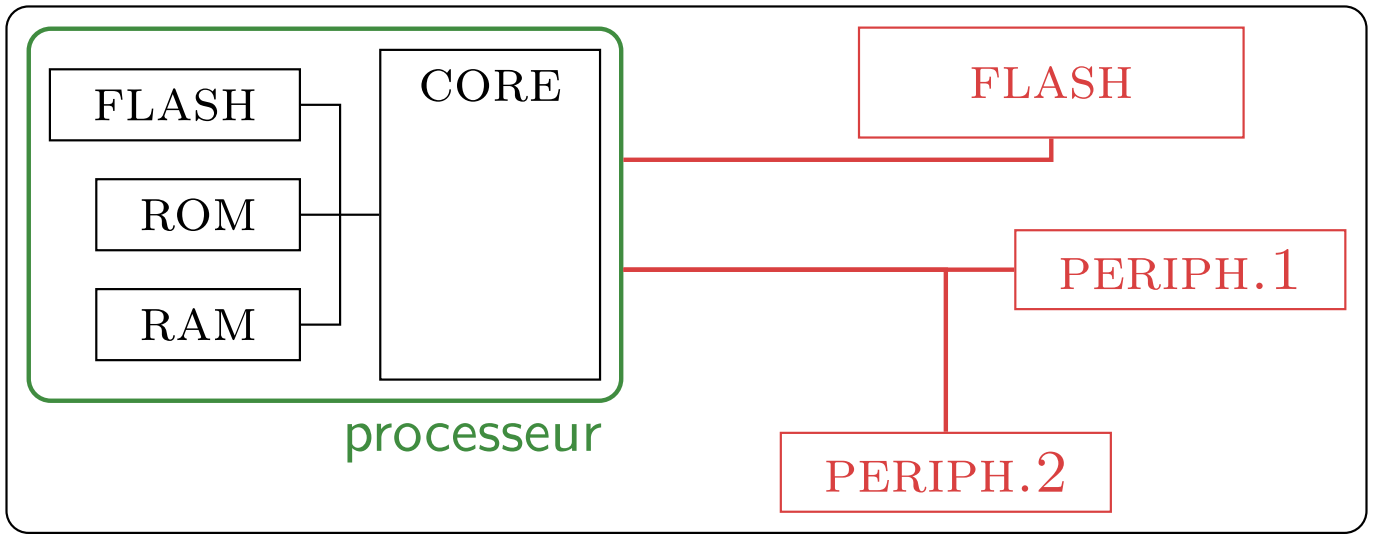




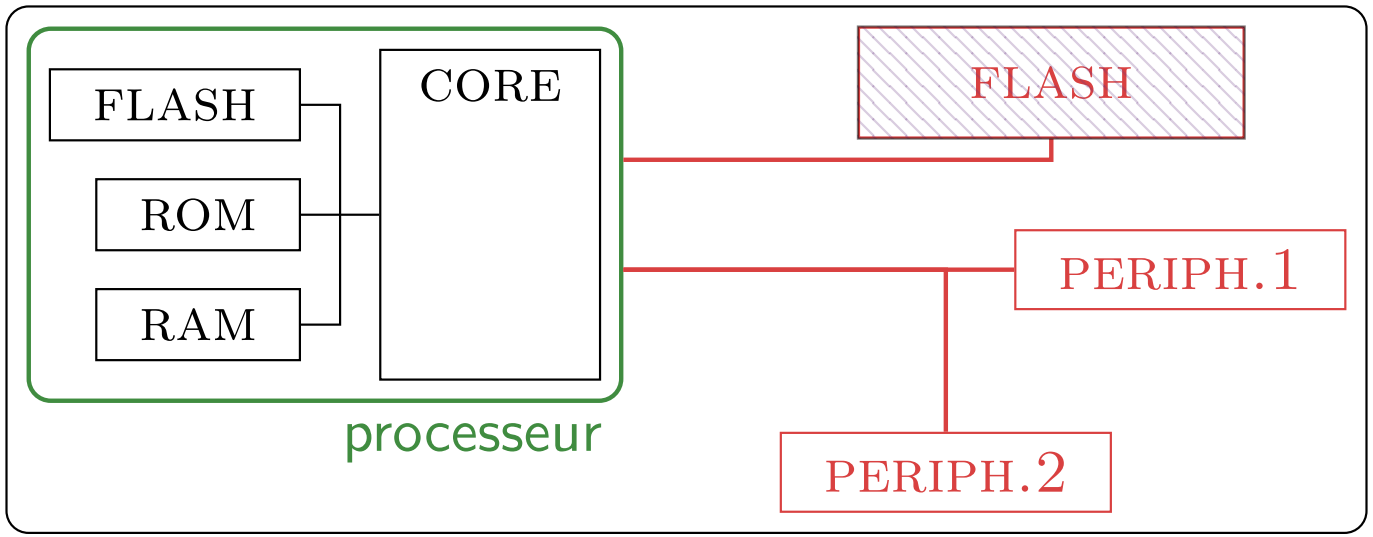
Si l'attaquant ne peut pas sonder les bus et mémoires internes,

- la zone de confiance est le processeur,
- l'attaquant accède à la FLASH externe et son bus,
- l'attaquant accède aux périphériques et leur bus.

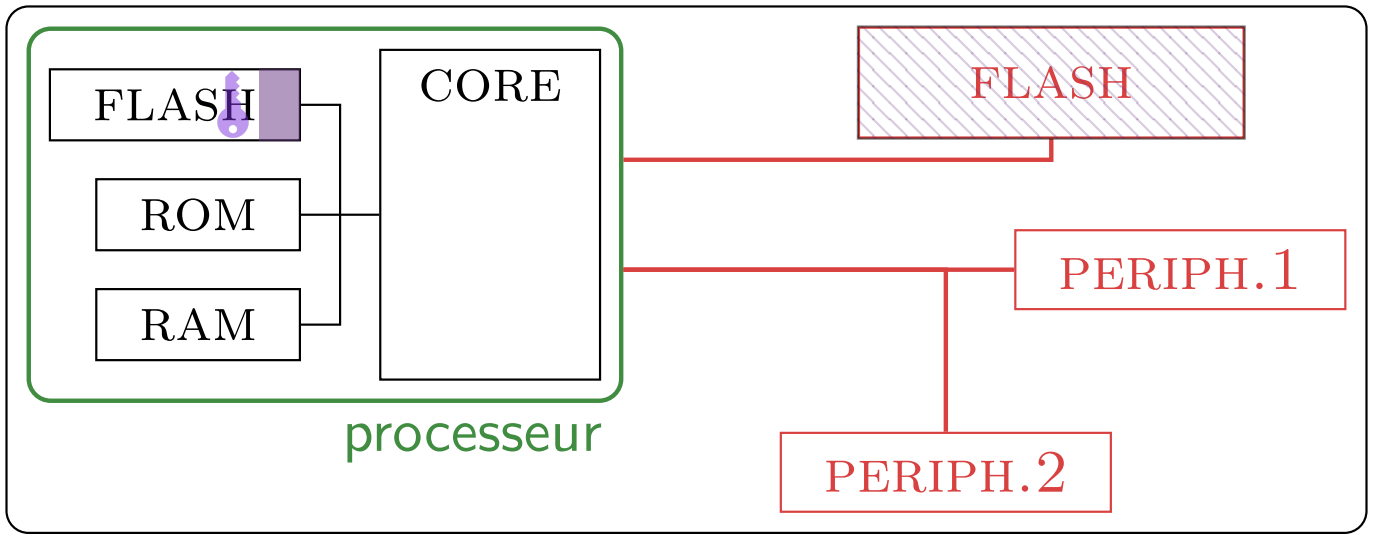

FLASH
ROM
RAM
CORE
processeur
FLASH
PERIPH.1
PERIPH.2



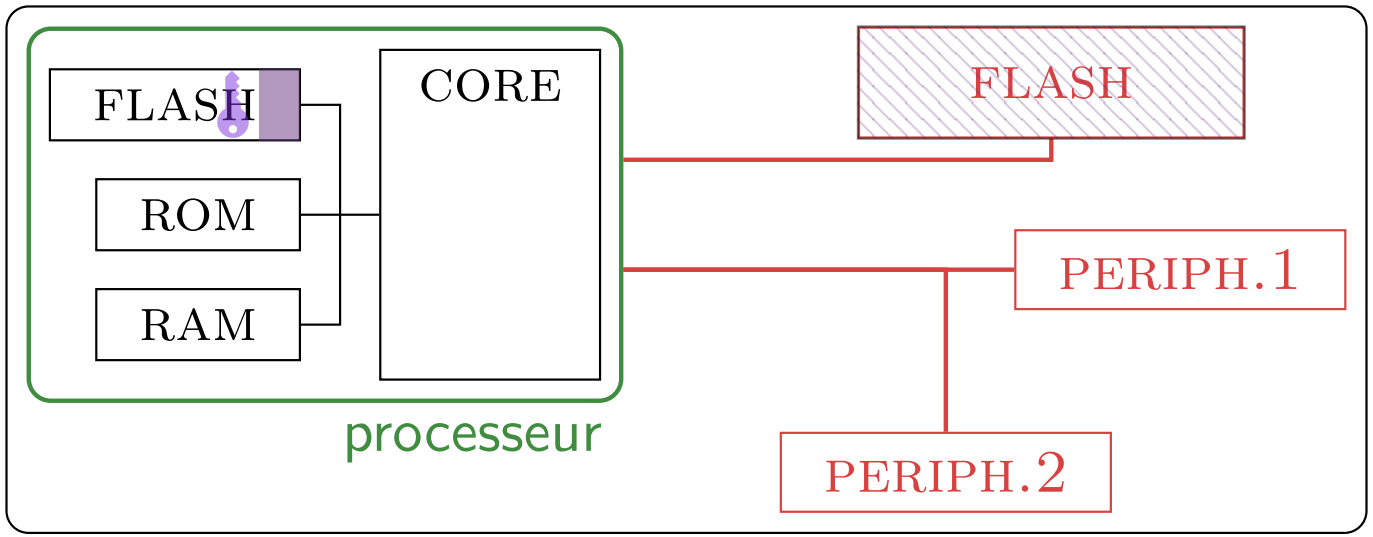
- Modification de la mise à jour.



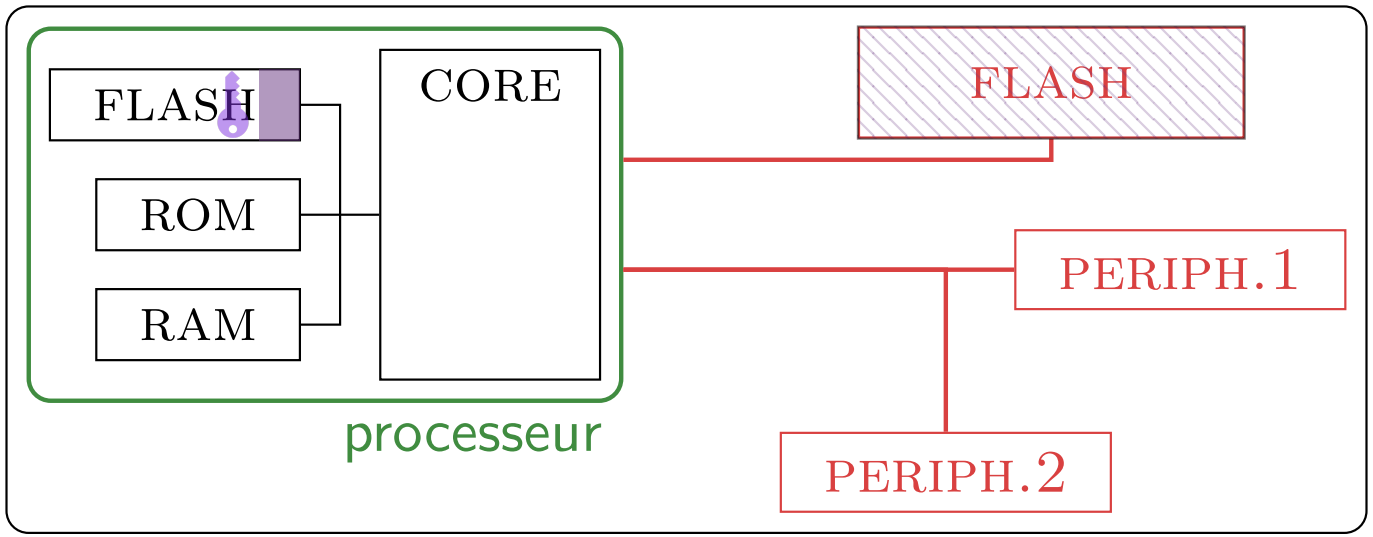
- Modification de la mise à jour.
  $\Longrightarrow$ MAC (+ chiffrement pour protection contre la rétro).



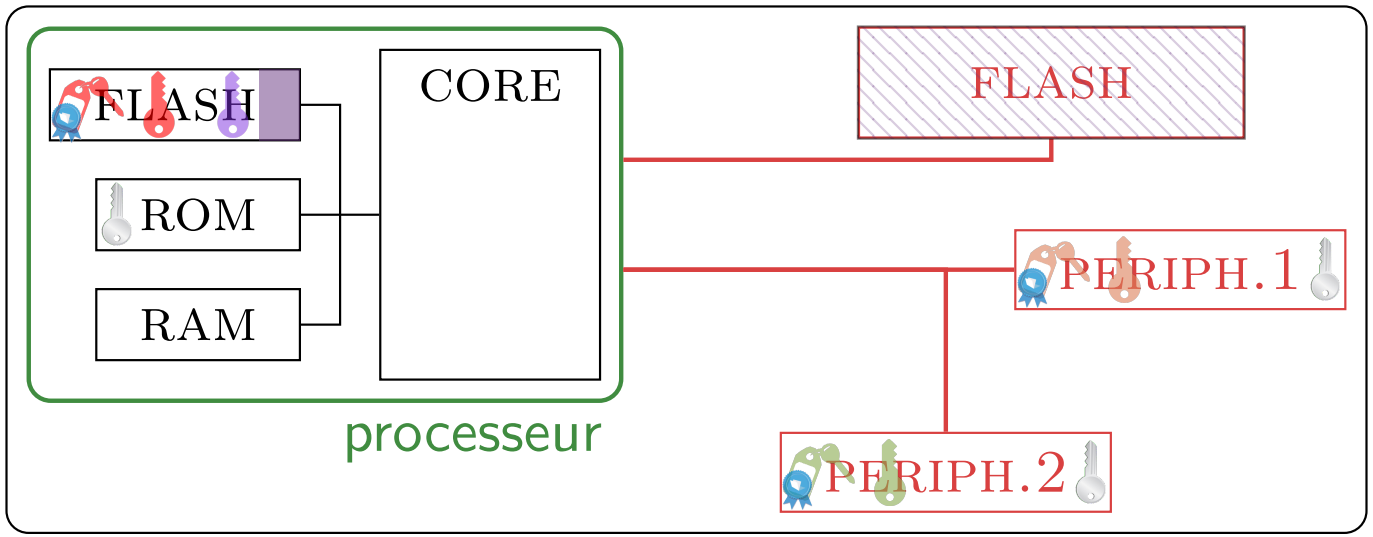
- Modification de la mise à jour.
  $\Longrightarrow$ MAC (+ chiffrement pour protection contre la rétro).



- Modification de la mise à jour.
  $\Longrightarrow$ MAC (+ chiffrement pour protection contre la rétro).
- Espionnage / modifications sur le bus des périphériques.



- Modification de la mise à jour.
  $\Longrightarrow$ MAC (+ chiffrement pour protection contre la rétro).
- Espionnage / modifications sur le bus des périphériques.
  $\Longrightarrow$ Canal sécurisé via échange de clefs.



- Modification de la mise à jour.
  $\Longrightarrow$ MAC (+ chiffrement pour protection contre la rétro).
- Espionnage / modifications sur le bus des périphériques.
  $\Longrightarrow$ Canal sécurisé via échange de clefs.
  et donc certificats et clef publique racine

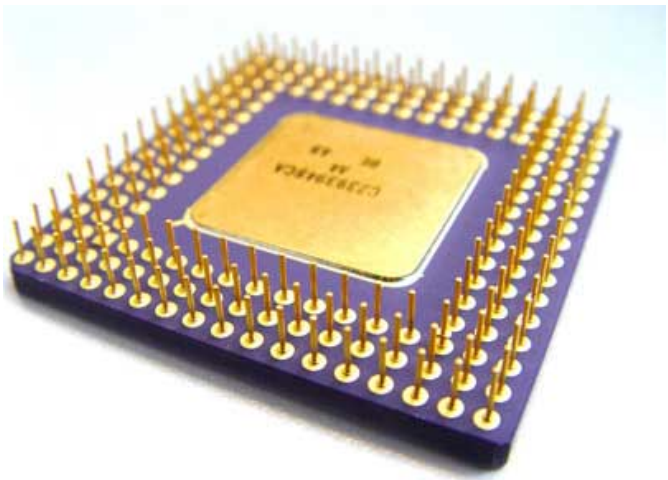
attaques actives

attaques actives

attaques passives

attaques actives

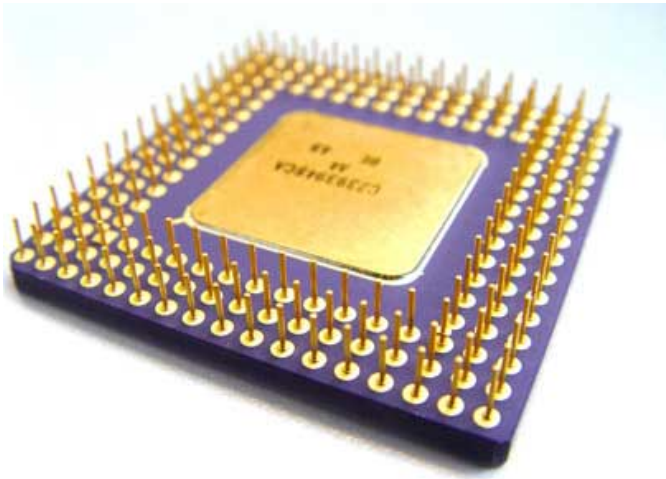
attaques passives



attaques actives

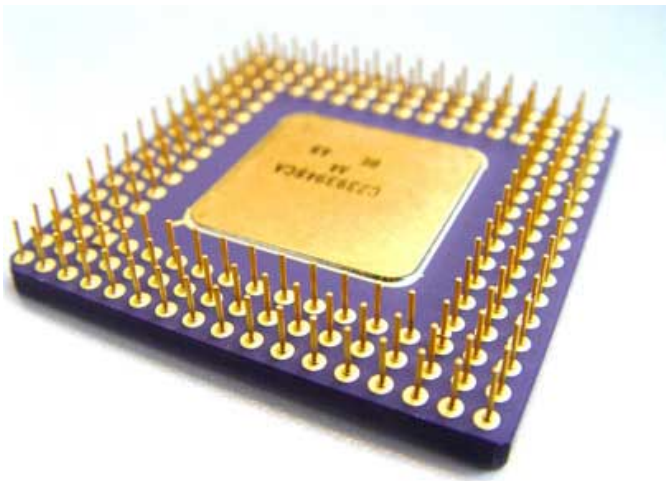
attaques passives

attaques actives

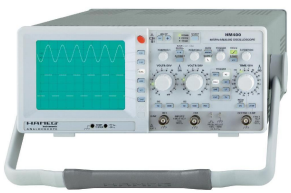


attaques actives

attaques passives



attaques actives



attaques passives

Attaques passives

- Canaux auxiliaires (temps de calcul, consommation ...)

Attaques actives

- Injection de fautes (rayon laser, variation de température ...)
- Delayreing et rétroconception
- Cheval de Troie
- Micro-probing

Moyens

- Glitch sur l'horloge ou l'alimentation.
- Préparation de l'échantillon (ouverture) puis
  - tir laser sur les transistors,
  - injection localisée d'ondes EM.

Effets

Modification de l'opcode

- en mémoire,
- sur le bus,
- au décodage de l'instruction,

vers une instruction sans effet de bords pour cette exécution.

RSA : $N = p\,q$ et

$$m = c^d \mod N$$

RSA-CRT :

$$m_p = c^{d \mod p-1} \mod p \quad , \quad m_q = c^{d \mod q-1} \mod q$$

Différentes techniques de recombinaison

Classique :

$$m = (m_q\,p^{-1} \mod q)\,p + (m_p\,q^{-1} \mod p)\,q \mod N$$

Garner :

$$m = m_q + q(q^{-1}(m_p - m_q) \mod p)$$

Faute : $m_p \longrightarrow \hat{m}_p$

$$\begin{aligned} m &= (m_q\, p^{-1} \mod q)\, p + (m_p\, q^{-1} \mod p)\, q \mod N \\ \hat{m} &= (m_q\, p^{-1} \mod q)\, p + (\hat{m}_p\, q^{-1} \mod p)\, q \mod N \end{aligned}$$

$$m - \hat{m} = \big((m_p - \hat{m}_p)\, q^{-1} \mod p\big)\, q \mod N$$

On a un multiple de q : on retrouve q avec

$$q = \mathrm{pgcd}(N, m - \hat{m})$$

Faute : $m_p \longrightarrow \hat{m}_p$

$$\begin{aligned} m &= m_q + q(q^{-1}(m_p - m_q) \mod p) \\ \hat{m} &= m_q + q(q^{-1}(\hat{m}_p - m_q) \mod p) \end{aligned}$$

$$m - \hat{m} = q\left(q^{-1}(m_p - \hat{m}_p) \mod p\right)$$

On a un multiple de q : on retrouve q avec

$$q = \mathrm{pgcd}(N, m - \hat{m})$$

Faute : $m_q \longrightarrow \hat{m}_q$

$$\begin{aligned} m &= m_q + q(q^{-1}(m_p - m_q) \mod p) \\ \hat{m} &= \hat{m}_q + q(q^{-1}(m_p - \hat{m}_q) \mod p) \end{aligned}$$

$$\begin{aligned} m - \hat{m} &= m_q - \hat{m}_q + q\left(q^{-1}(\hat{m}_q - m_q) \mod p\right) \\ &= 0 \mod p \end{aligned}$$

On a un multiple de p : on retrouve p avec

$$p = \mathrm{pgcd}(N, m - \hat{m})$$

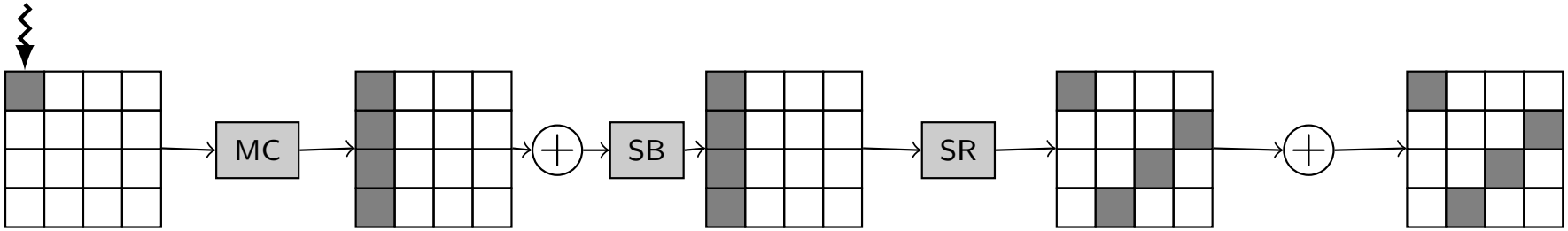
Il existe des contre-mesures aux attaques précédentes.

On peut attaquer avec plus de calculs fautés ou plus de fautes durant un calcul.

Mais il existe aussi une attaque en faute sur N :

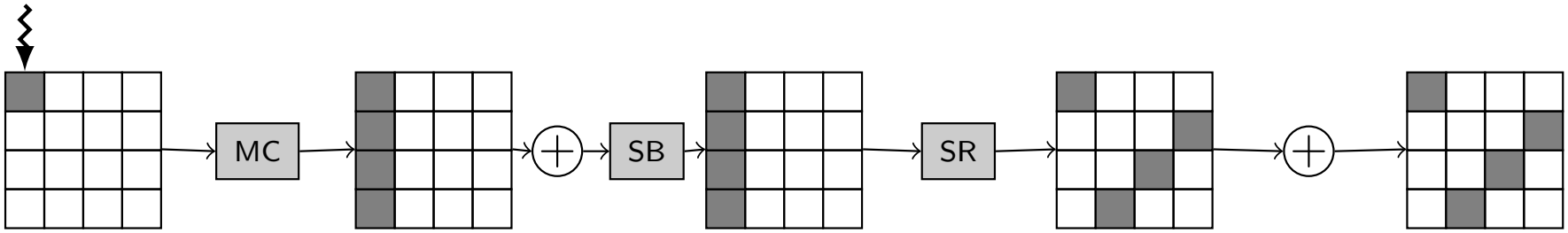
$$\begin{aligned} m &= (m_q\, p^{-1} \mod q)\, p + (m_p\, q^{-1} \mod p)\, q \mod N \\ \hat{m} &= (m_q\, p^{-1} \mod q)\, p + (m_p\, q^{-1} \mod p)\, q \mod \hat{N} \end{aligned}$$

On peut retrouver la factorisation de N par réduction de réseaux.



Case noire : il y a une différence entre deux exécutions.

Case blanche : il n'y a pas de différence entre deux exécutions.



Seulement 2^8 valides

sur 2^{32}.

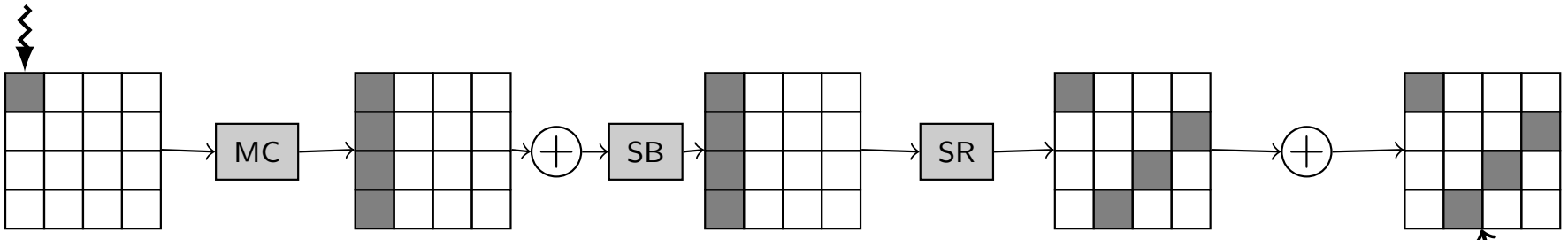
MixColumn est linéaire :

$$\texttt{MC}(P) \oplus \texttt{MC}(P \oplus E) = \texttt{MC}(E).$$

Les différences possibles sont donc :

$$\{\texttt{MC}(0), \texttt{MC}(1), \texttt{MC}(2), \ldots, \texttt{MC}(255)\}.$$

Rappels cryptographiques
Faute sur le tour 9 (Piret et Quisquater)

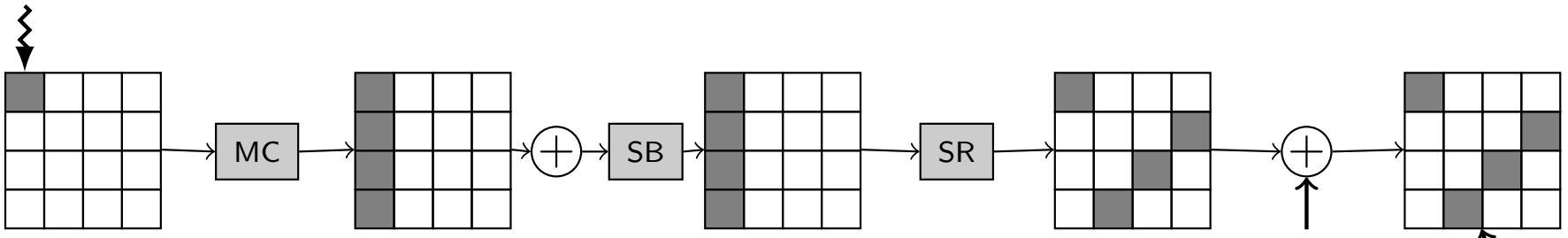


Seulement 2^8 valides

sur 2^{32}.

on connait les chiffrés

Rappels cryptographiques
Faute sur le tour 9 (Piret et Quisquater)



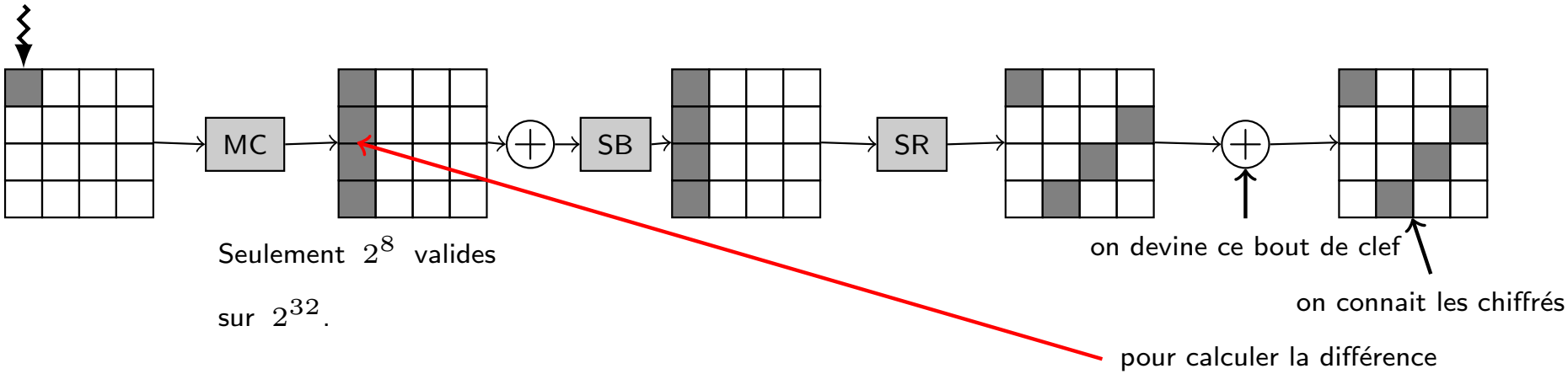
Seulement 2^8 valides

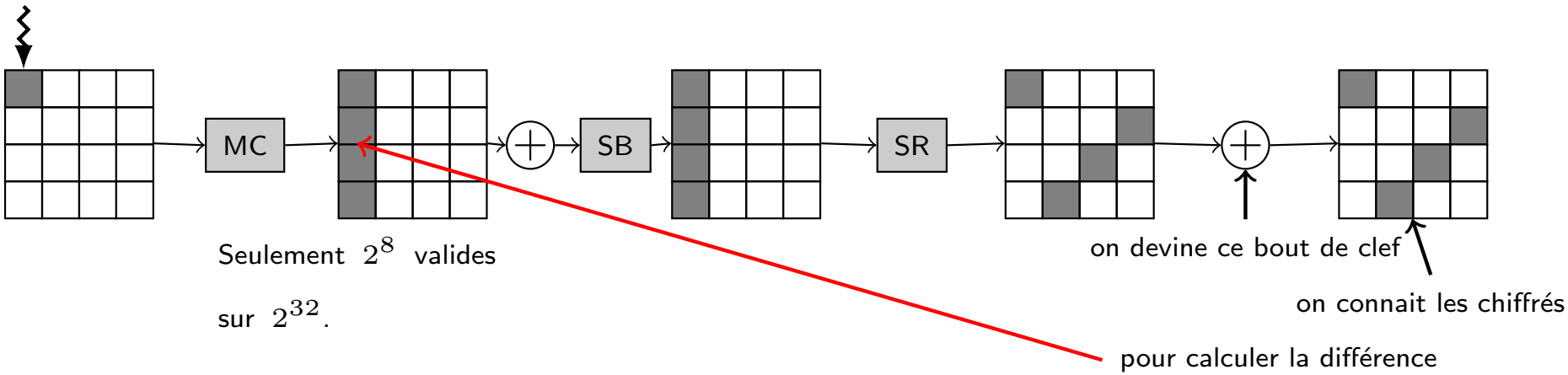
sur 2^{32}.

on devine ce bout de clef

on connait les chiffrés

Rappels cryptographiques
Faute sur le tour 9 (Piret et Quisquater)



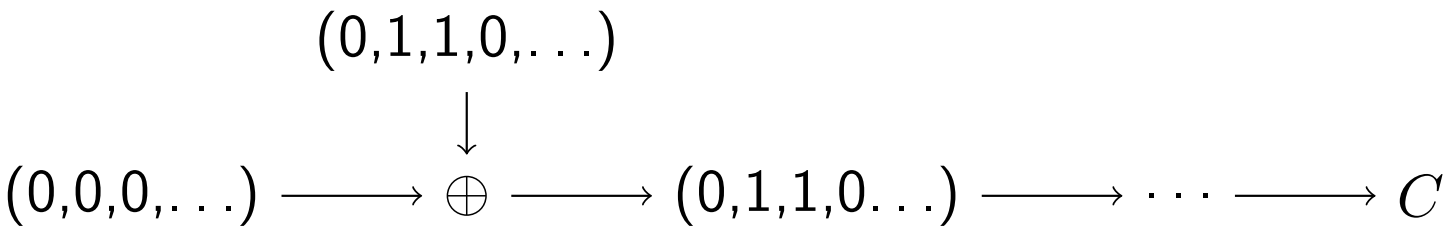


1. On précalcule l'ensemble des 2^8 "bonnes valeurs".
   - bonne clef : la différence appartient au bon ensemble,
   - mauvaise clef : la différence est *aléatoire* (et donc souvent hors de l'ensemble).
2. Il faut 2 fautes par colonne pour retrouver toute la clef.
3. On peut éviter de tester 2^{32} clefs car les calculs sont indépendents d'un octet à l'autre.

Modèle de faute

Collage à 0 d'un bit.

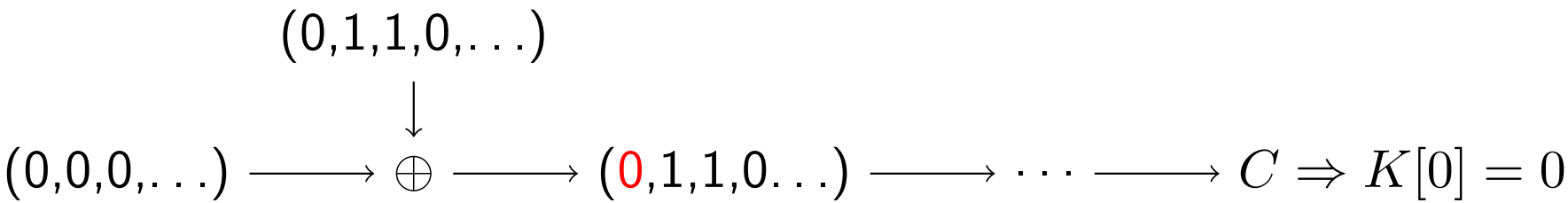
- Clair $P = (0, 0, \ldots, 0)$.
- Attaque un bit après XOR de la première sous-clef.



Modèle de faute

Collage à 0 d'un bit.

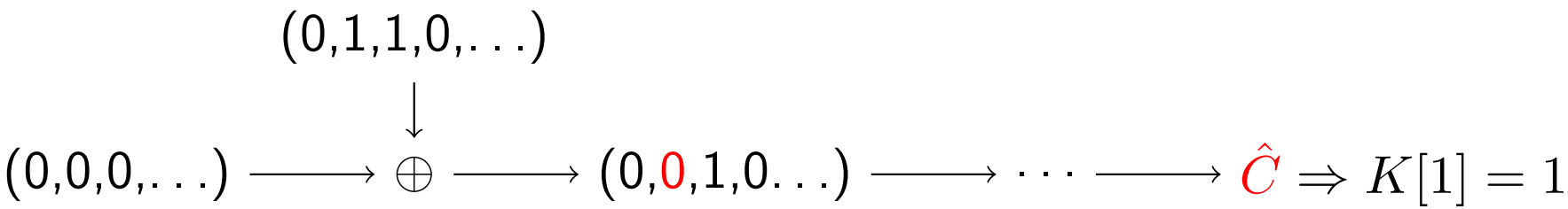
- Clair $P = (0, 0, \ldots, 0)$.
- Attaque un bit après XOR de la première sous-clef.



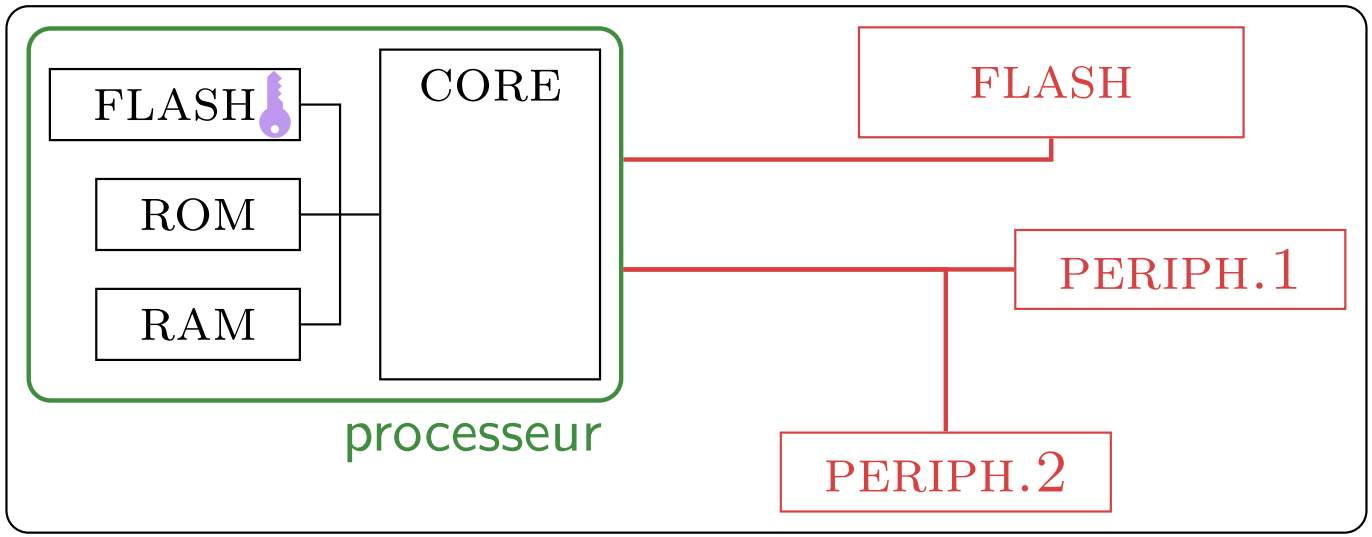
Modèle de faute

Collage à 0 d'un bit.

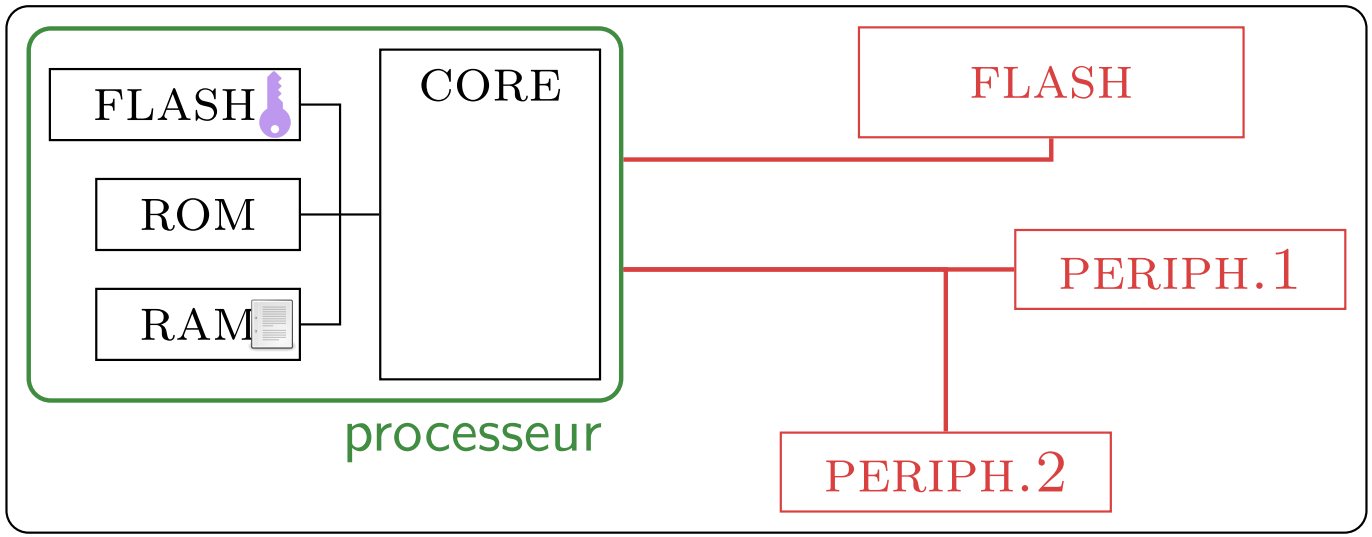
- Clair $P = (0, 0, \ldots, 0)$.
- Attaque un bit après XOR de la première sous-clef.



Écriture en FLASH externe

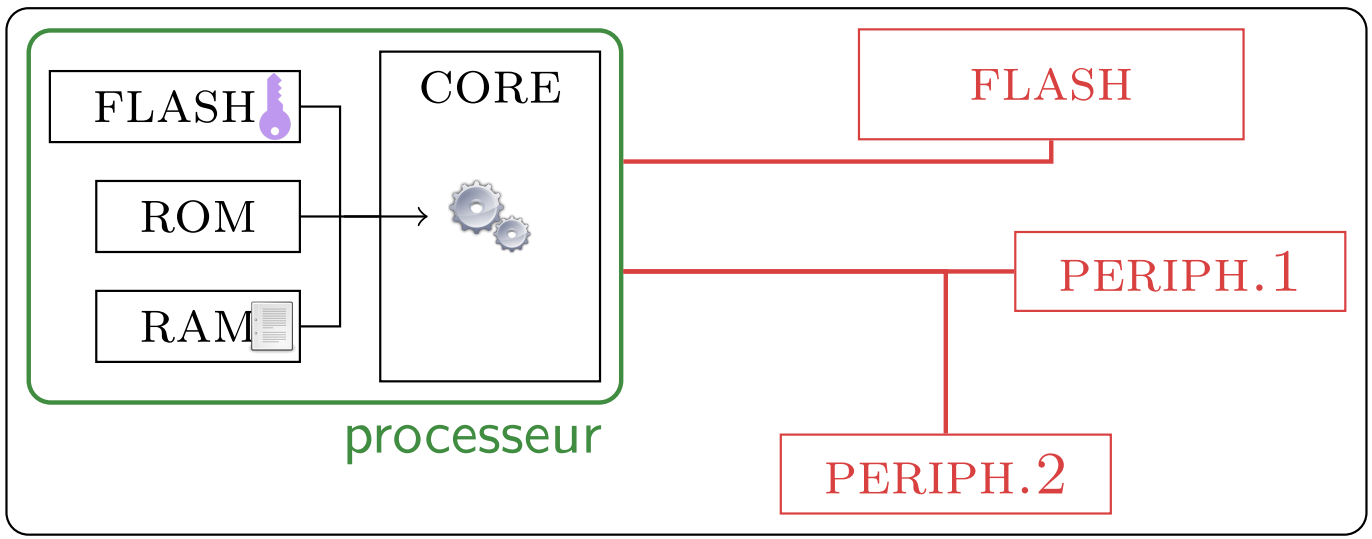


Écriture en FLASH externe



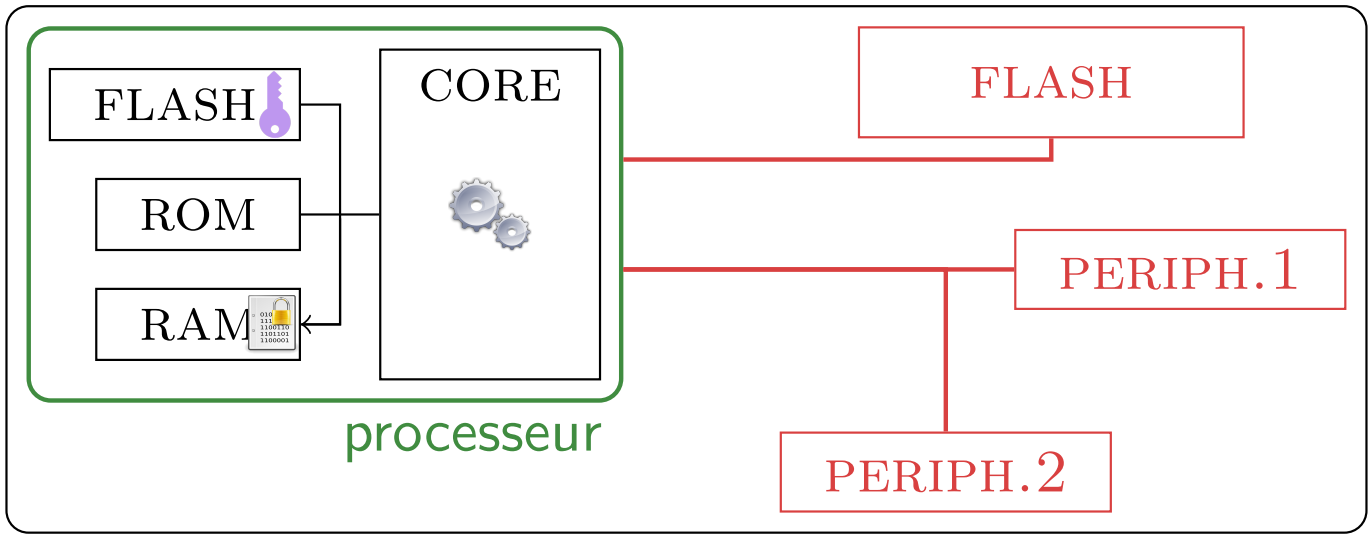
Fonctionnement nominal

Écriture en FLASH externe



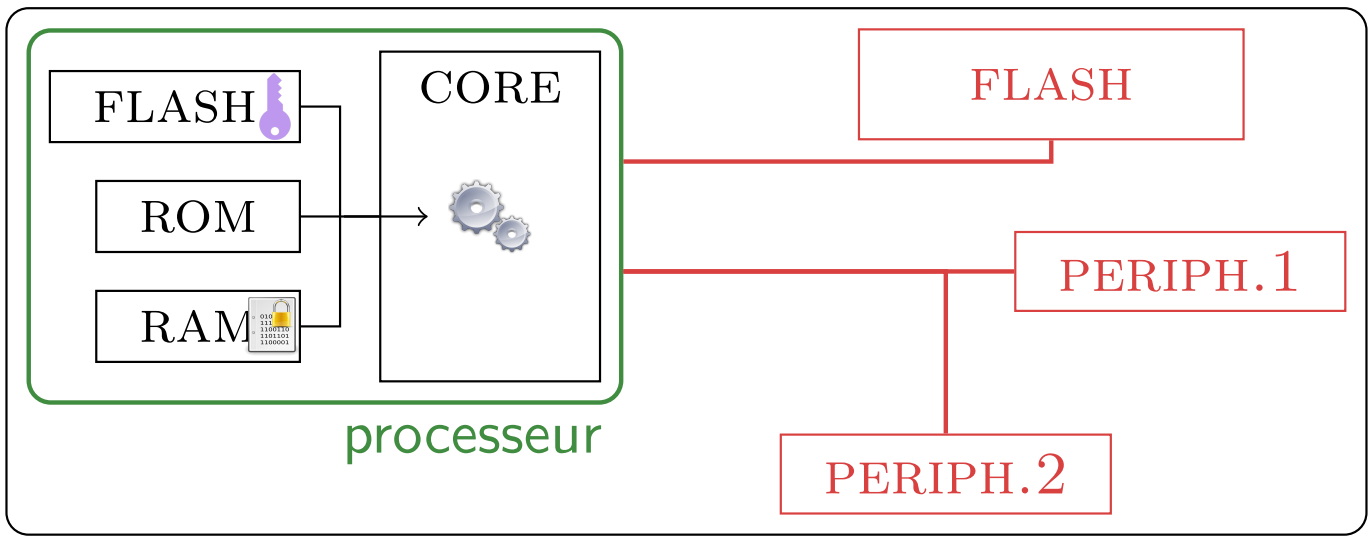
Fonctionnement nominal

Écriture en FLASH externe



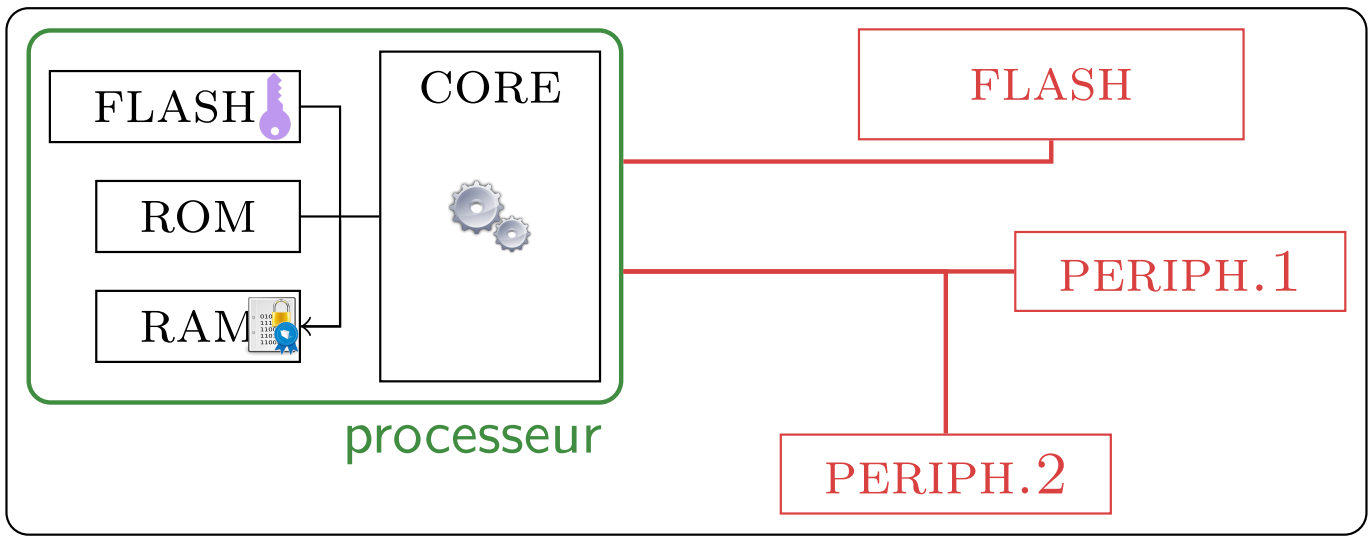
Fonctionnement nominal

Écriture en FLASH externe



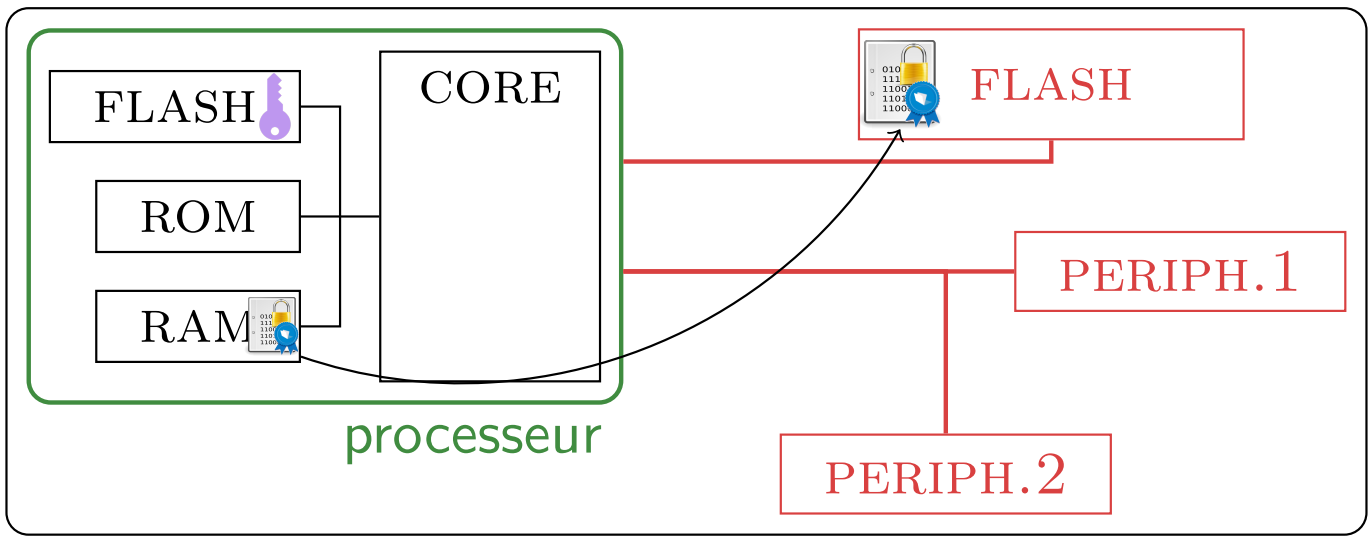
Fonctionnement nominal

Écriture en FLASH externe



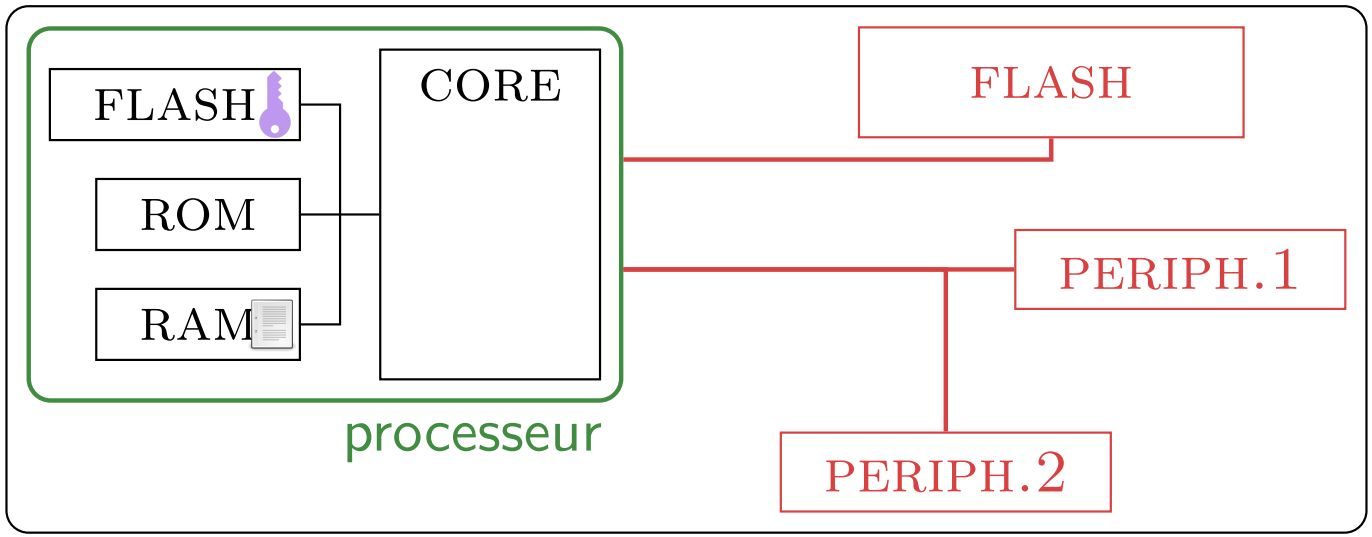
Fonctionnement nominal

Écriture en FLASH externe



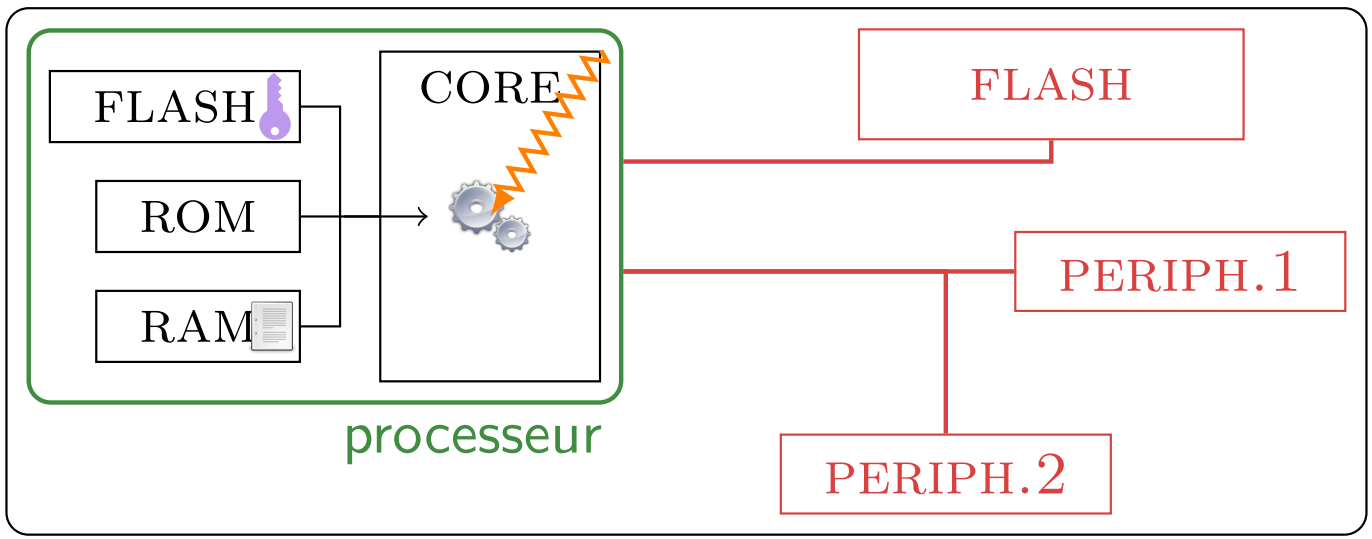
Fonctionnement nominal

Écriture en FLASH externe



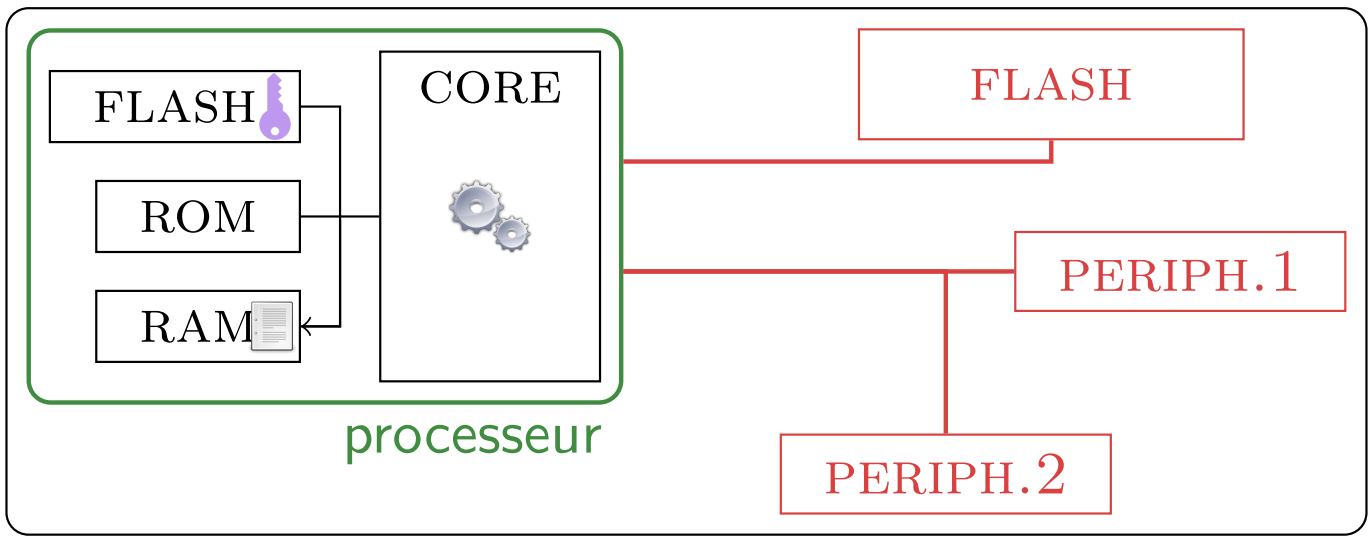
Fonctionnement avec faute

Écriture en FLASH externe



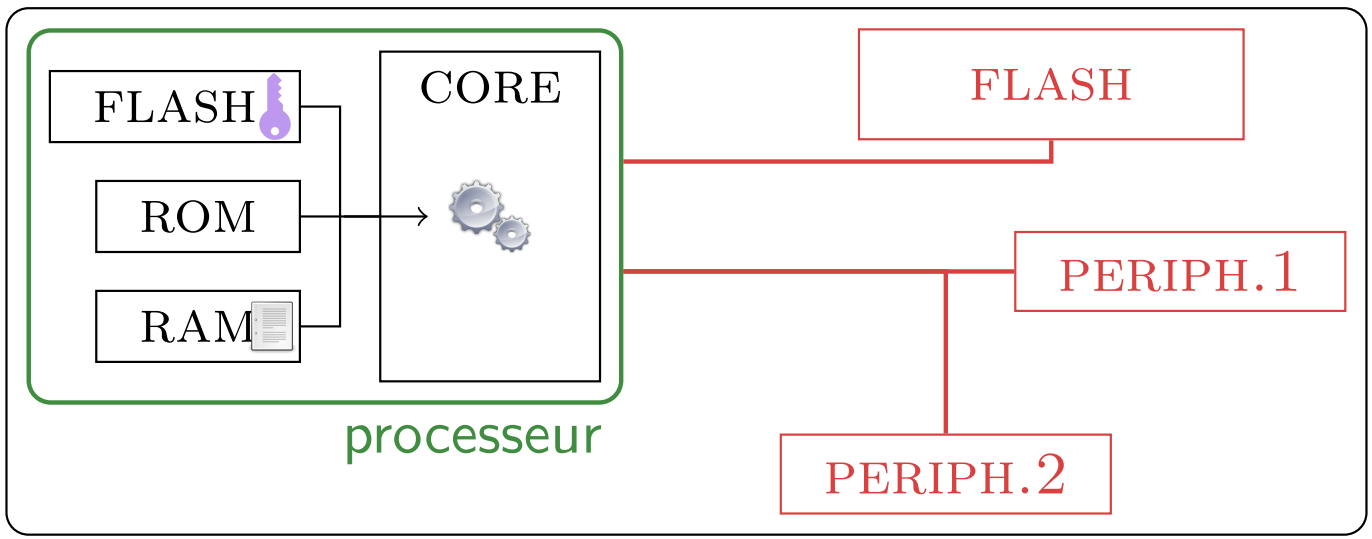
Fonctionnement avec faute

Écriture en FLASH externe



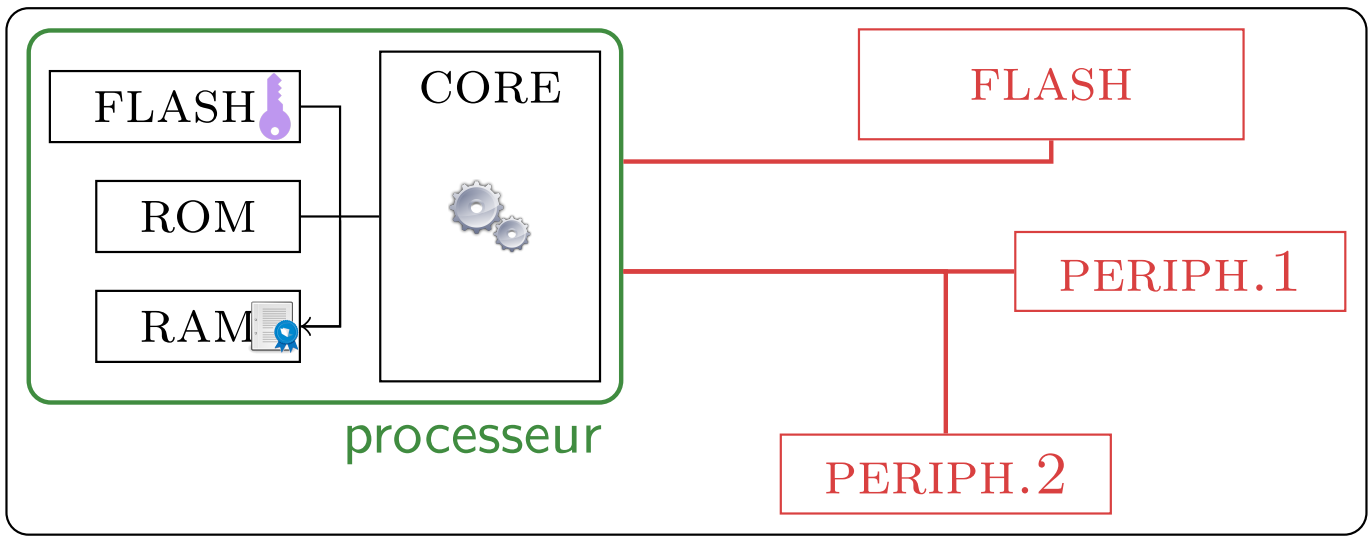
Fonctionnement avec faute

Écriture en FLASH externe



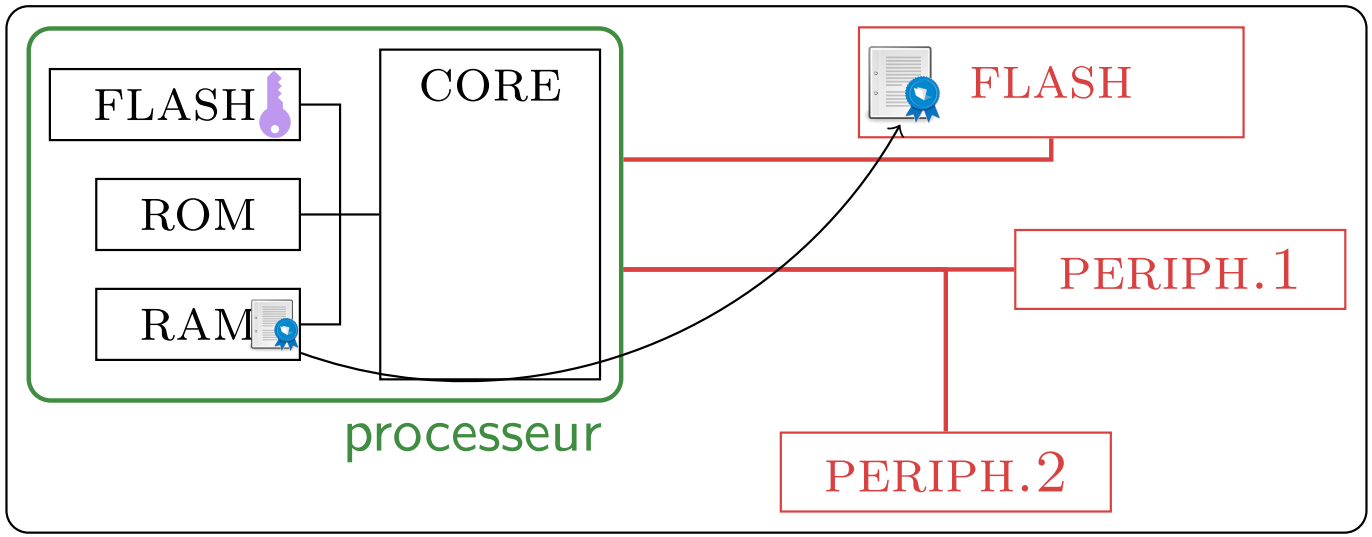
Fonctionnement avec faute

Écriture en FLASH externe



Fonctionnement avec faute

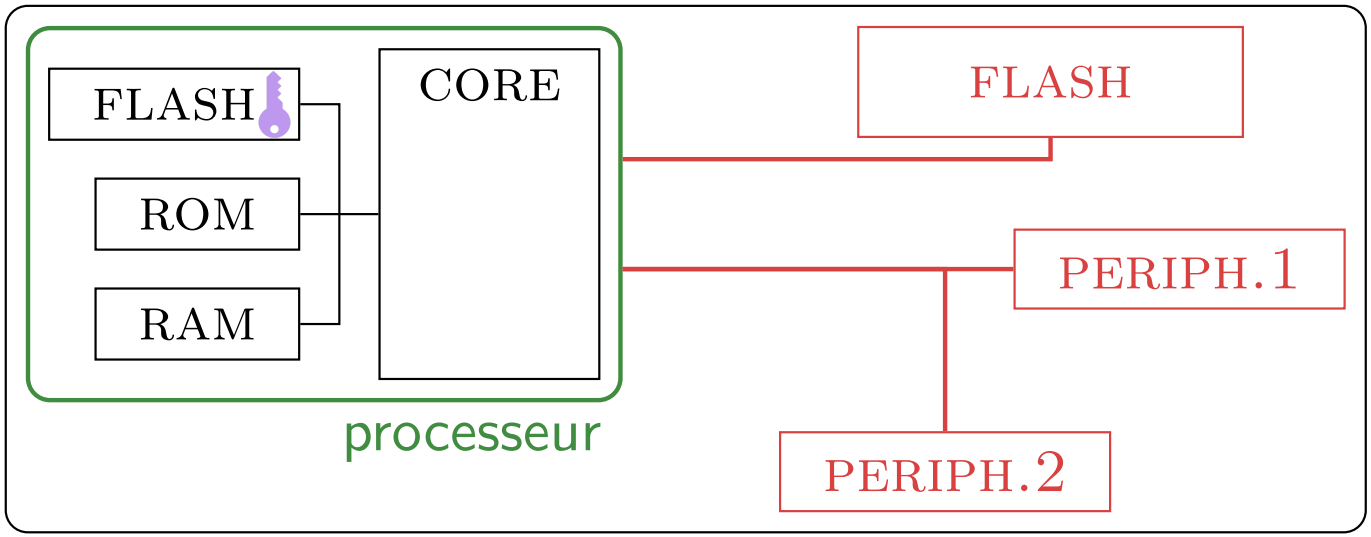
Écriture en FLASH externe



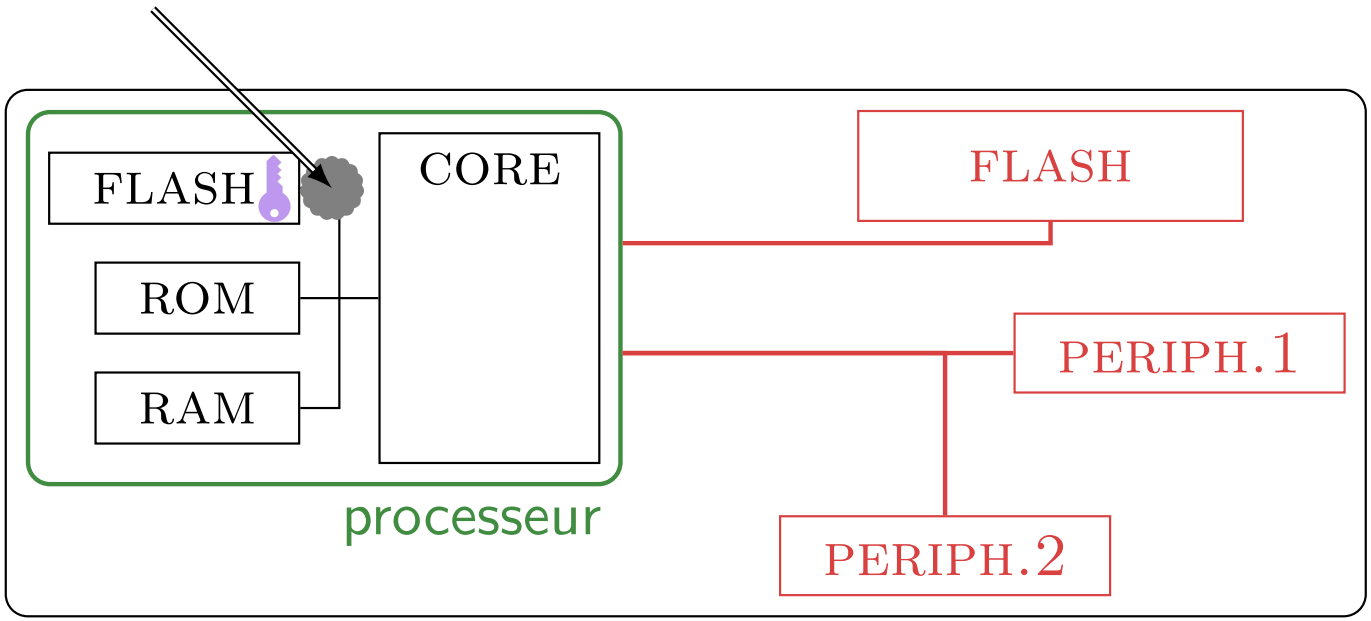
Fonctionnement avec faute

La donnée est stockée en clair en mémoire externe !

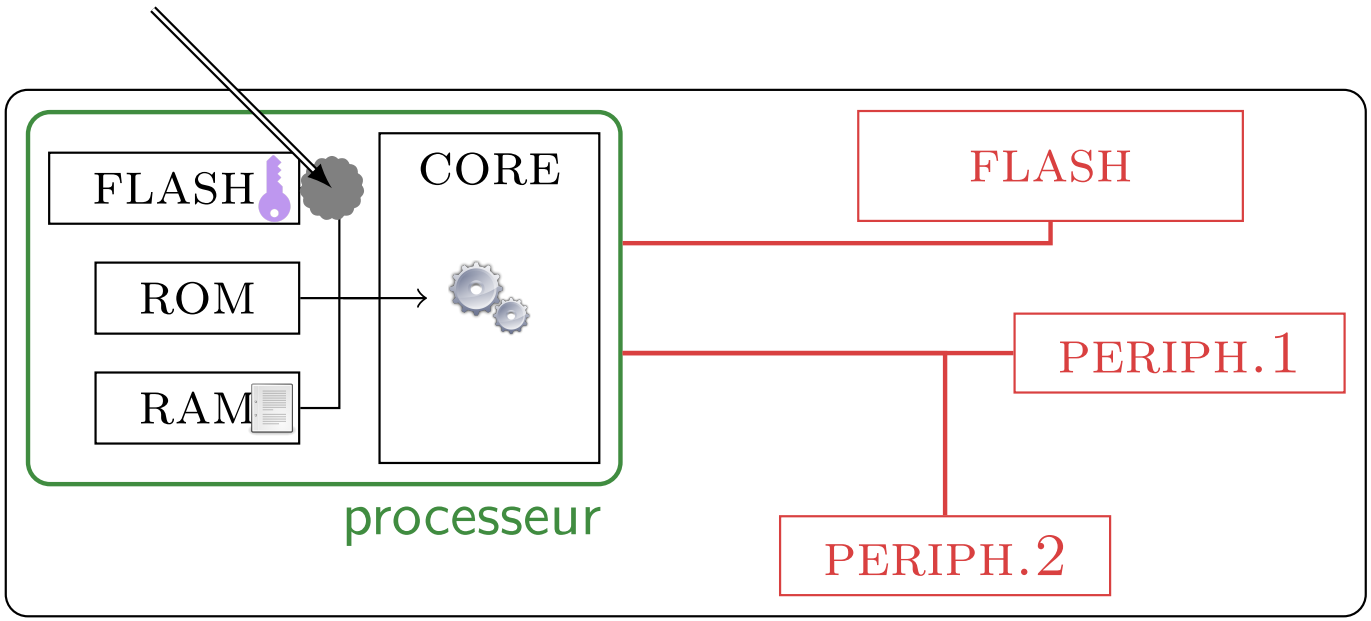
Utilisation interne d'une clef



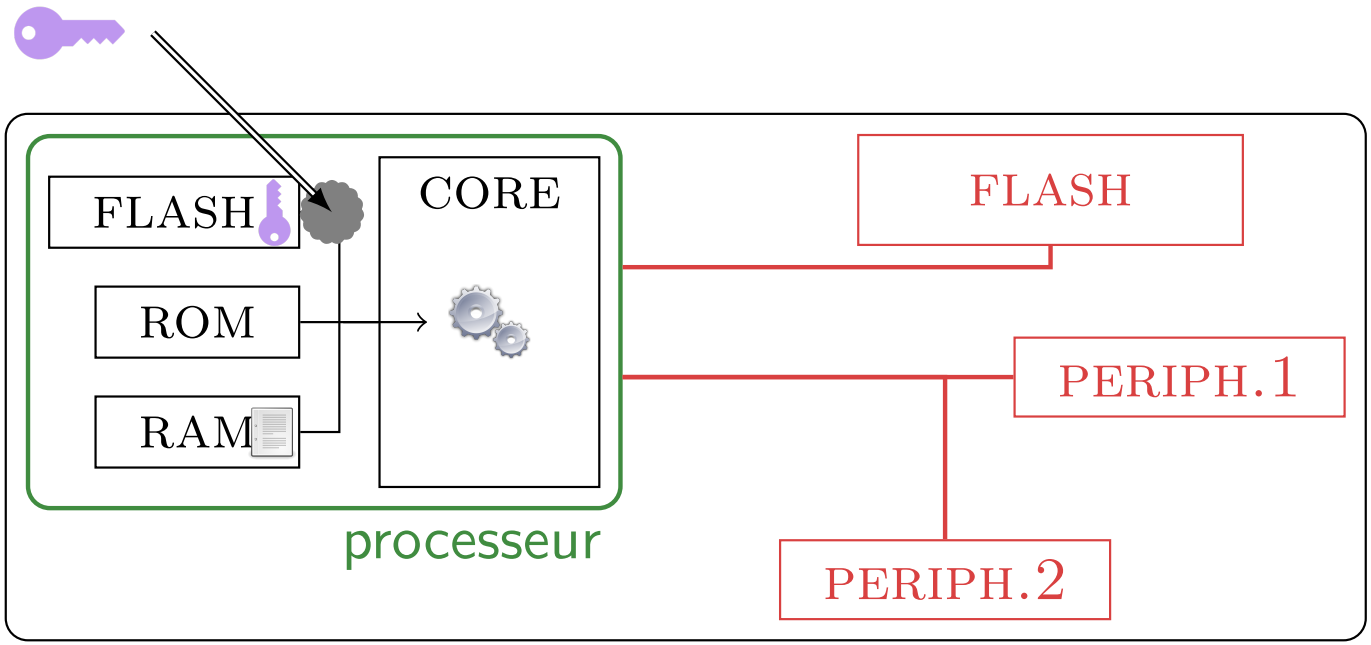
Utilisation interne d'une clef



Utilisation interne d'une clef



Utilisation interne d'une clef



On peut aussi directement récupérer une donnée.

Détecteurs

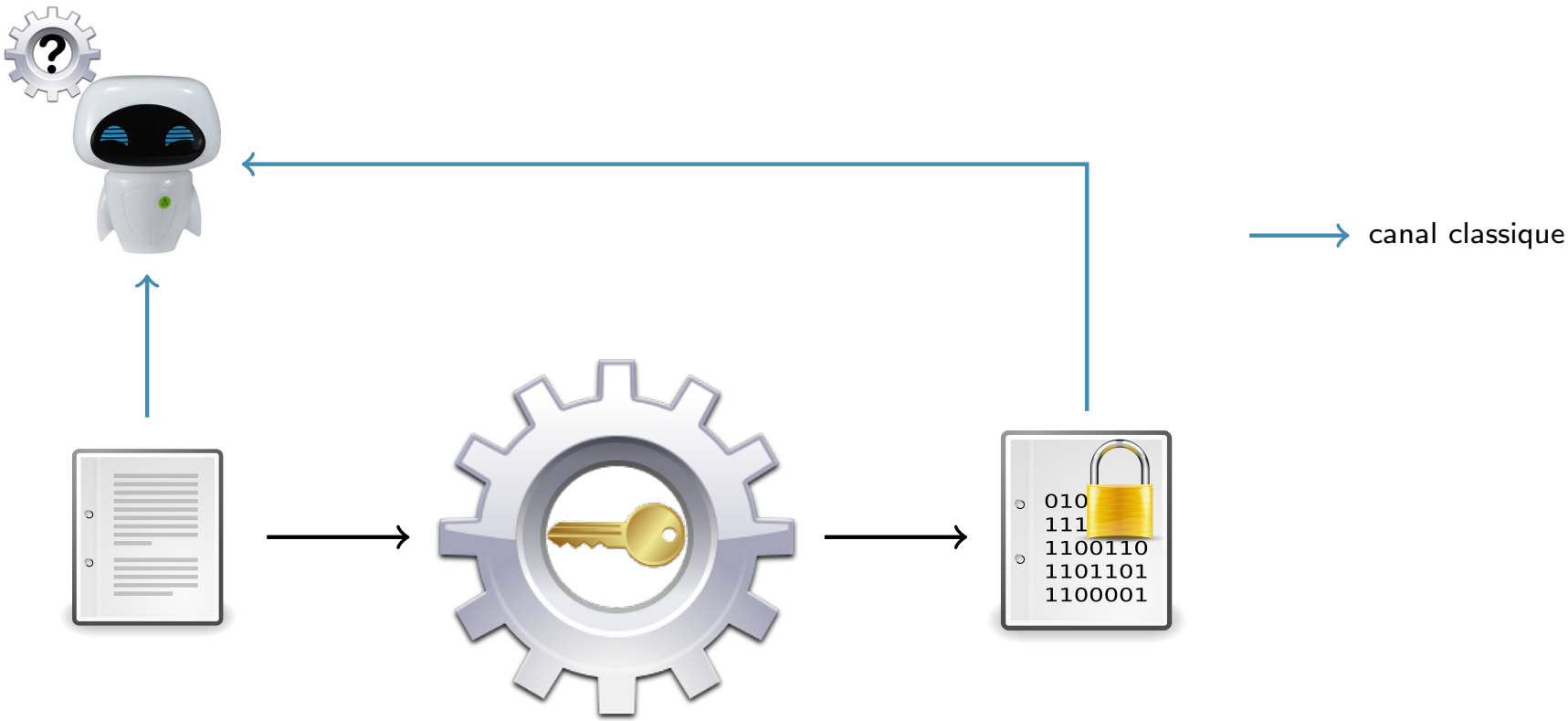
- Détecteurs optiques (fautes lasers).
- Lissage (PLL pour l'horloge, capacités pour l'alimentation).
- Shield actif de protection (micro-probing, ouverture).

Obfuscation

- Scrambling de la mémoire.
- Enfouissement des bus.
- Mélange des points mémoire à la logique.

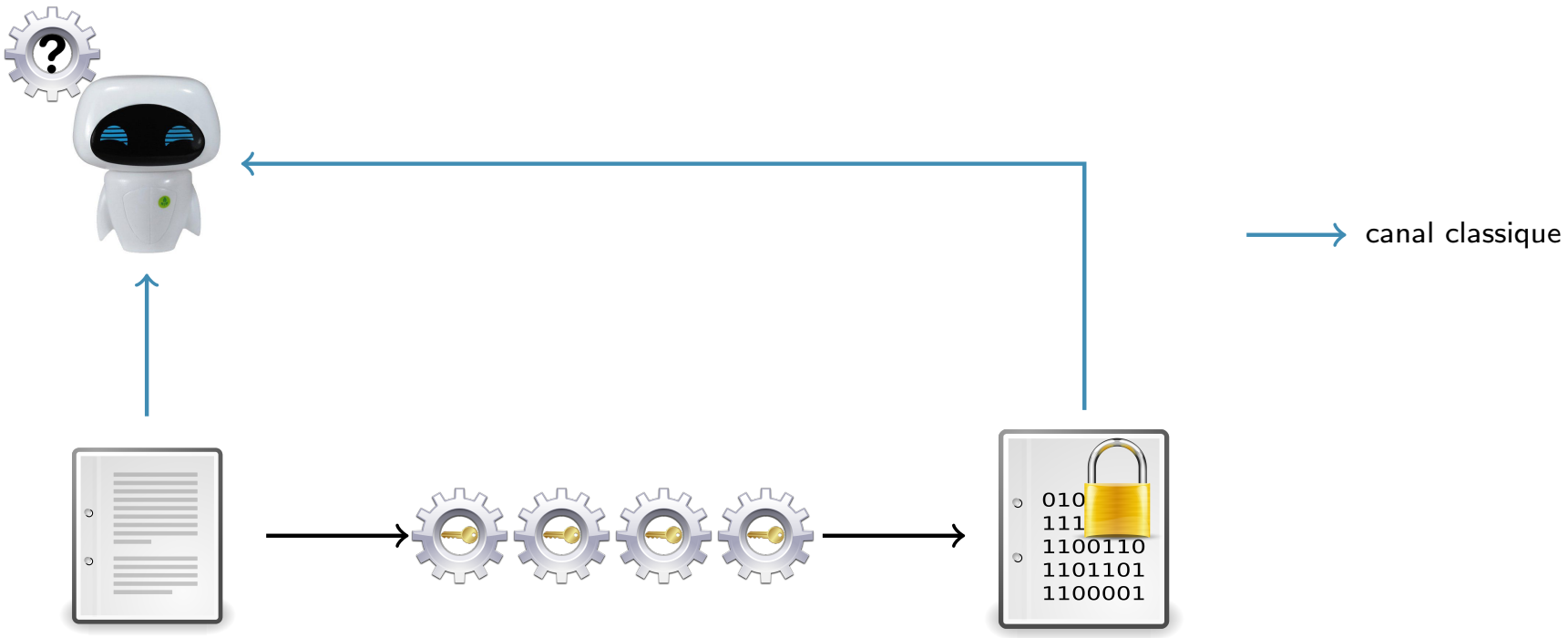
Attaques par canaux auxiliaires

Principe général



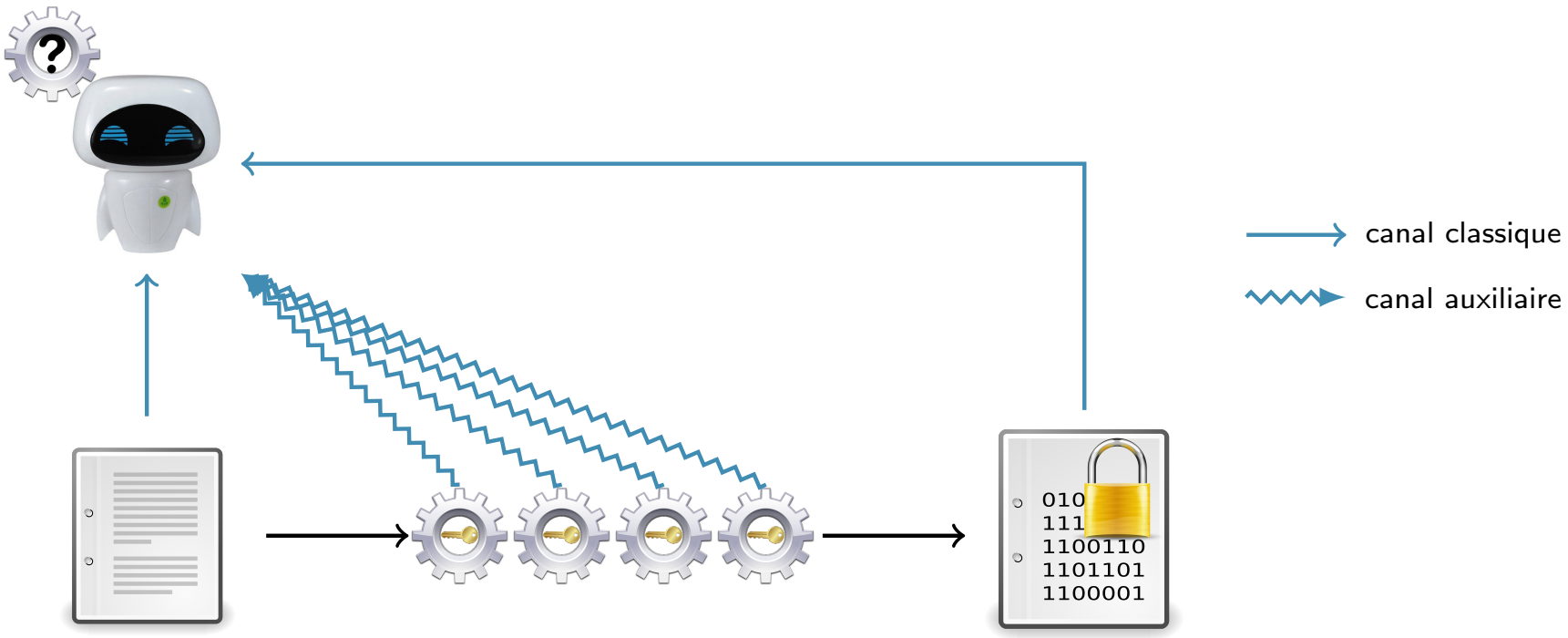
Attaques par canaux auxiliaires

Principe général



Attaques par canaux auxiliaires

Principe général



On obtient des informations sur les états intermédiaires

où les liens entre clair/chiffré et clef sont peu complexes

Toute observation peut être dangereuse :

- temps d'exécution,
- consommation de courant,
- rayonnements électromagnétiques,
- émissions de photons,
- émissions sonores,
- potentiel électrique d'un corps en contact avec le PC (!)
- ...

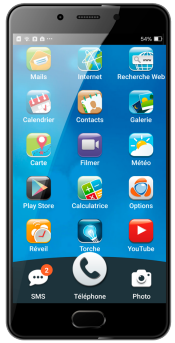
Attaques par canaux auxiliaires

Exemple du code PIN

Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



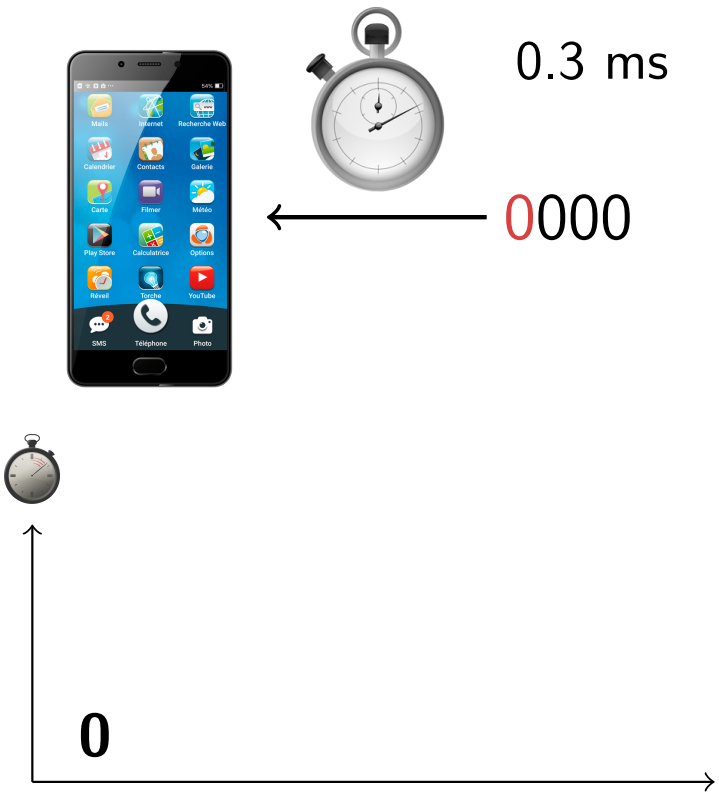
Attaques par canaux auxiliaires

Exemple du code PIN

Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



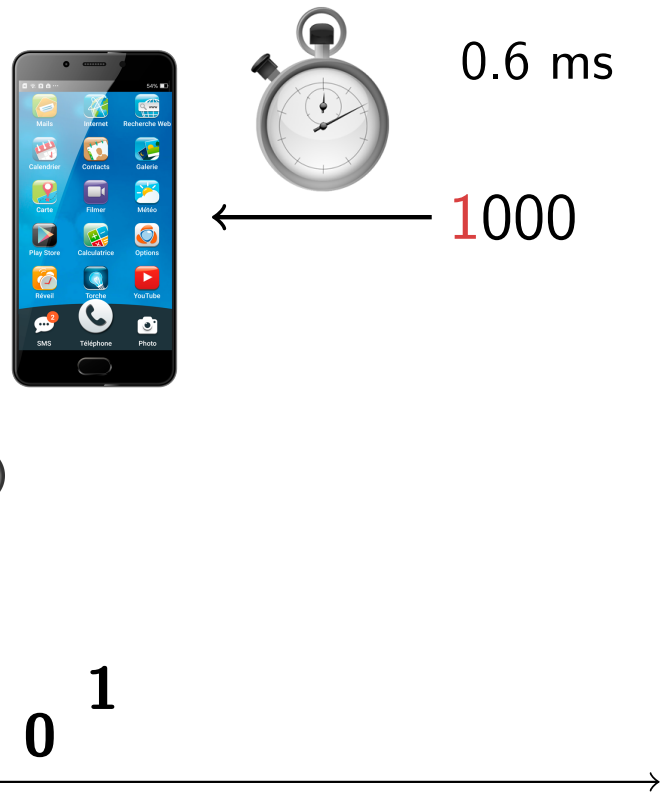
Attaques par canaux auxiliaires

Exemple du code PIN

Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

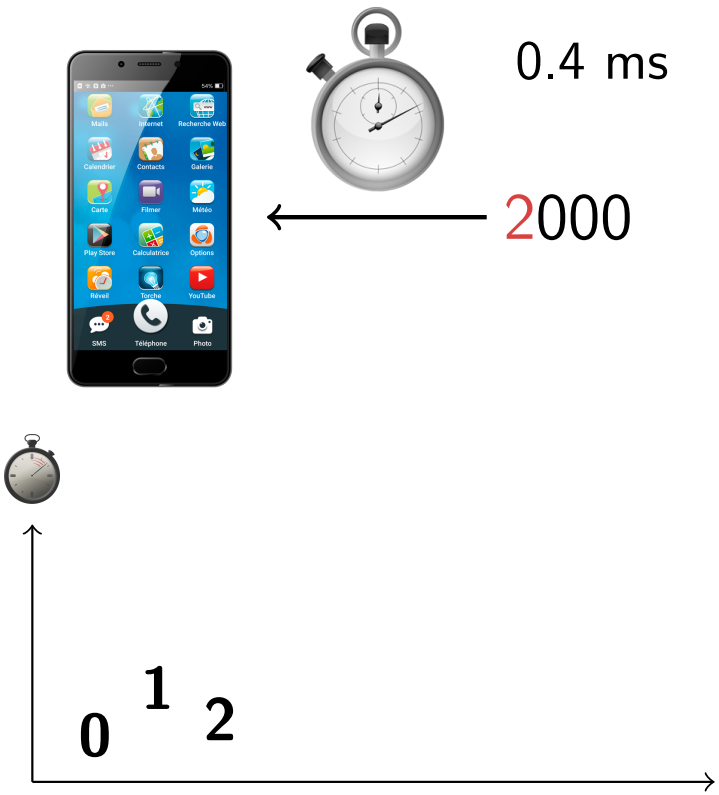
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

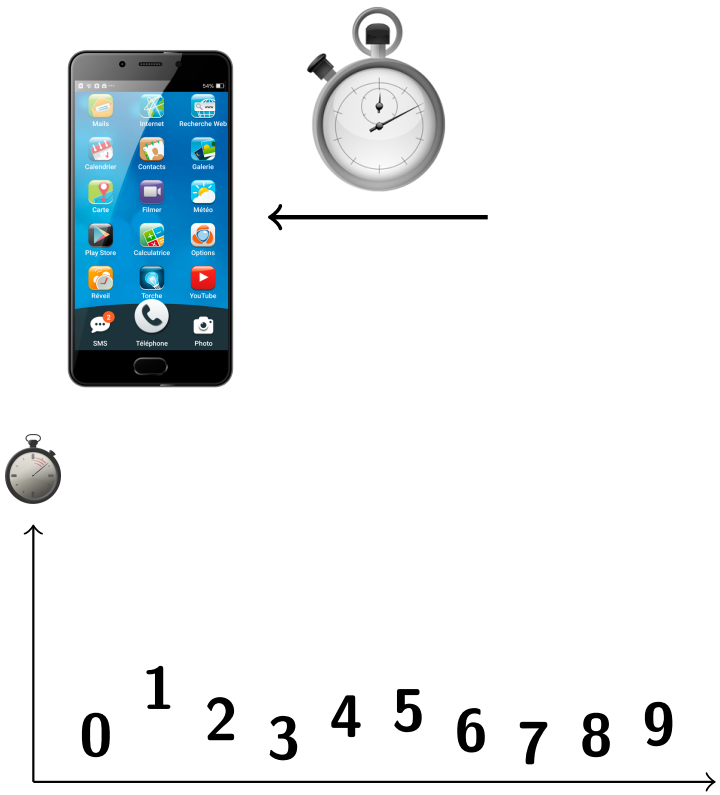
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

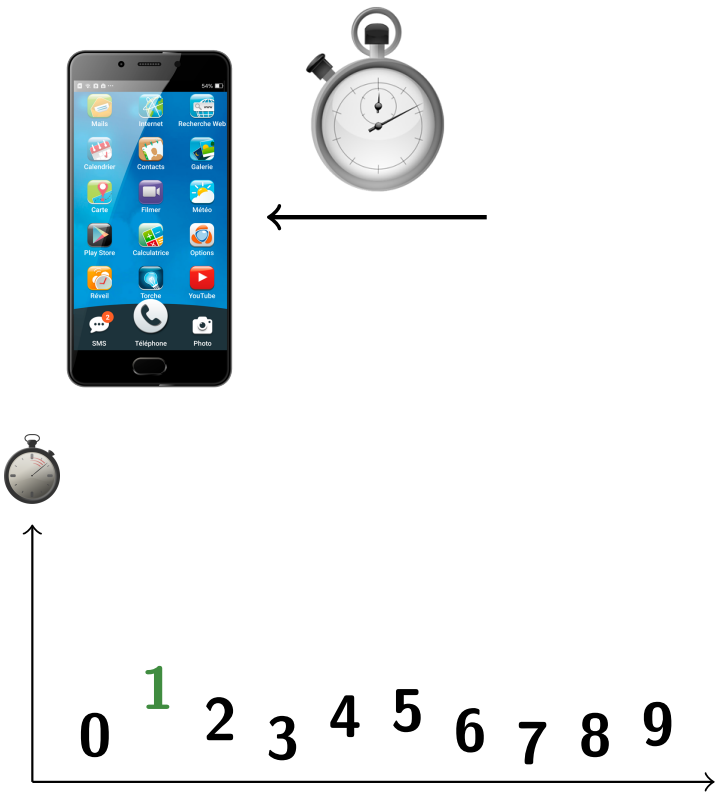
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

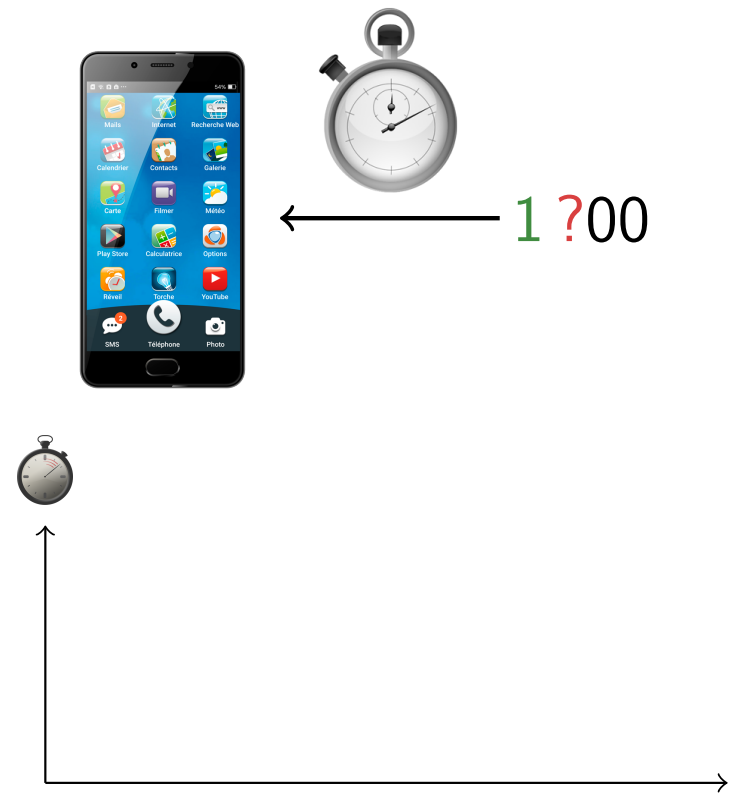
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



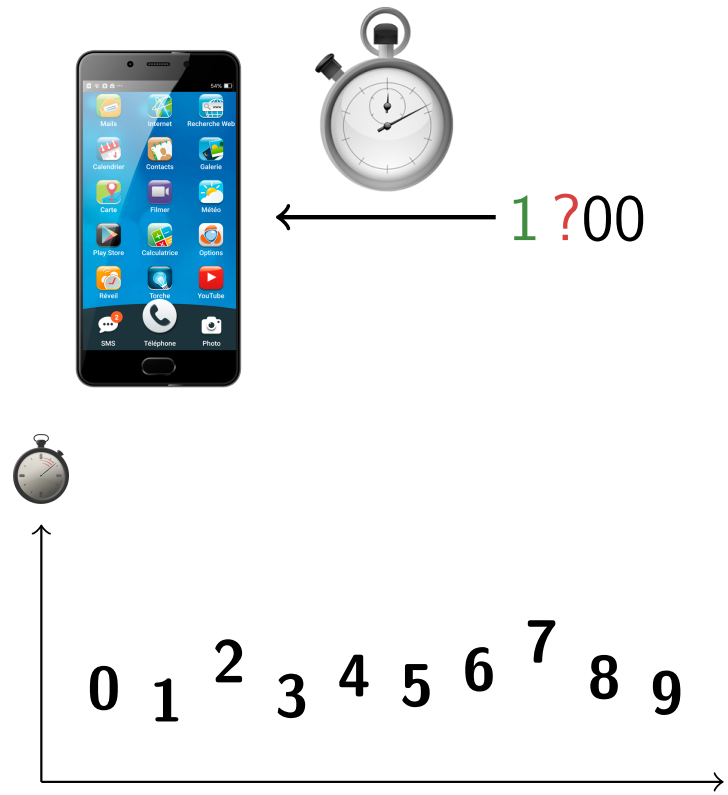
Attaques par canaux auxiliaires

Exemple du code PIN

Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

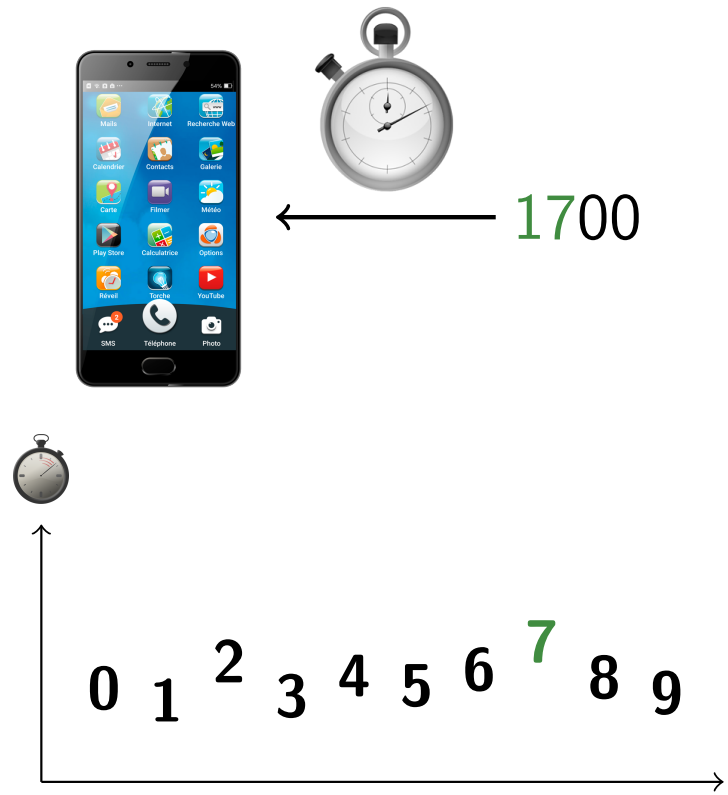
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



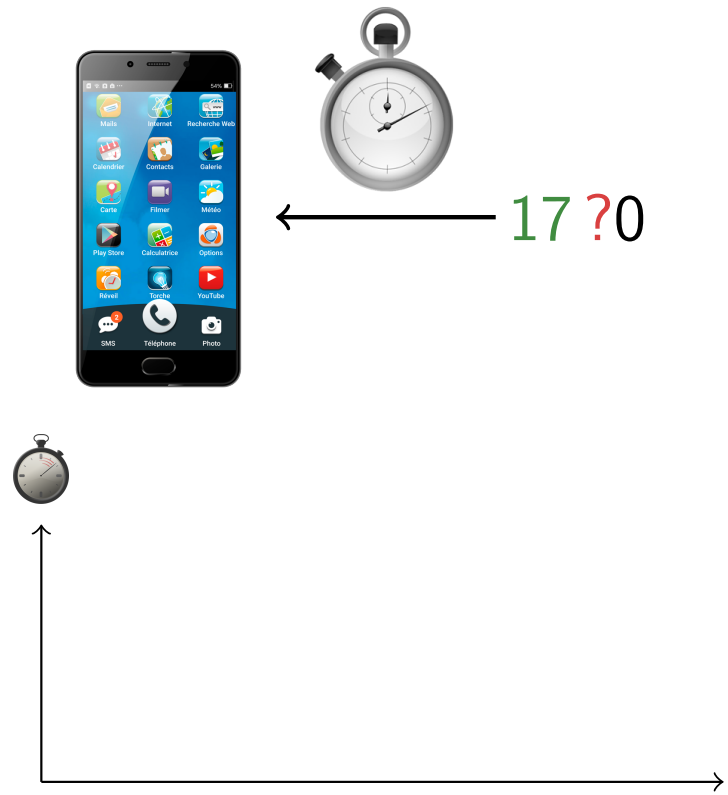
Attaques par canaux auxiliaires

Exemple du code PIN

Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

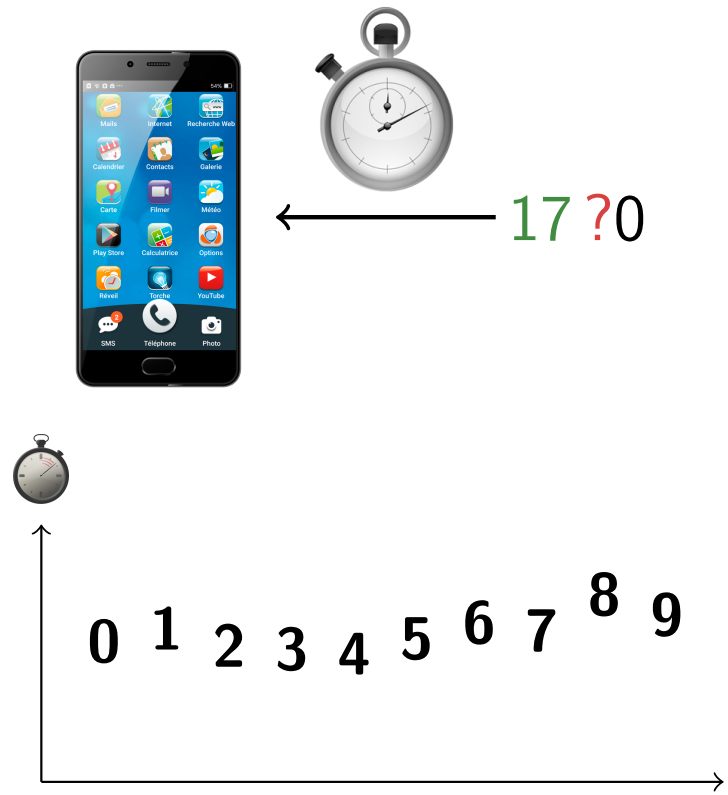
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

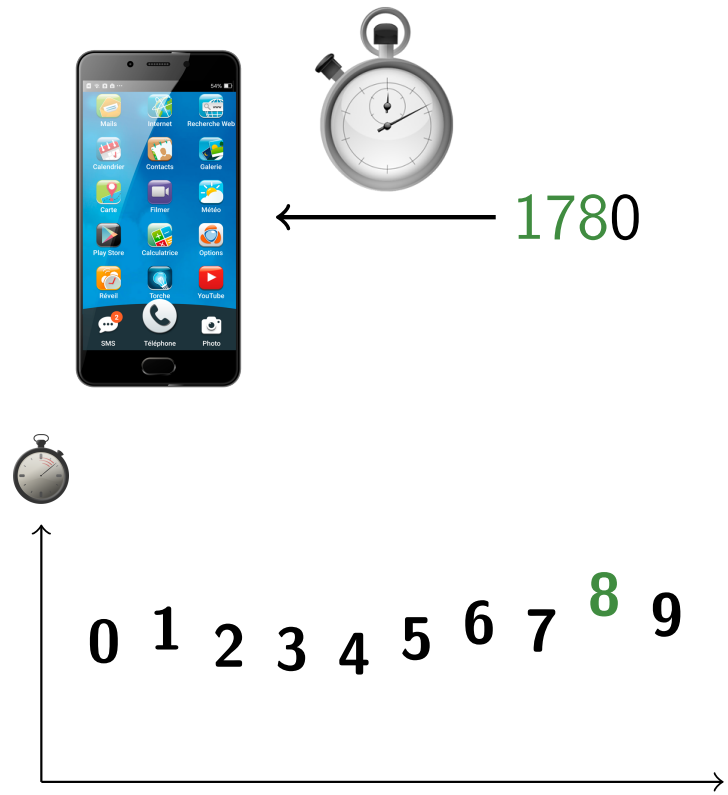
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

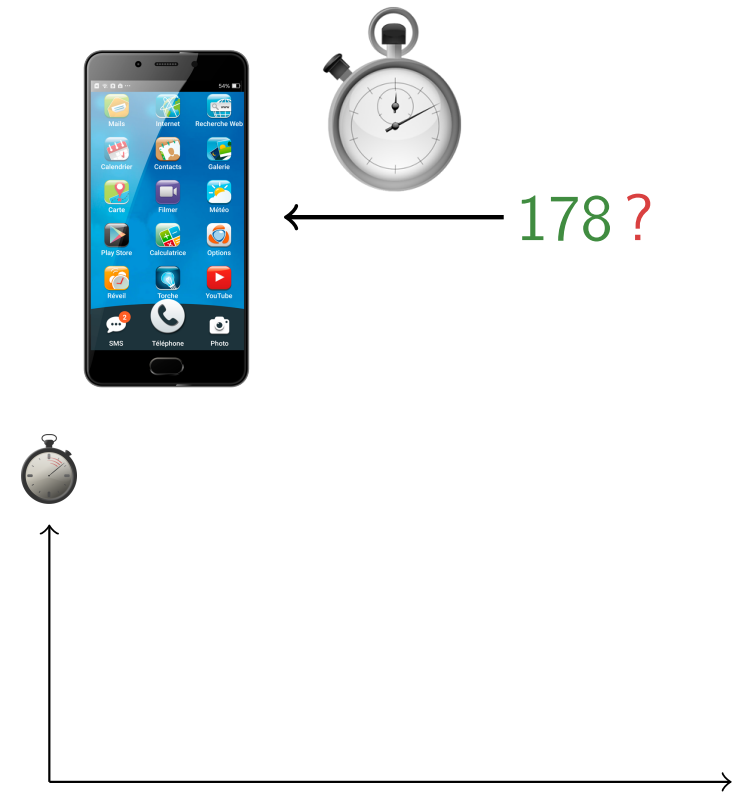
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

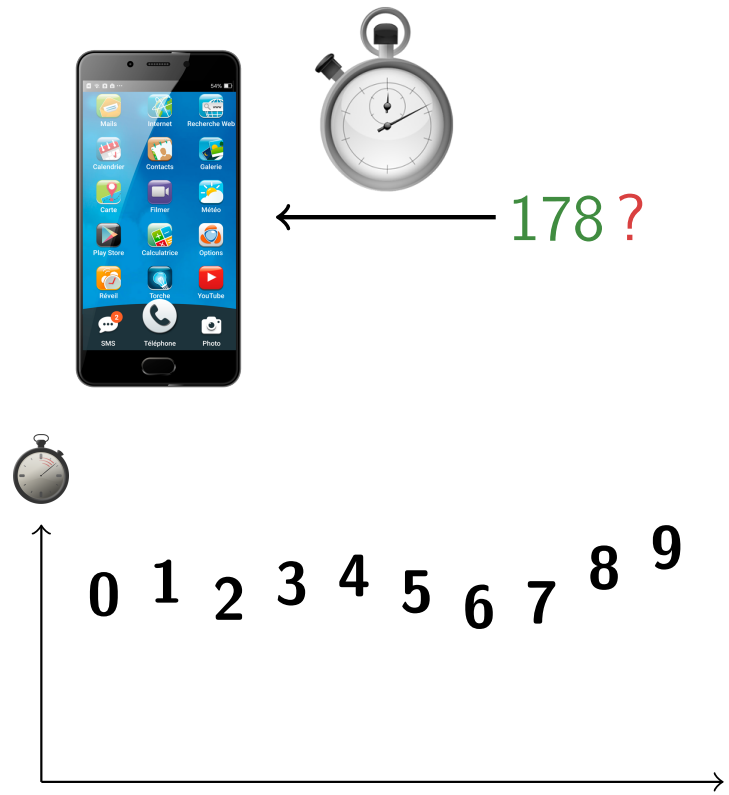
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```



Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

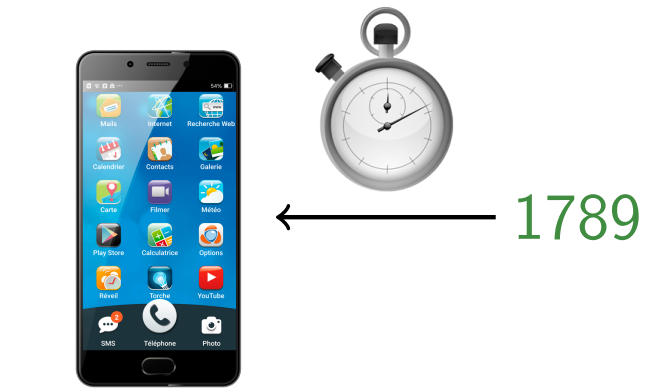
```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```

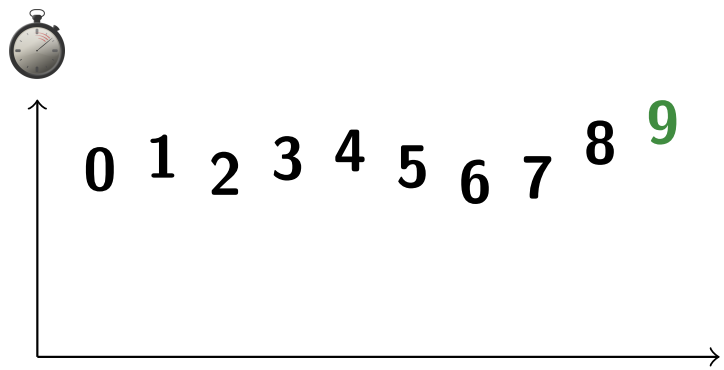


Code PIN de 4 chiffres $\Longrightarrow$ en moyenne 5000 essais.

Si mal implanté $\Longrightarrow$ au plus 37 essais.

```
bool testPIN(int code[4])
{
  for (int i=0 ; i<4 ; i++)
  {
    if (code[i]!=code_ref[i])
      return false;
  }
  return true;
}
```





- Temps d'une multiplication scalaire dans ECDSA
  $\rightarrow$ taille du nonce secret
  $\Rightarrow$ clef retrouvée après 1000 observations.
- Temps de retour d'erreur Mac-then-Encrypt avec padding
  $\rightarrow$ info. sur la correction du padding
  $\Rightarrow$ bloc déchiffré en $2^{15.08}$ requêtes.
- Temps de chiffrement par AES (tables en mémoire)
  $\rightarrow$ lien avec la variable clair $\oplus$ clef
  $\Rightarrow$ clef retrouvée après quelques milliers de chiffrement.
- Toutes les attaques basées sur le mécanisme de cache.
- Toutes les attaques pas inventées car maintenant on fait attention !