



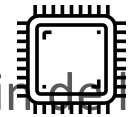
Foreshadow / L1 Terminal Fault

(CVE-2018-3615, CVE-2018-3620 et CVE-2018-3646)

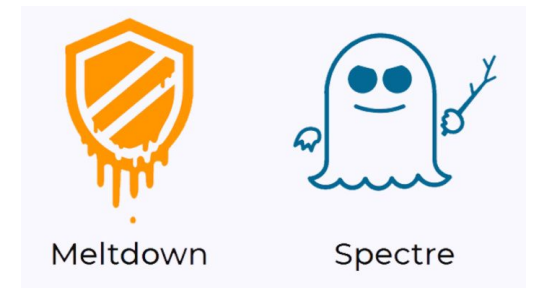
Attaquer les enclaves sécurisées d'Intel

Foreshadow : Contexte



- Attaque sur les parties les plus sécurisées des processeurs Intel
 - Intel SGX permet de créer une enclave sécurisée/ TEE (Trusted Executive Environment), théoriquement sécurisée même si le noyau est compromis
 - Cibles de Foreshadow: Les enclaves sécurisées SGX et les données du cache L1
- Attaque par canal auxiliaire : Exécution spéculative des CPUs 
 - Principe : Commencer à exécuter une instruction avec la fin de l'exécution de la précédente
 - Lorsque les deux instructions sont liées, le processeur peut prédire le résultat de la première pour lancer en avance le deuxième instruction
 - Permet d'augmenter fortement la vitesse de traitement des processeurs, mais les rendent aussi plus vulnérables.
- 3^{ème} faille de ce type en moins d'un an, après Meltdown et Spectre

Meltdown / Spectre :

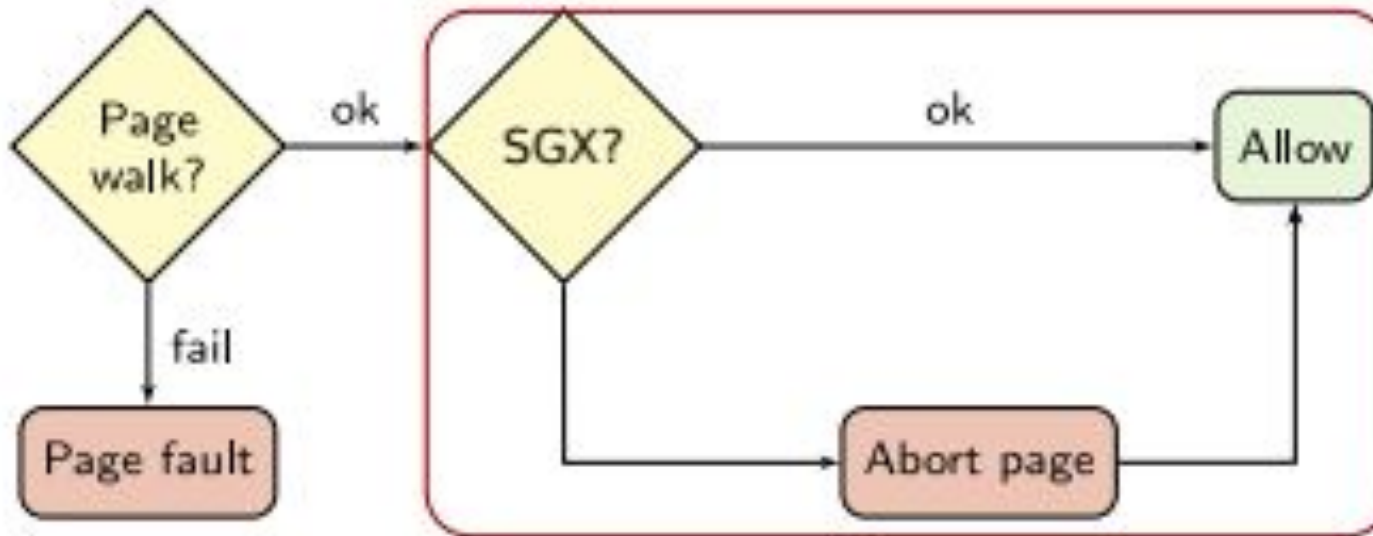


- Meltdown/Spectre s'appuient sur l'exécution spéculative :
 - Génération de requêtes au cache
 - Chronométrage de la réponse afin de déterminer si la requête est un succès ou un échec
 - Meltdown se focalise sur les processeurs Intel qui peuvent spéculer sur des données protégées du système
 - Spectre s'adapte à l'environnement logiciel pour influencer les spéculations exécutives qui vont avoir lieu et trouver un secret dans les caches non protégés.
- **Meltdown est un cas particulier simple à mettre en place et efficace de Spectre.**

	Meltdown	Spectre
Besoins	Pouvoir exécuter du code sur le système ciblé Difficile à faire à distance	Pouvoir exécuter du code sur le système ciblé Peut être fait à distance
Application sur SGX	Non	Oui
Facilité d'exécution	Relativement simple	Plus difficile à exploiter : il faut convaincre le noyau d'exécuter le code, et doit donc être adapté à chaque environnement.
Patches / Protection	Patches existants, comme KPTI et KAISER, qui diminuent un peu la performance des processeurs	Problème de logique du CPU -> pas de patch possible ! Possibilité de patcher les compilateurs pour interdire toute spéculation -> baisse de performance
Processeurs impactés	Intel	Intel, AMD, ARM

Foreshadow : Principe

- Foreshadow : inspiré par les 2 attaques précédentes
 - Attaque concentrée sur SGX
 - Principe : Éviter les abort page en provoquant des défauts de pages



Foreshadow NG

- Foreshadow New Generation
 - Comme toutes les informations du cache L1 peuvent être lues, l'impact de Foreshadow peut être étendu à plusieurs systèmes, notamment :
 - Noyau du système d'exploitation
 - Mémoire du SMM (System Management Mode)
 - Hyperviseur (Virtual Machine Manager)



- Machines virtuelles

Exploitation

- Conséquences :
 - Accès aux données sensibles
 - Création de fausses enclaves sécurisées
 - Disparition de la frontière entre les machines virtuelles

- Applications concrètes possibles :

- Man in the middle :



- Possibilité d'interférer dans la décryption de fichiers -> détournement de cryptomonnaies ...
 - Autoriser l'installation de n'importe quel logiciel
 - Bloquer des sites web

- Accéder aux données de toutes les machines virtuelles hébergées sur l'hôte physique

Problèmes rencontrés pour implémenter l'exploit

- Difficultés pour créer une enclave sécurisée SGX
 - Uniquement sur les processeurs Intel de dernière génération
 - Plateforme de développement à la Windows, sous license et peu documenté
 - Existence de patches qui protègent en partie contre Foreshadow (empêchant une attaque sur un cloud amazon)

Démo

Démo - Bg

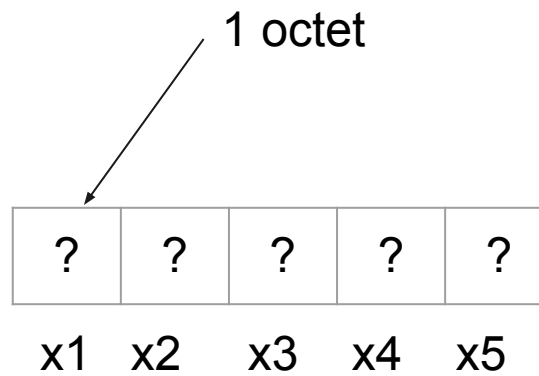
Cache I1

?	?	?	?	?
---	---	---	---	---

255 oracles en RAM

oracle 1
oracle 2
...
oracle 255

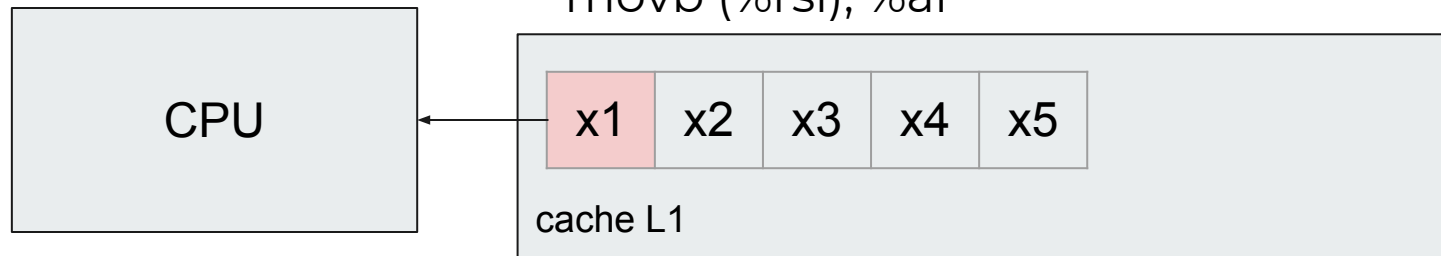
Démo - Bg



Démo - Step 1

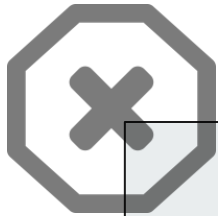
transcient execution :

`movb (%rsi), %al`



Démo - Step 1

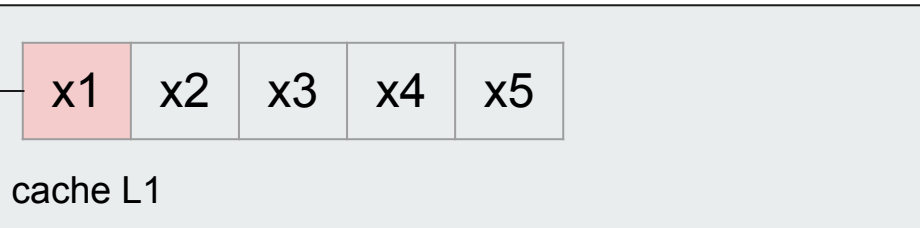
Page Fault



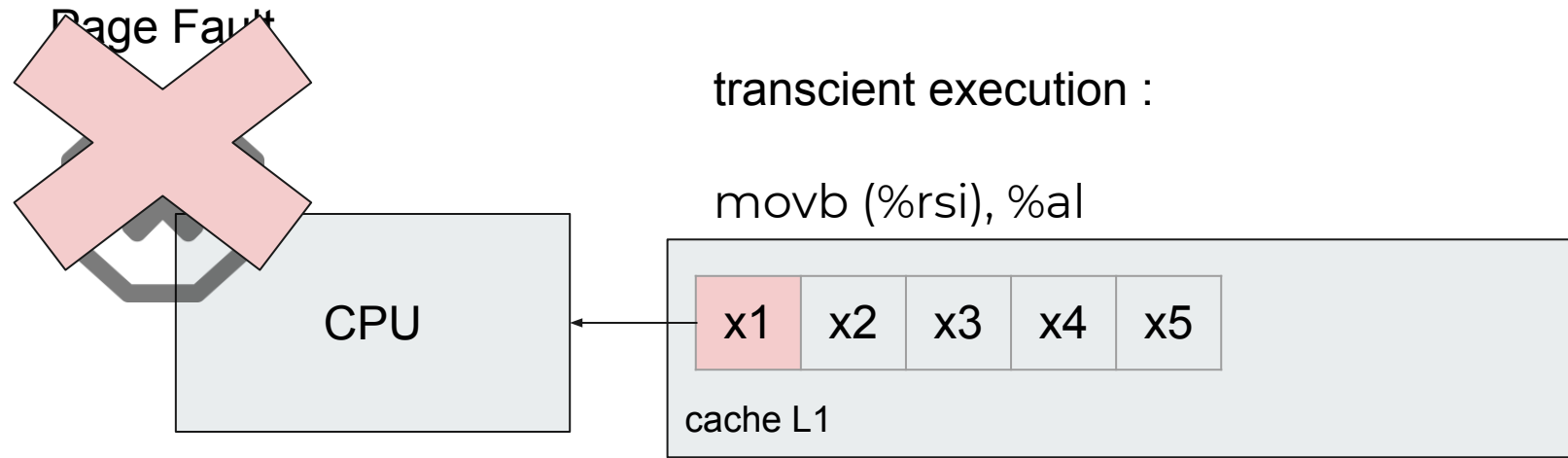
CPU

transcient execution :

`movb (%rsi), %al`



Démo - Step 1

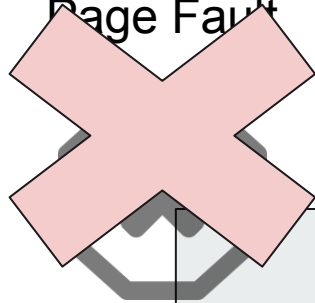


Démo - Step 1

race condition



Page Fault



CPU

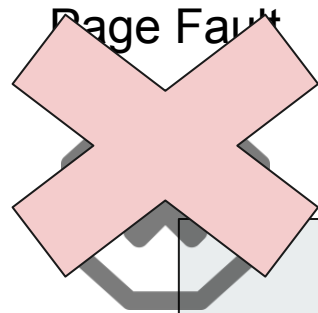
transcient execution :

`movb (%rsi), %al`



Démo - Step 1

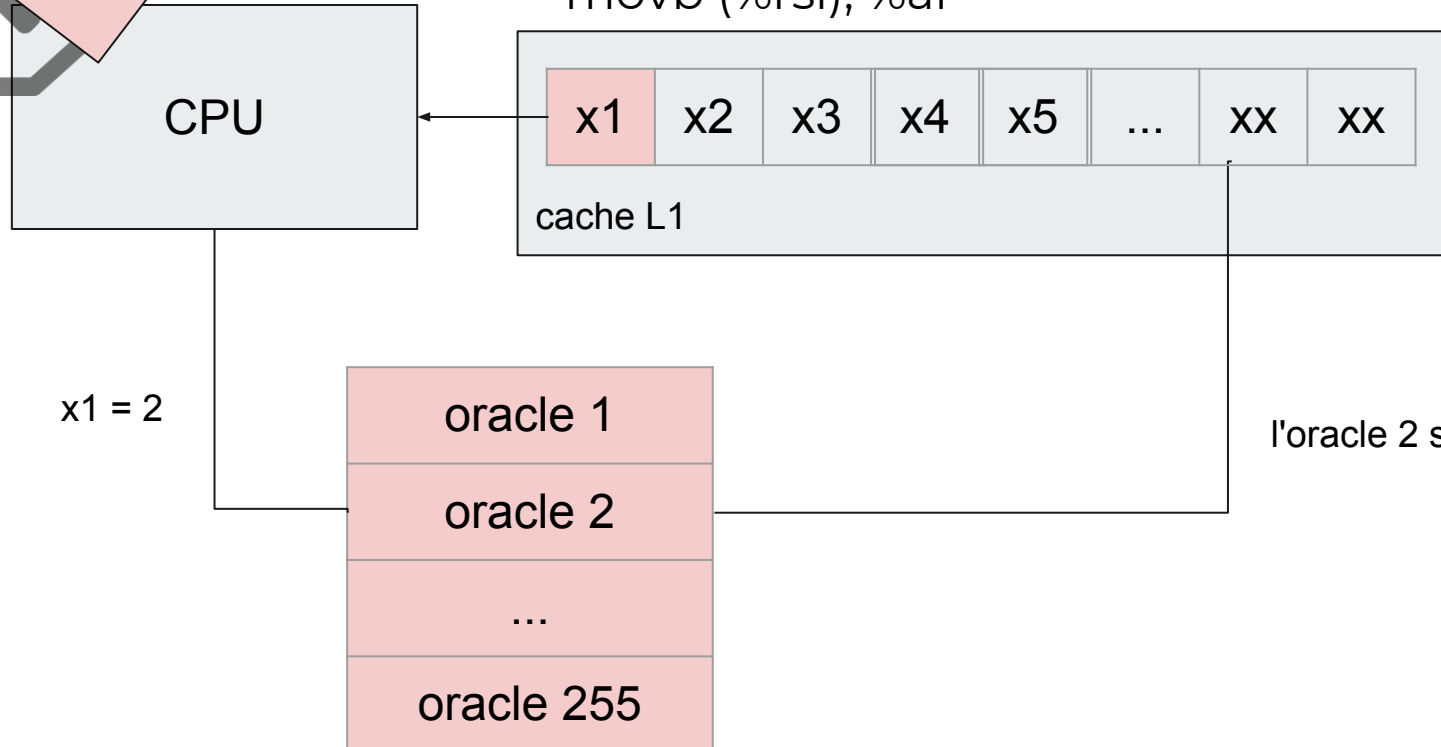
race condition



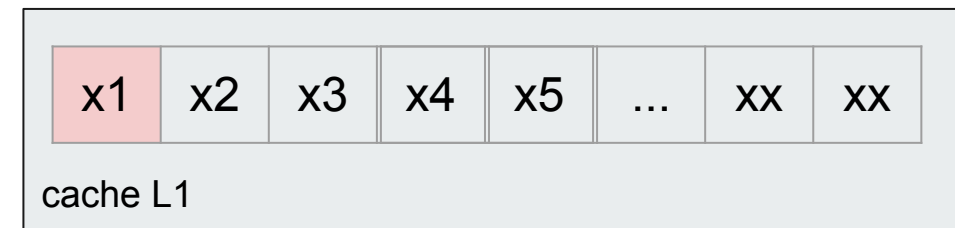
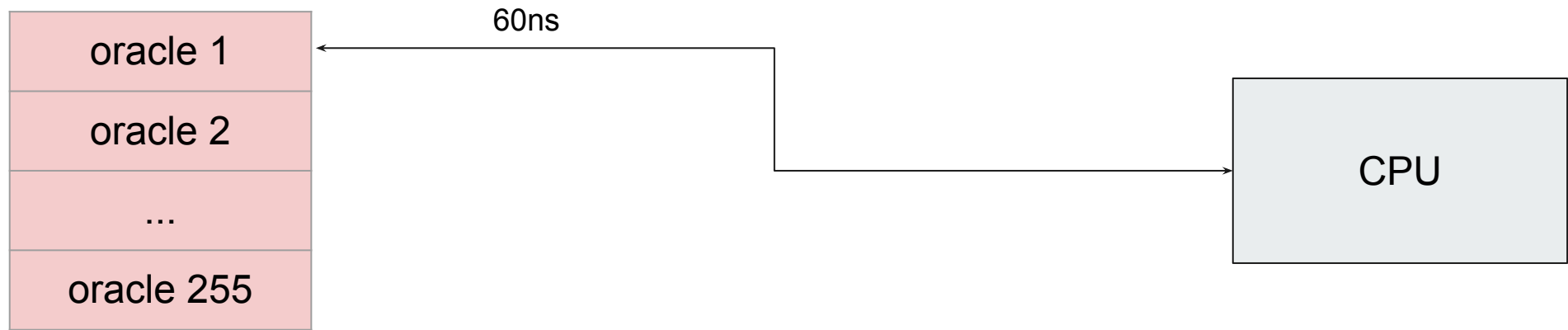
Page Fault

transcient execution :

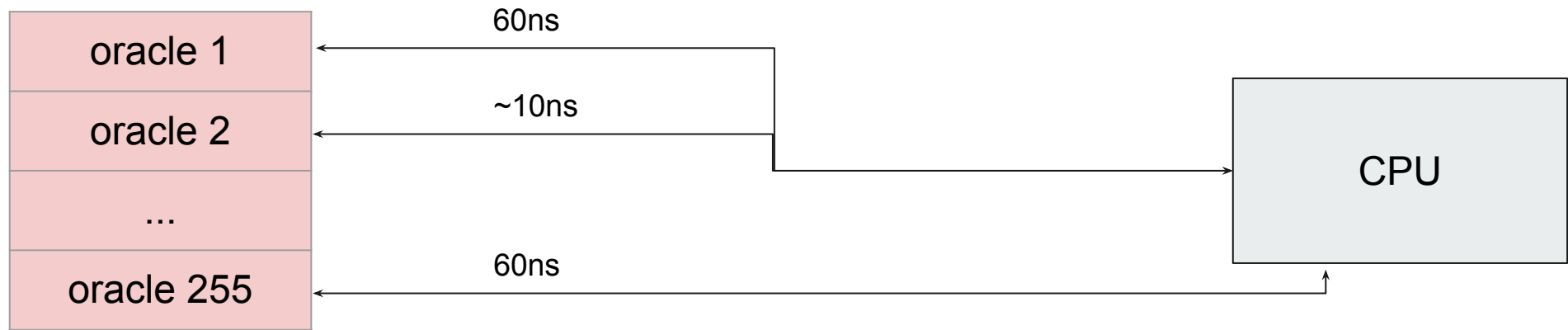
`movb (%rsi), %al`



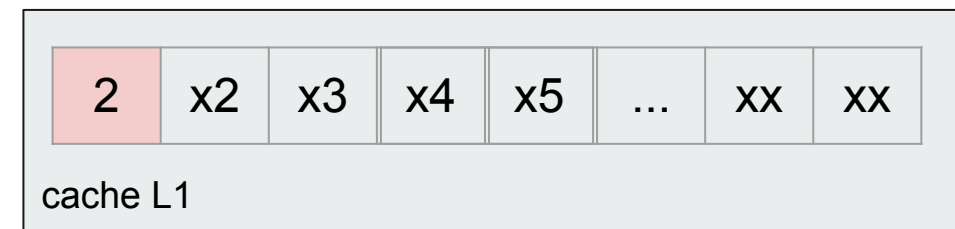
Démo - Step 2



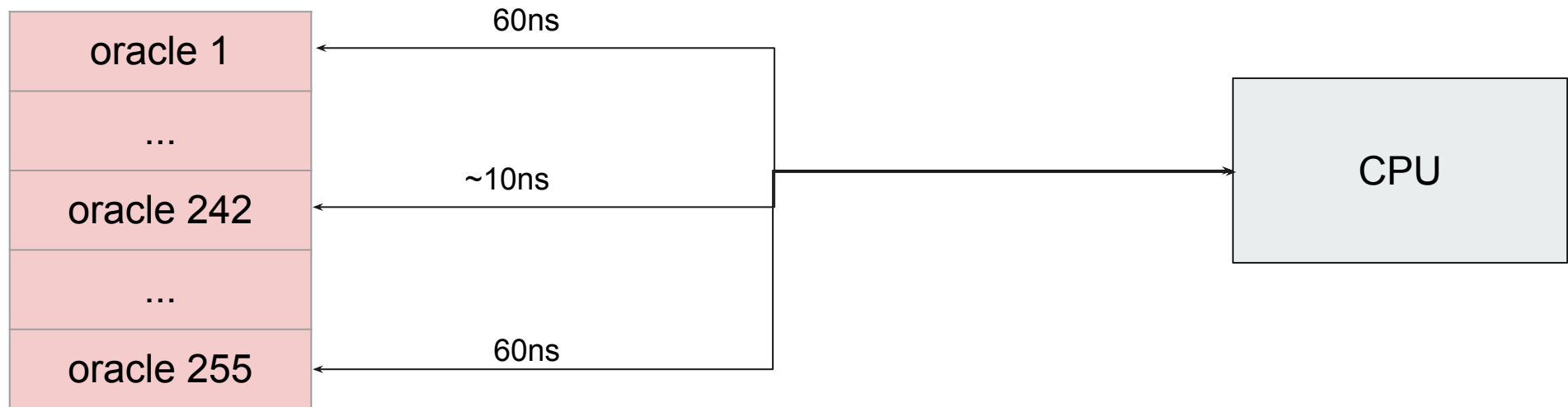
Démo - Step 2



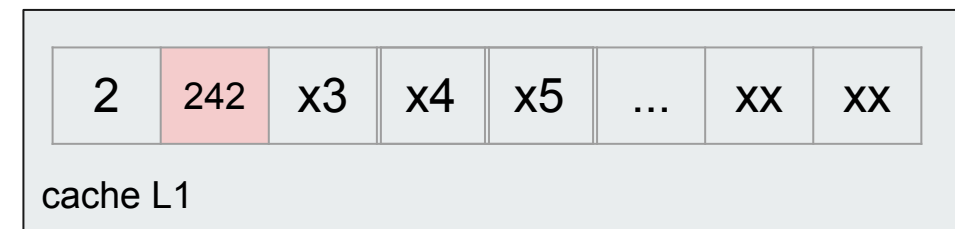
L'oracle 2 est en cache, il dialogue donc beaucoup plus rapidement avec le CPU que les autres. Le premier octet du cache L1 contient la valeur 2.



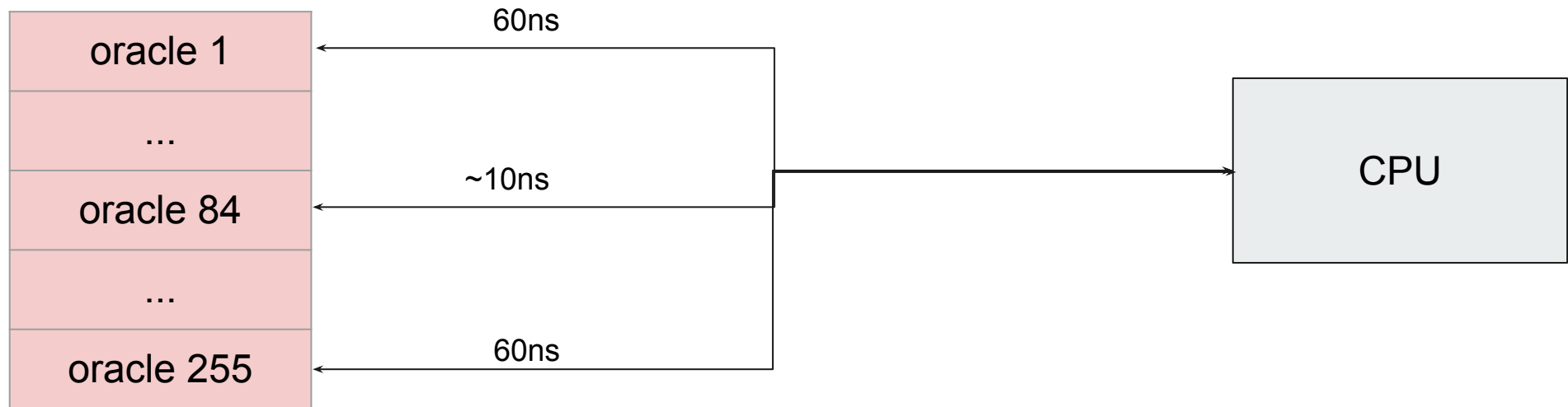
Démo - Step 2



Donc le second octet du cache L1
contient la valeur 242.



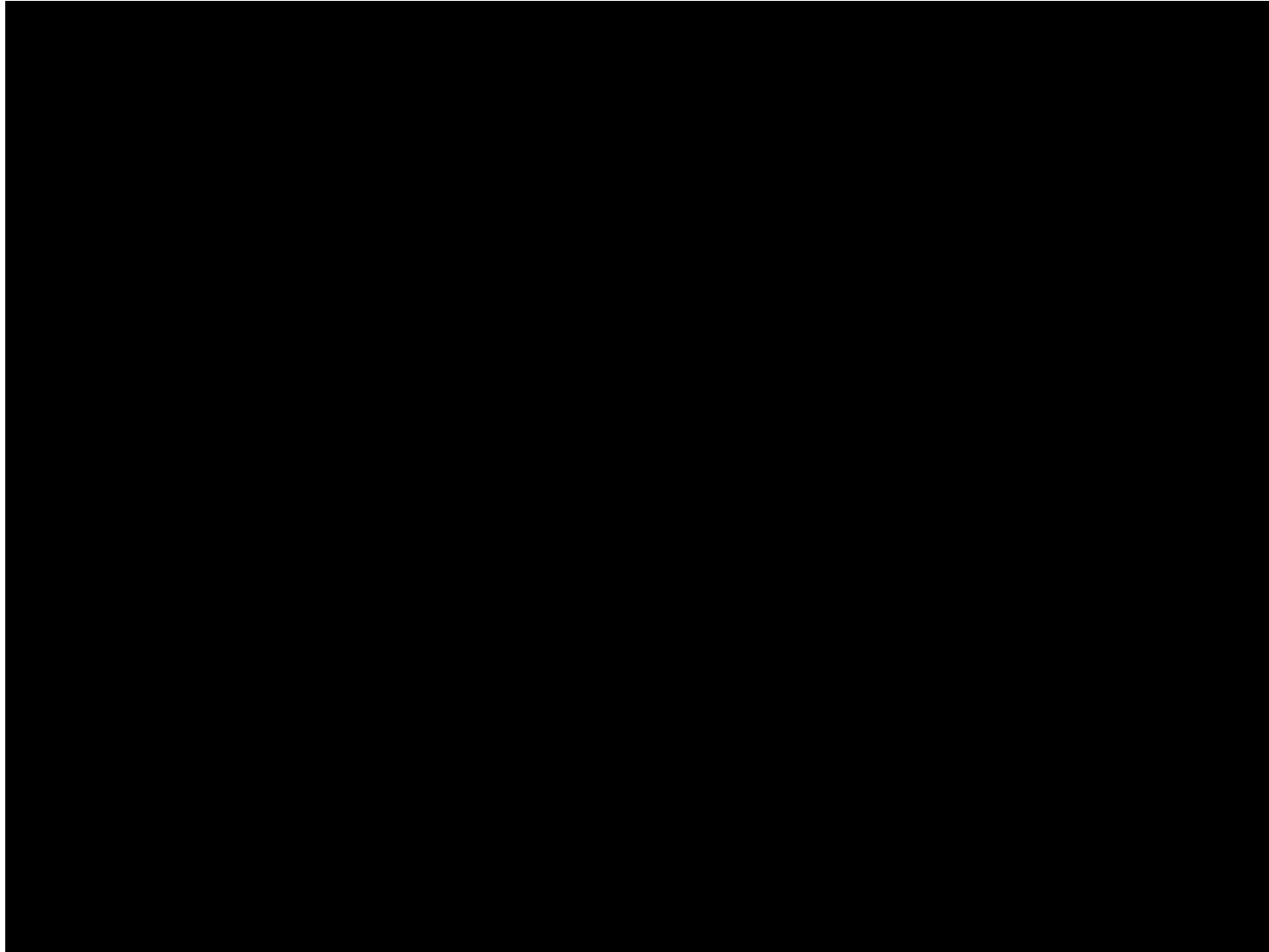
Démo - Step 2



Donc le second octet du cache L1
contient la valeur 84.



Démo



<https://github.com/sebastien-lb/ssi-project>

Annexe :

- Exécution spéculative (des CPUs) :
 - Commencer à exécuter une instruction avec la fin de l'exécution de la précédente
 - Lorsque les deux instructions sont liées, le processeur peut prédire le résultat de la première pour lancer en avance le deuxième instruction
 - Si la prédiction est vraie : gain de temps
 - Sinon : suppression du résultat de la deuxième instruction
- Permet d'augmenter fortement la vitesse de traitement des processeurs, mais le rend aussi vulnérable

Annexe :

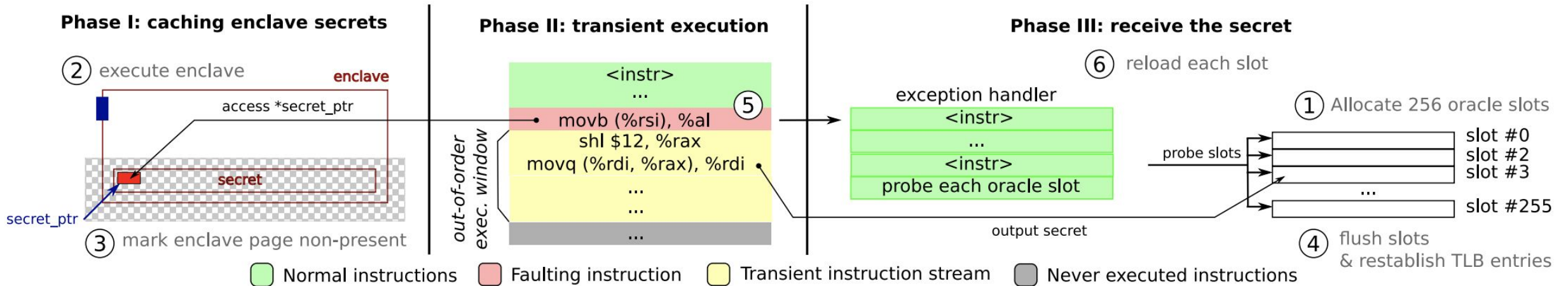


Figure 2: Basic overview of the Foreshadow attack to extract a single byte from an SGX enclave.

Annexe :

- Enclave sécurisée / TEE (Trusted Executive Environment) :
 - Trusted Executive Environment : A trusted execution environment (TEE) is a secure area of a main processor. It guarantees code and data loaded inside to be protected with respect to confidentiality and integrity. A TEE as an isolated execution environment provides security features such as isolated execution, integrity of applications executing with the TEE, along with confidentiality of their assets. In general terms, the TEE offers an execution space that provides a higher level of security than a rich mobile operating system open (mobile OS) and more functionality than a 'secure element' (SE).
 - Isolates a portion of physical memory to protect select code and data from view or modification -> enclaves
 - The key of the protection is a trusted hardware -> even if the kernel is compromised, the enclave should be secured
- Utilisations : Cryptomonnaies, sécurisation de Tor, cloud d'ibm ...

Annexe :

- Meltdown :
- Principe : Apprendre ce qu'il y a dans le cache en mesurant le temps de réponse pour accéder à des données.
- Étapes d'une attaque :
 - Allocation d'une partie de la mémoire hors du cache (M)
 - Lire un octet de l'espace d'adressage du noyau (le secret) -> cette opération sera refusé par le système, mais l'exécution spéculative permet aux autres instructions de se faire en parallèle
 - On multiplie cet octet par la taille d'une page, qu'on utilise pour indexer une partie de la mémoire M
 - On lit chaque octet de M, un octet sera beaucoup plus rapide à lire que les autres car il aura été indexé dans le cache -> on peut en déduire le secret

Sources :

- https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-van_bulck.pdf
- https://www.usenix.org/sites/default/files/conference/protected-files/security18_slides_bulck.pdf
- <https://www.ovh.com/fr/blog/failles-de-securite-spectre-meltdown-explication-3-failles-mesures-correctives-public-averti/>
- <https://nvd.nist.gov/vuln/detail/CVE-2018-3615>
- <https://nvd.nist.gov/vuln/detail/CVE-2018-3620>
- <https://nvd.nist.gov/vuln/detail/CVE-2018-3646>
- https://www.youtube.com/watch?v=n_pa2AisRU8
- <https://foreshadowattack.eu/>