# Sécurisation de programmes assembleur face aux attaques visant les processeurs embarqués - Résumé

Nicolas Moro

Cette thèse s'intéresse à la sécurité des programmes embarqués face aux attaques par injection de fautes. La prolifération des composants embarqués et la simplicité de mise en ?uvre des attaques rendent impérieuse l'élaboration de contre-mesures.

**Mots-clés :** attaques par injection de fautes ; injection électromagnétique ; modèle de fautes ; contre-mesures vérifiées ; assembleur ; saut d'instruction

## 1 Modèle de fautes au niveau assembleur

Un modèle de fautes par l'expérimentation basé sur des attaques par impulsion électromagnétique a été élaboré. Les résultats expérimentaux ont montré que les fautes réalisées étaient dues à la corruption des transferts sur les bus entre la mémoire Flash et le pipeline du processeur. Ces fautes permettent de réaliser des remplacements ou des saut d'instructions ainsi que des modifications de données chargées depuis la mémoire Flash. Une partie des travaux sur la définition du modèle de fautes ont été publiés dans [1].

## 2 Contre-mesure de tolérance aux fautes

Le remplacement d'une instruction par une autre bien spécifique est très difficile à contrôler ; par contre, le saut d'une instruction ciblée a été observé fréquemment, est plus facilement réalisable, et permet de nombreuses attaques simples. Une contre-mesure empêchant ces attaques par saut d'instruction, en remplaçant chaque instruction par une séquence d'instructions, a été construite et vérifiée formellement à l'aide d'outils de model-checking. Un exemple d'utilisation de cette contre-mesure pour l'instruction `bl` d'appel de sous-fonction est présenté dans le listing 1. La contre-mesure proposée a été présentée dans [2] et une version détaillée a été publiée dans [3].

Listing 1 – Séquence de remplacement pour une instruction `bl`

```
1   adr r12, label_retour   ; mise à jour du pointeur de retour
```

```
  adr r12, label_retour
  add lr, r12, #1        ; mise à 1 du dernier bit du pointeur
  add lr, r12, #1        ; (spécification pour le mode Thumb)
  b   fonction            ; branchement vers la sous-fonction
  b   fonction

label_retour              ; destination du pointeur de retour
```

## 3 Test expérimental de la contre-mesure

Cette contre-mesure ne protège cependant pas les chargements de données depuis la mémoire Flash. Elle peut néanmoins être combinée avec une autre contre-mesure au niveau assembleur qui réalise une détection de fautes. Plusieurs expérimentations de ces contre-mesures ont été réalisées, sur des instructions isolées et sur des codes complexes issus d'une implémentation de FreeRTOS. La contre-mesure proposée se révèle être un très bon complément pour cette contre-mesure de détection et permet d'en corriger certains défauts. Les premiers tests expérimentaux sur ces deux contre-mesures ont été publiés dans [4].

## Références bibliographiques


[1] N. Moro, A. Dehbaoui, K. Heydemann, B. Robisson et E. Encrenaz, "Electromagnetic Fault Injection : Towards a Fault Model on a 32-bit Microcontroller", in *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*, Santa Barbara, California, USA : IEEE, août 2013, p. 77–88. DOI : 10.1109/FDTC.2013.9.

[2] K. Heydemann, N. Moro, E. Encrenaz et B. Robisson, "Formal verification of a software countermeasure against instruction skip attacks", in *2nd Workshop on Security Proofs for Embedded Systems (PROOFS)*, Santa Barbara, California, USA, 2013.

[3] N. Moro, K. Heydemann, E. Encrenaz et B. Robisson, "Formal verification of a software countermeasure against instruction skip attacks", *Journal of Cryptographic Engineering*, t. 4, n° 3, p. 145–156, fév. 2014. DOI : 10.1007/s13389-014-0077-7.

[4] N. Moro, K. Heydemann, A. Dehbaoui, B. Robisson et E. Encrenaz, "Experimental evaluation of two software countermeasures against fault attacks", in *2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, Arlington, Virginia, USA : IEEE, mai 2014, p. 112–117. DOI : 10.1109/HST.2014.6855580.