Title: Protecting a software against Control-Flow Integrity attacks

Supervisors: Catherine Parent, Laurent Mounier (Vérimag)  
(e-mail: Prenom.Non@univ-grenoble-alpes.fr)

Fault injection is an attack technique which consists in modifying at runtime the memory content (either the code or the data) of an execution platform. The attacker goal is usually to change the program behaviour in order to get privileged accesses. This attack can be performed for instance using either laser rays, electro-magnetic fields or power glitches. Typical targets are embedded systems like smart cards, or IoT devices.

To protect a platform against this kind of attack, a possible way is to insert counter measures inside the code, namely extra lines of code whose purpose is to ensure that the program behave as expected. One of these counter-measures is called CFI (control-flow integrity) [1]. It consists in verifying at runtime that some program locations are actually accessed according to the initial program semantics (e.g., some part of the code has not been skipped/modified due to an injected fault).

The purpose of this internship is to investigate a code analysis technique based on abstract interpretation to automatically produce a CFI counter-measure. This technique consists in statically computing (using abstract interpretation techniques) some invariant control-flow properties, by mean of “abstract program counters”, that can be checked at runtime on specific program locations.

Two tools developed at Verimag are already available:
• PAGAI [2], allowing to produce control-flow invariants;
• Lazart [3], allowing to simulate fault injections and to check the program robustness.

The objective of this internship is twofold:
1. to propose a suitable approach to achieve CFI using control-flow invariants
2. to experiment (and probably extend) the combination of these two tools in order to evaluate the relevance of this approach on realistic examples.

Biblio

