POLITECNICO DI Torino

Summary

Simone Dutto
Computer Engineering
Master's Degree

Virtual Tool-boxing for Robust Management of Cross-layer Heterogeneity in Complex Cyber-physical Systems

Supervisors: prof. Stefano Di Carlo, prof. Alessandro Savino

Introduction «Error monitoring is a critical procedure for most computing systems, varying from HPC to embedded systems domains.»
Technology scaling (increasing transistors density, low energy consumption requirements) has been discovered being a key factor in raising the rate of errors in hardware devices, specifically the ones called soft errors because they are temporary and caused by external conditions and not material ware-out or design bugs.
These errors are much difficult to prevent or spot because of their nature, so researchers have to find the a way to cope with them.
All of the proposed methods during these years have some limitations: first of all, they require specific hardware supports, they either work at the hardware level directly or exploiting hardware error reporting. This is, indeed, the safest way to detect errors but it relies on hardware manufacturers to add reporting mechanisms for this or that specific error. Also, hardware techniques are not process-wise, because they keep tracks agnostically of errors in memory structures. Exactly for these reasons, it is interesting to explore faults detection using hardware metrics monitoring (not error reporting) and

machine learning, to enable flexibility and precision trying to keep the mechanism as light as possible both hardware and software-wise.

Goal The ultimate goal of this thesis is to discuss what can be done in terms of machine learning models to detect faults and finally to give a primal implementation of a tool to monitor processes in a Linux environment. Figure 1 represents the framework for a monitoring tool working with hardware inspection kernel-side and machine learning evaluation user-level.

Thesis breakdown The thesis is composed of 3 key chapters:

- *Simulation environment:* in this chapter will be explained how the simulated system is created and how it is handled by the simulator, gem5, implementation-wise. Later the FIMSIM, the fault injector, functionalities will be inspected and explained alongside the different types of faults and their respective effects on the system. Finally, the metrics generated from different simulations are explained in the context of model creation.
- *Data Analysis and Models:* at the start it is explained how data are aggregated and cleaned. After that, different features selection techniques are analyzed. Then, a review of related works on fault detection is investigated and finally, each selected model is tested and performances are compared.
- *OS implementation:* the third chapter is about implementing the whole system on a current Linux distribution. Different options are analyzed and at the end it is explained how the system, Figure 1, is organized to expand an already implemented metric extractor to evaluate a process to decide whether or not there has been a fault during its execution.

Conclusion and Future Works The whole project is a demonstration of how powerful can be to use the potentialities of machine learning in a fault detection system built upon hardware metrics. In fact, the final framework interacts minimally with the hardware and the best performing model is the simplest.
To continue this work, it would be essential to add support for other metrics in the hardware extractor module. After that, a path would be to run the patched operating system inside the simulator. This would give an idea of

how the system can detect injected faults having the right pre-trained models.
Ultimately, it would be interesting to collect more data and analyze different models and how they interact with different fault types and results.

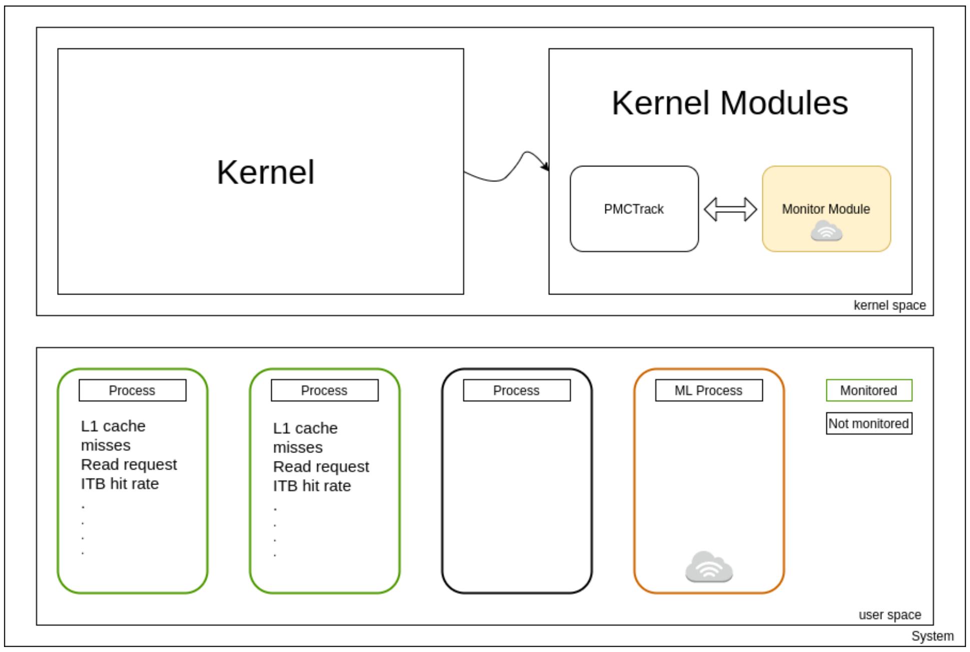

Figure 1. Final system: the monitor kernel module is used to handle the whole fault detection, the PMCTrack module to extract the metrics then forwarded to a Machine Learning process running inside (or outside) the machine that gives the evaluation back to the monitor module.