PHMon: A Programmable Hardware Monitor and Its Security Use Cases

Leila Delshadtehrani, Sadullah Canakci, Boyou Zhou, Schuyler Eldridge, Ajay Joshi, and Manuel Egele

delshad@bu.edu

Boston University

August 12, 2020

Current Trend

Growing demand to enforce security policies in hardware

Current Trend

Growing demand to enforce security policies in hardware

[AUSTIN, PLDI'94] [DEVIETTI, ASPLOS'08] [NAGARAKATTE, PLDI'09]

Current Trend

Growing demand to enforce security policies in hardware

PHMon

A hardware monitor and the full software stack around it

- A programmable hardware monitor interfaced with a RISC-V Rocket processor on an FPGA
- OS support
- Software API
- Security use cases

How Does It Work?

A user/admin con gures the hardware monitor

The hardware monitor collects the runtime execution information

- Checks for the speci ed events, e.g., detects branch instructions
- Performs follow-up actions, e.g., an ALU operation

HW Functionality

- Collect the instruction trace
- Find matches with programmed events
- Take follow-up actions

Software Interface

- A list of functions that use RISC-V's standard ISA extension
  - Con gure PHMon
  - Communicate with PHMon

OS Support

- Per process OS support
  - Maintain PHMon information during context switches
- Interrupt handling OS support
  - Delegate interrupt to OS
  - Terminate the violating process

Implementation

- PHMon as a RoCC, written in Chisel HDL
  - Interfaced with the in-order RISC-V Rocket core
- Linux kernel v4.15
- RISC-V gnu toolchain for cross-compilation

Evaluation

- Prototyped on Xilinx Zynq Zedboard
  - Rocket core + PHMon
- Open-sourced at https://github.com/bu-icsg/PHMon

PHMon-based Shadow Stack

- Simple and exible
  - Two MUs
  - Shared memory space
    - Allocated by OS as a user-space memory
- Secure
- E cient
  - For SPECint2000, SPECint2006, and MiBench benchmarks, on average, 0.9% performance overhead

American Fuzzy Lop (AFL)
[Zalewski, 2013]

- A state-of-the-art fuzzer
- Two main units
  - The fuzzing logic
  - The instrumentation suite
    - Compiler-based
    - QEMU-based

https://rabbitbreeders.us/american-fuzzy-lop-rabbits/

American Fuzzy Lop (AFL)
[Zalewski, 2013]

A state-of-the-art fuzzer

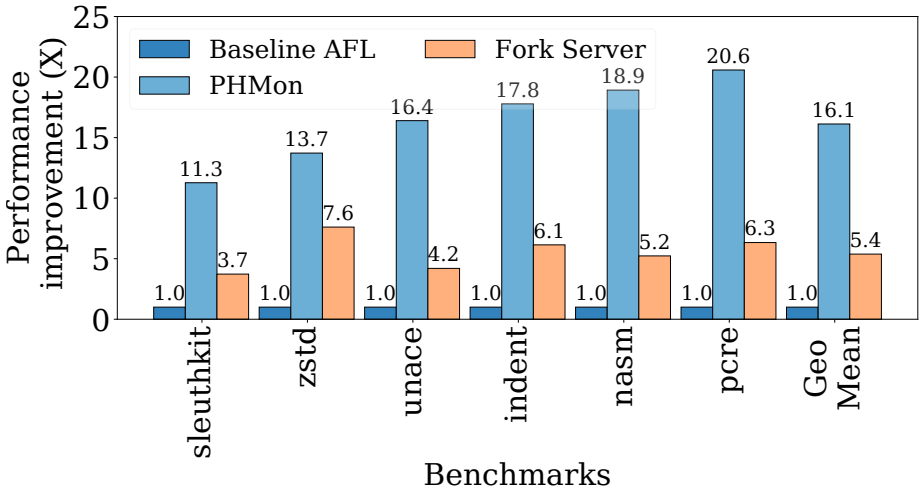
Two main units

- The fuzzing logic
- The instrumentation suite
  - Compiler-based
  - **QEMU-based**

QEMU-based AFL

PHMon-based AFL

https://rabbitbreeders.us/american-fuzzy-lop-rabbits/



QEMU-based AFL

PHMon improves AFL's performance by 16× over the baseline

Power overhead: 5%

Area overhead: 13.5%

PHMon-based AFL