ICRI-CARS, CloSer, SELIot
Secure Systems Group, Aalto University

Research Institute for Collaborative Autonomous and Resilient Systems

Thomas Nyman, Ghada Dessouky, Shaza Zeitouni, Aaro Lehikoinen
Andrew Paverd, N. Asokan, Ahmad-Reza Sadeghi

HardScope: Protecting Embedded Systems against Data-oriented attacks

How can variable visibility rules be enforced at run-time to prevent run-time attacks?

We present HardScope, a novel hardware extension for Run-time Scope Enforcement in embedded systems

Motivation

- Variable visibility rules make it less likely to reference unintended variables
- Run-time attacks violate assumptions about what data is referenced at compile time vs. run-time
- Mechanisms for variable scope enforcement at run-time can significantly reduce potential of run-time attacks

Challenges

- Dynamic scope ≠ lexical scope: variable visiblity information not typically available at run-time
- Granularity of enforcement: effective compartmentalization requires fine granularity for subjects (code) and objects (data)
- Context sensitive access: same piece of code may operate under different set of rules depending on where it is called from
- Pervasiveness: efficiently mediate all memory accesses

High-level ideal

Instrument program code during compilation to

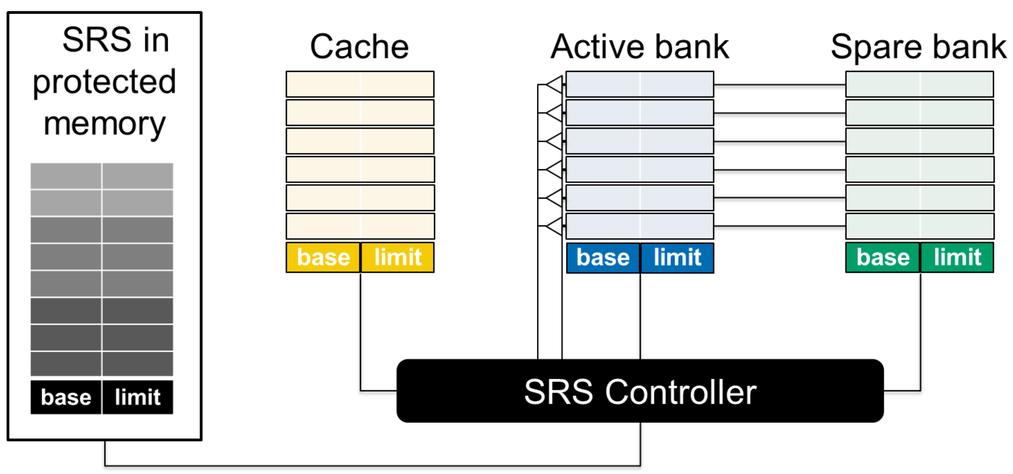
- split code up into distinct execution contexts, i.e. the 'environment' of a piece of code, e.g. function instance
- associate each execution context with storage regions, i.e. portion of data memory accessed in the execution context

Modify underlying hardware with HardScope instructions to:

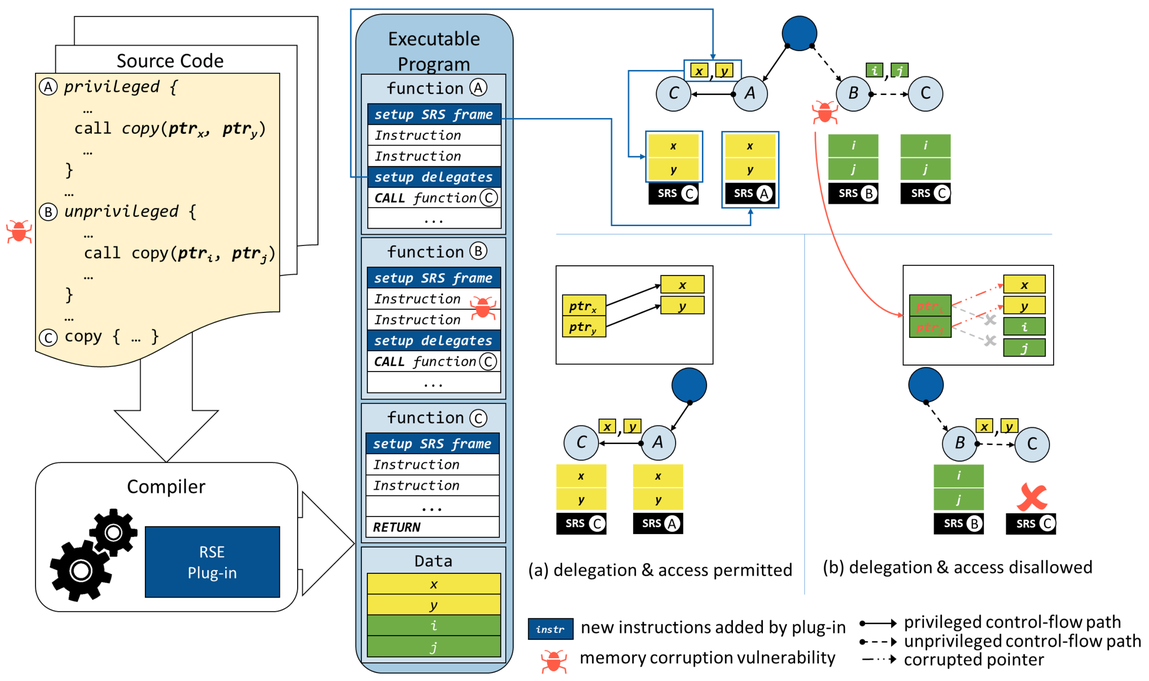
- accumulate rules for storage regions associated with the current execution context [new storage region instructions]
- track changes of execution context at run-time [new scope block instructions]
- treat new code activations as separate execution contexts, and track dynamic data [new data delegation instructions]
- enforce that each execution context only accesses memory in its storage regions [modified load / store instructions]

Storage Region Stack

- Enables enforcement without slowing down loads / stores as active rules stored at top of stack and cached for fast access
- Overhead from cache management amortized over several instructions on execution context change



HardScope hardware organization



High-level overview of HardScope instrumentation & operation

HardScope PoC integrated in RISC-V PULPino SoC on FPGA

- Hardware-component for managing run-time access rules
- Six new instructions added to RISC-V instruction set
- Compiler plug-in that instruments software for HardScope

Benefits of HardScope

- Provides resilience against multiple classes of attacks, e.g. ROP, DOP
- Granularity of enforcement adjustable, e.g. module-, function-, code-block- compartmentalization
- Low-overhead, only ~3.2% for function granularity enforcement in CoreMark embedded benchmark

Aalto University

TECHNISCHE UNIVERSITÄT DARMSTADT

https://github.com/runtime-scope-enforcement

https://arxiv.org/abs/1705.10295