MOSE: Automated Detection of Module-Specific Semantic Errors

University of Minnesota

Challenges:

- Modern systems: Large and complex
- Highly diverse and customized
- How to automatically generate error-checking rules for module-specific cases?

Solution:

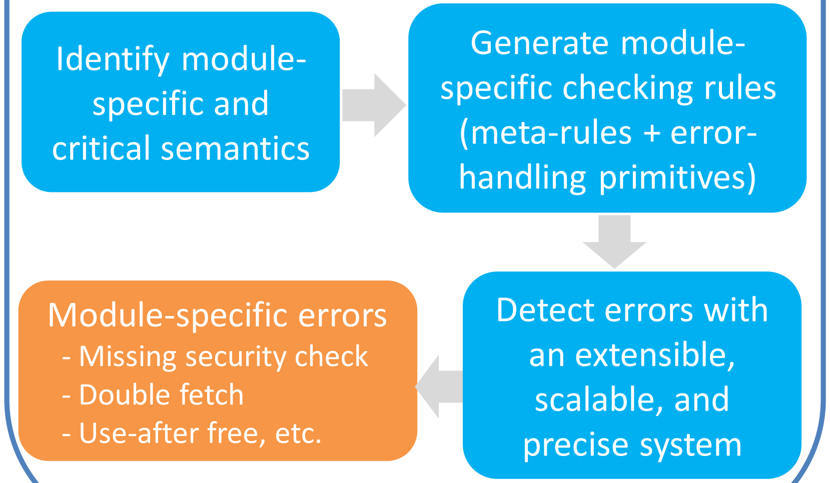
- Develop and instantiate general "meta-rules"
- Infer critical semantics and rules based on error-handling primitives

#1815621,
Kangjie Lu (kjlu@umn.edu) and
Stephen McCamant (mccamant@cs.umn.edu),
University of Minnesota-Twin Cities

Large and complex programs

- C/C++/assembly
- >25M SLoC
- >20K diverse modules
- >10 fixes/hour
- Billions of users
- Security-critical

Framework overview



Identify module-specific and critical semantics

Generate module-specific checking rules (meta-rules + error-handling primitives)

Detect errors with an extensible, scalable, and precise system

Module-specific errors
- Missing security check
- Double fetch
- Use-after free, etc.

Scientific Impact:

- Semantic error is the major source of vulnerabilities
- Automatically detects module-specific semantic errors which are often missed by prior approaches
- Techniques such as criticality inference, and staged symbolic execution are generic to future research

Broader Impact:

- Have found and fixed >700 new bugs in widely used OS kernels (Linux, Android, etc.)
- Have open-sourced two projects; triggered some general changes in Linux
- Reported findings at CCS'19, USENIX Security'19, etc.
- Findings are integrated in OS and Security courses at UMN