White-box Software Isolation with Fully Automated Black-Box Proofs

Jiaqi Tan, Rajeev Gandhi, Priya Narasimhan
Dept. Of Electrical & Computer Engineering, Carnegie Mellon University

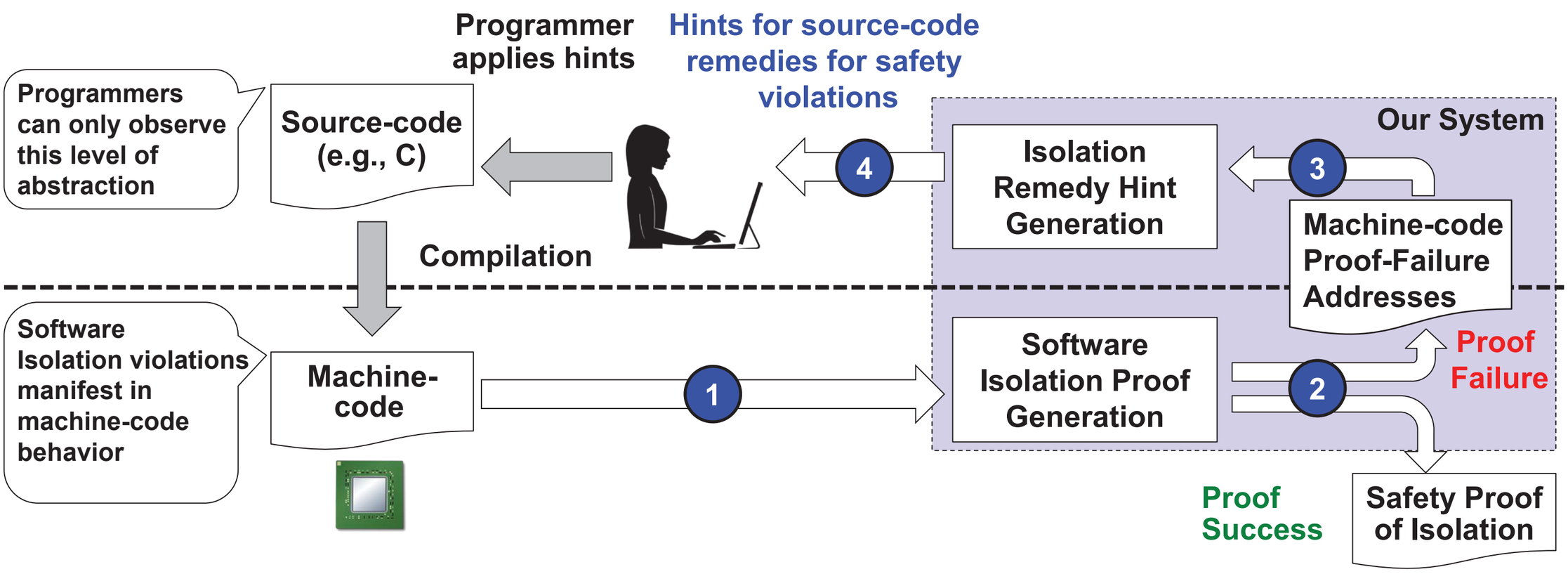
Motivation

- **Software Isolation: No new or unintended behaviors can be introduced via external inputs**
- **Software Isolation important for:**
  - **› Safely running untrusted code in trusted host, e.g., running ad hoc crowd-sourced code**
  - **› Preventing subversion of critical systems, e.g., medical devices, avionics systems**
- **Since Software Isolation is important, we want:**
  - **› Strong evidence that critical software is isolated**
  - **› Programmer-evident isolation mechanisms**
- **Our Approach:**
  - **› Formal verification: Provides strong evidence**
  - **› Isolation mechanisms: Must be in source-code**

Goals / Scope

- **Goals:**
  - **› Enable white-box software isolation: No post-compilation modifications required**
  - **› Enable black-box software isolation proofs:**
    - **Fully automated proofs**
    - **No specialized inputs, e.g., loop invariants, function pre-/post-conditions**
- **Scope: ARM machine-code programs**
  - **› ARM: Dominant mobile/embedded platform: Many critical applications**
  - **› Machine-code: Minimizes Trusted Computing Base (TCB): Excludes compiler from TCB**

Architecture



Design and Implementation

- **Software Isolation: Memory & Control-flow Safety**
  - **› Programmer-evident software isolation: Isolation mechanisms are programmer-visible**

1 2 Automated Safety Property Verification

- **› Input: Machine-code of program**
- **› Output: Safety proof, or proof failure addresses**
- **Approach:**
  - **› Extends Hoare Logic for ARM machine-code [Myreen'07] to reason about safety properties**
  - **› Abstract Interpretation: Automated proof obligation discharge with failure termination**
  - **› Developed logic framework: AUSPICE [Tan'15]**
  - **› Implemented in HOL4 theorem prover**

3 4 Isolation-Remedy Hint-Generation

- **› Input: Machine-code Proof Failure Addresses and source-code of program**
- **› Output: Source-code hints to remediate isolation failures**
- **Approach:**
  - **› Walk Abstract Syntax Tree (AST) of source line to extract offending expression**
  - **› Generate hints for source-code to repair isolation violation at offending location**
  - **› Uses LLVM-Clang compiler front-end**
- **Programmer applies hints to source-code, taking into account program semantics, and recompiles program**

[Tan'15] J. Tan, H. Tay, R. Gandhi, and P. Narasimhan. AUSPICE: Automated Safety Property Verification for Unmodified Executables. In Working Conference on Verified Software: Tools, Theories and Experiments (VSTTE). Jul 2015.
[Myreen'07] M. Myreen, A. Fox, and M. Gordon. Hoare Logic for ARM Machine Code. In Fundamentals of Software Engineering (FSEN), 2007.

Carnegie Mellon University

Parallel Data Laboratory