Formal Methods for Security

Breakout 5

Adam Chlipala (MIT)   Deian Stefan (UCSD)

Topic: Formal methods (broadly)

Why is it important to society? to a secure and trustworthy cyberspace?

1. Principled way to build secure and trustworthy systems

   ⇒ Eliminate whole classes of bugs vs. hand-to-hand combat

1. Formal thinking is an opportunity to harden a design

   ⇒ Modular and simple design is key to formal reasoning and security

1. Good way to measure security

   ⇒ Need to be be explicit about attacker model and edge cases

Highlights of existing body of work and adoption

1. Mainstream principled programming languages: Rust, WebAssembly
2. All major browsers and mobile platforms use formally verified crypto code
3. TLS 1.3 was hardened by FM: semantics eliminated classes of design bugs
4. Lots of (and increasing) adoption of FM for security in industry:
    a. Amazon: verification for access control
    b. Facebook: static analysis (Infer) to find and eliminate different class of bugs
    c. Microsoft: formal verification of network virtualization parsers (EverParse)
    d. Mozilla: runtime and static verification of JavaScript compilation
    e. Fastly/Bytecode Alliance: formal verification of WebAssembly
    f. ARM: formal semantics of ISAs
    g. Certora, CertiK: static analysis and formal verification to check smart contracts

Challenges

1. Education
   a. Newcomers find the area daunting: breadth of tools, all the tips and tricks
   b. Software industry broadly unfamiliar with the logical/mathematical foundations
2. Scope
   a. Lot of work on functional correctness, increasingly more work on security
   b. Need to broaden scope to privacy, high-level policies (e.g., GDPR)
3. Scalability
   a. Tool design for usability and algorithmic improvements to speed up execution
4. Tooling and usability
   a. Connections across tools, languages, layers, and interfaces (hardware, software, network)
   b. Many tools are still low-level
   c. Need more “in the field” collaborations (e.g., between FM and systems security)

Future directions and collaboration needs

1. Need for introductory resources for the broad area of formal methods
    a. Tutorials at security conferences?
    b. University curriculum (teaching “what is correctness?”)
2. Many technical directions
    a. Improving scalability in different dimensions
    b. Compositional methods, techniques for incremental deployment
    c. Cross-language, cross-layer and extensible analysis
3. Need collaborations across areas
    a. Integrate FM into software-engineering tools and workflows
    b. Need to design new (operating, distributed, runtime, etc.) systems with FM from the start

What else?

Hardware is changing. How we program is changing. Industry is adopting FM.

We have an opportunity to (re)build the new generation systems to be principled from the start.