What is secure compilation?

ătălin Hrițcu
What is secure compilation?

Secure compilation aims to preserve high-level language abstractions in compiled code, even against adversarial low-level contexts.

- Secure compilation more than full abstraction

Fully abstract compilation et al fit this intuition
Show 2 other properties fitting this intuition

Secure compilation more than full abstraction
Side-channel attacks

Very powerful attacker (but realistic!)

Achieving full abstraction against such a powerful low-level attacker seems hopeless

level contexts can’t observe low
### What can we do?

**Option 0:**
- deny/ignore/postpone the problem, stick with full abstraction and weak attackers

**Option 1:**
- defend against side-channel attacks

**Option 2:**
- devise weaker secure compilation properties that are immune to side-channels

I’ll focus on these, but there might be more options.
Option 1: defend against side-channels

Hopeless:

t
two arbitrary programs

More realistic:

Property:

t
t existing

Limited scope:
Option 2: devise weaker property that is immune to side-channels

Hopeless: observational equivalence

Property: robust compilation

- preservation of safety in adversarial context (robust safety)
- conjectures:
  - \( tt \)
  - less extensional, \( tt \) achievable, still useful
  - data invariants, integrity properties
broad view on secure compilation

Different security goals / attacker models

- constant time preservation, robust compilation, ...

Different enforcement mechanisms

Different proof techniques