Lecture 1: What is Secure Compilation?

CS350

Marco Patrignani

Practicalities

- Course hours: WF 9:30 - 11:20

Practicalities

- Course hours: WF 9:30 - 11:20
- Cancelled lectures

Practicalities

- Course hours: WF 9:30 - 11:20
- Cancelled lectures
- Type of course: lectures + presentations

Practicalities

- Course hours: WF 9:30 - 11:20
- Cancelled lectures
- Type of course: lectures + presentations
- Course goal:
  - present background and motivation behind SC
  - discuss formal techniques for SC
  - discuss the most recent developments in SC

Practicalities

- Course hours: WF 9:30 - 11:20
- Cancelled lectures
- Type of course: lectures + presentations
- Course goal:
    - present background and motivation behind SC
    - discuss formal techniques for SC
    - discuss the most recent developments in SC
- Evaluation: presentations, reports. (side projects are also an option)

Practicalities

- Pose questions

Practicalities

- Pose questions
- SC is a very active research field with many unsolved difficult problems to work on

Practicalities

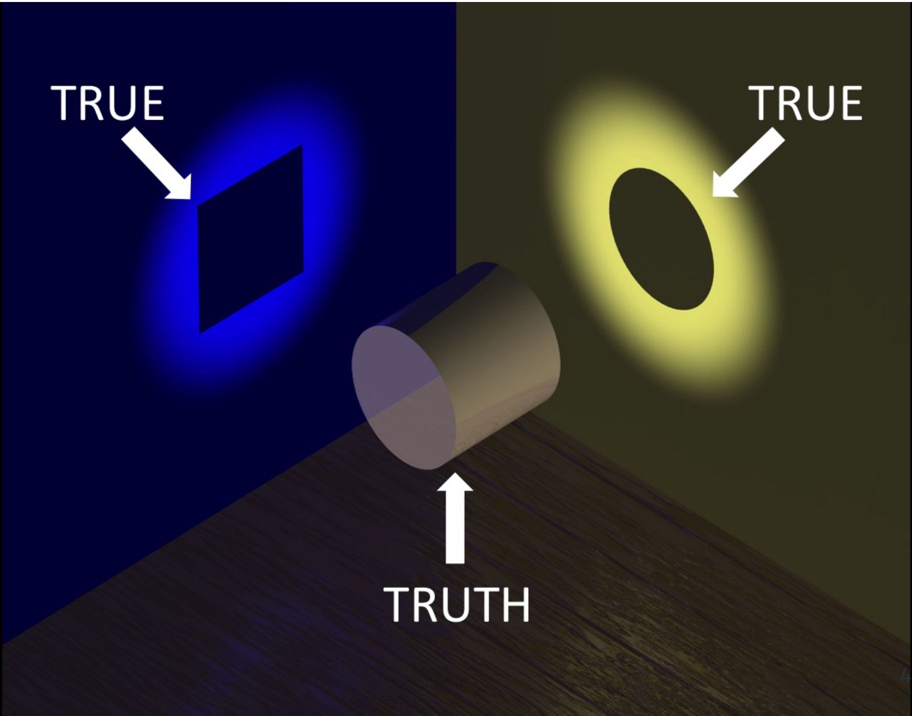
- Pose questions
- SC is a very active research field with many unsolved difficult problems to work on (for some questions there is no answer yet)

Practicalities

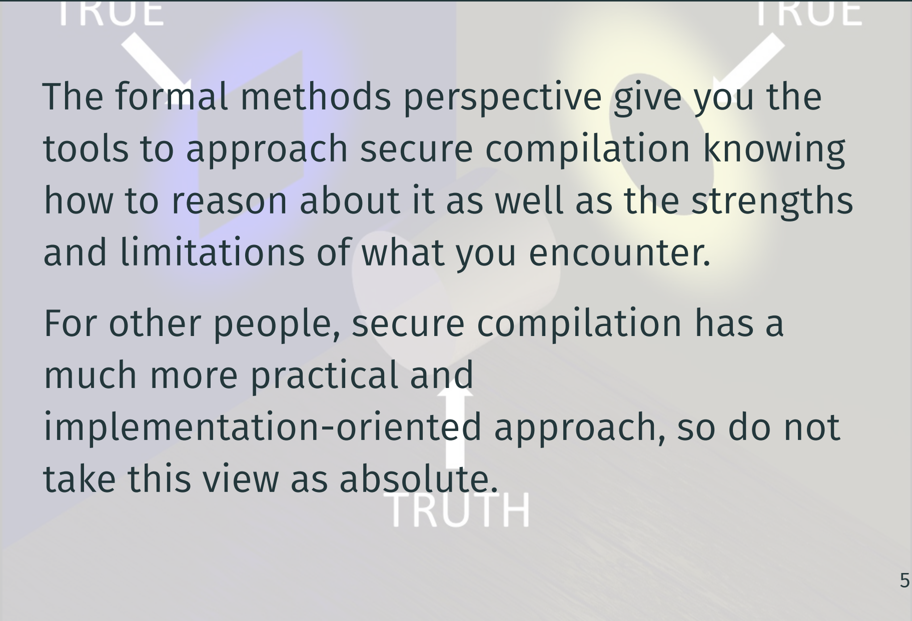
- Pose questions
- SC is a very active research field with many unsolved difficult problems to work on
  (for some questions there is no answer yet)
- Course flavour: formal methods.

Practicalities

- Pose questions
- SC is a very active research field with many unsolved difficult problems to work on (for some questions there is no answer yet)
- Course flavour: formal methods.
- You are encouraged to discuss how to bridge the gap between formality and practicality


TRUE
TRUE
TRUTH

A Note on Flavour



The formal methods perspective give you the tools to approach secure compilation knowing how to reason about it as well as the strengths and limitations of what you encounter.

For other people, secure compilation has a much more practical and implementation-oriented approach, so do not take this view as absolute.

Couse Outline

Topic	Reason / Also used for
Program equivalence	Reasoning about programs (e.g., their security)
Full Abstraction [1]	Criterion for SC / Comparing expressiveness of systems
Program behaviour (traces)	Reasoning about program security: (hyper)properties
Robust Compilation [2]	Several Criteria for SC that preserve (hyper)properties
Trace-preserving compilation [3]	Criterion for SC / Translating the meaning of safety
Complete Backtranslation example	Understand how to formalise languages / Detailing FAC and RC proofs
PMA results [4,5]	Understanding SC work

1. Formal Approaches to Secure Compilation. Patrignani *et al.* ACM CSUR '19 → [1], [58], [67], [97]
2. Exploring Robust Property Preservation for Secure Compilation. Abate *et al.* Arxiv '18
3. Secure Compilation as Hyperproperty Preservation. Patrignani *et al.* CSF '17
4. Secure Compilation to Protected Module Architectures. Patrignani *et al.* ACM TOPLAS '15
5. On Modular and Fully Abstract Compilation. Patrignani *et al.* CSF '16

Papers List

Send preferences for each group within 2 weeks.

1. Fully-Abstract Compilation by Approximate Back-Translation. Devriese *et al.* POPL '16
2. Fully Abstract Compilation via Universal Embedding. New *et al.* ICFP '16
3. Fully Abstract Compilation to JavaScript. Fournet *et al.* POPL '13
4. Noninterference for free. Bowman *et al.* ICFP '15

A. When Good Components Go Bad: Formally Secure Compilation Despite Dynamic Compromise. Abate *et al.* CCS '18

B. The Correctness-Security Gap in Compiler Optimization. D'Silva *et al.* LangSec '15
+ Parametricity versus the Universal Type. Devriese *et al.* POPL '18

C. FunTAL: Reasonably Mixing a Functional Language with Assembly. Patterson *et al.* PLDI '17

D. Jasmin: High- assurance and high-speed cryptography. J. B. Almeida *et al.* CCS. 2017.

E. Secure compilation of side-channel countermeasures: the case of cryptographic "constant-time". G. Barthe *et al.* CSF. 2018.