What is Secure Compilation?

summer semester 18-19, block

Marco Patrignani[1,2]

Stanford University
CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Practicalities

- Monday, Tuesday, Wednesday, Friday, Monday, Tuesday

Practicalities

- Monday, Tuesday, Wednesday, Friday, Monday, Tuesday
- 4h30, 2 breaks remind me

Practicalities

- Monday, Tuesday, Wednesday, Friday, Monday, Tuesday
- 4h30, 2 breaks remind me
- Type of course: lectures + presentations

Practicalities

- Monday, Tuesday, Wednesday, Friday, Monday, Tuesday
- 4h30, 2 breaks remind me
- Type of course: lectures + presentations
- Course goal:
    - understand background and motivation behind SC
    - learn reasoning techniques for SC
    - know the most recent developments in SC

Practicalities

- Monday, Tuesday, Wednesday, Friday, Monday, Tuesday
- 4h30, 2 breaks remind me
- Type of course: lectures + presentations
- Course goal:
  - understand background and motivation behind SC
  - learn reasoning techniques for SC
  - know the most recent developments in SC
- Evaluation: presentations, reports.

Practicalities

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

Practicalities

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

Practicalities

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

Practicalities

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

Practicalities

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

A Note on Flavour

Formal methods give you the tools to reason about things and to reason about the motivation why things are done in a certain way.

Couse Outline

- Develop a super toy formal compiler

Couse Outline

- Develop a super toy formal compiler
- Prove it is correct, understand why it is not secure

Couse Outline

- Develop a super toy formal compiler
- Prove it is correct, understand why it is not secure
- Prove that it is Fully Abstract via Backtranslations

Couse Outline

- Develop a super toy formal compiler
- Prove it is correct, understand why it is not secure
- Prove that it is Fully Abstract via Backtranslations
- Understand why Full Abstraction yields security

Couse Outline

- Develop a super toy formal compiler
- Prove it is correct, understand why it is not secure
- Prove that it is Fully Abstract via Backtranslations
- Understand why Full Abstraction yields security
- Prove that it is Robustly Safe

Couse Outline

- Develop a super toy formal compiler
- Prove it is correct, understand why it is not secure
- Prove that it is Fully Abstract via Backtranslations
- Understand why Full Abstraction yields security
- Prove that it is Robustly Safe
- Understand why Robust Compilation yields security

Problems

- Programming abstractions are not preserved by compilers (linkers etc) (security is an abstraction)

Problems

- Programming abstractions are not preserved by compilers (linkers etc) (security is an abstraction)
- what does preserving abstractions mean?

Problems

- Programming abstractions are not preserved by compilers (linkers etc) (security is an abstraction)
- what does preserving abstractions mean?
- what tools are there to preserve abstractions?

Solutions

- Study what preserving abstractions means via secure compilation criteria

Solutions

- Study what preserving abstractions means via secure compilation criteria
- Devise efficient enforcement mechanisms to attain security

Solutions

- Study what preserving abstractions means via secure compilation criteria
- Devise efficient enforcement mechanisms to attain security
- Prove compilers can use these mechanisms for security

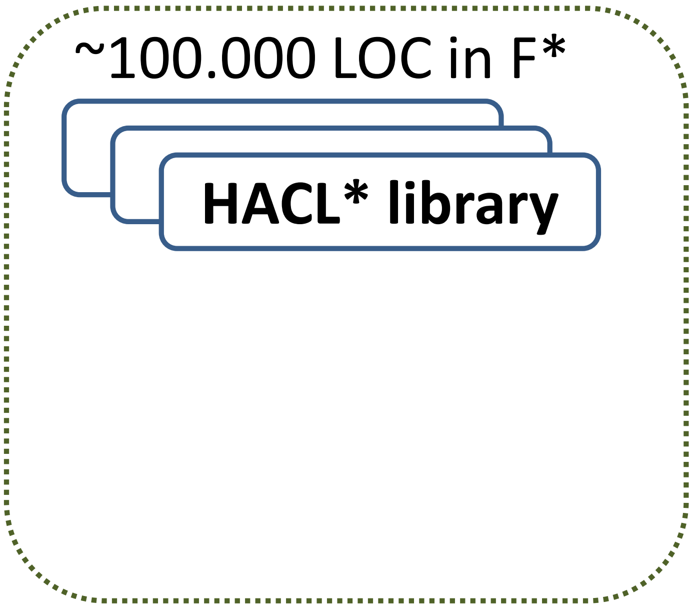
Recommended Reading

- http://drops.dagstuhl.de/opus/volltexte/2018/9
- https://blog.sigplan.org/2019/07/01/secure-compilation/

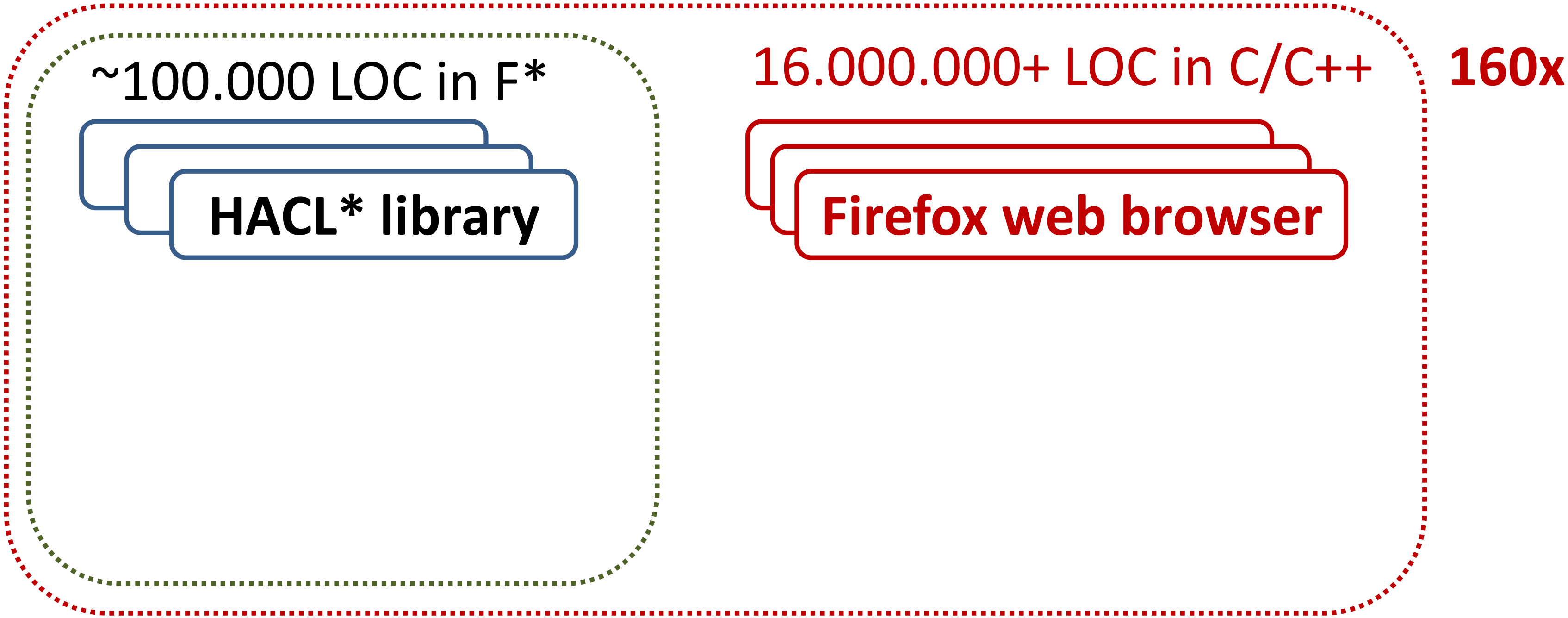
A First Example

(borrowed from Catalin Hritcu)

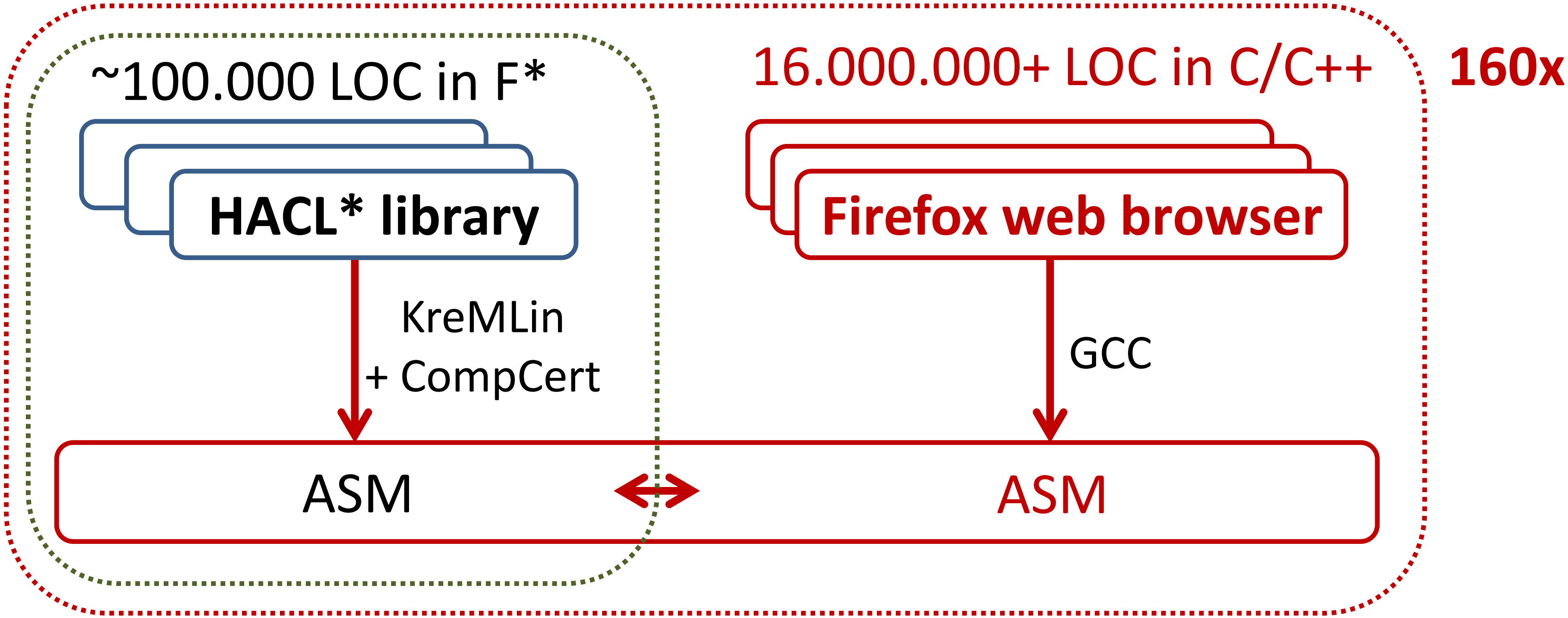
HACL* verified cryptographic library



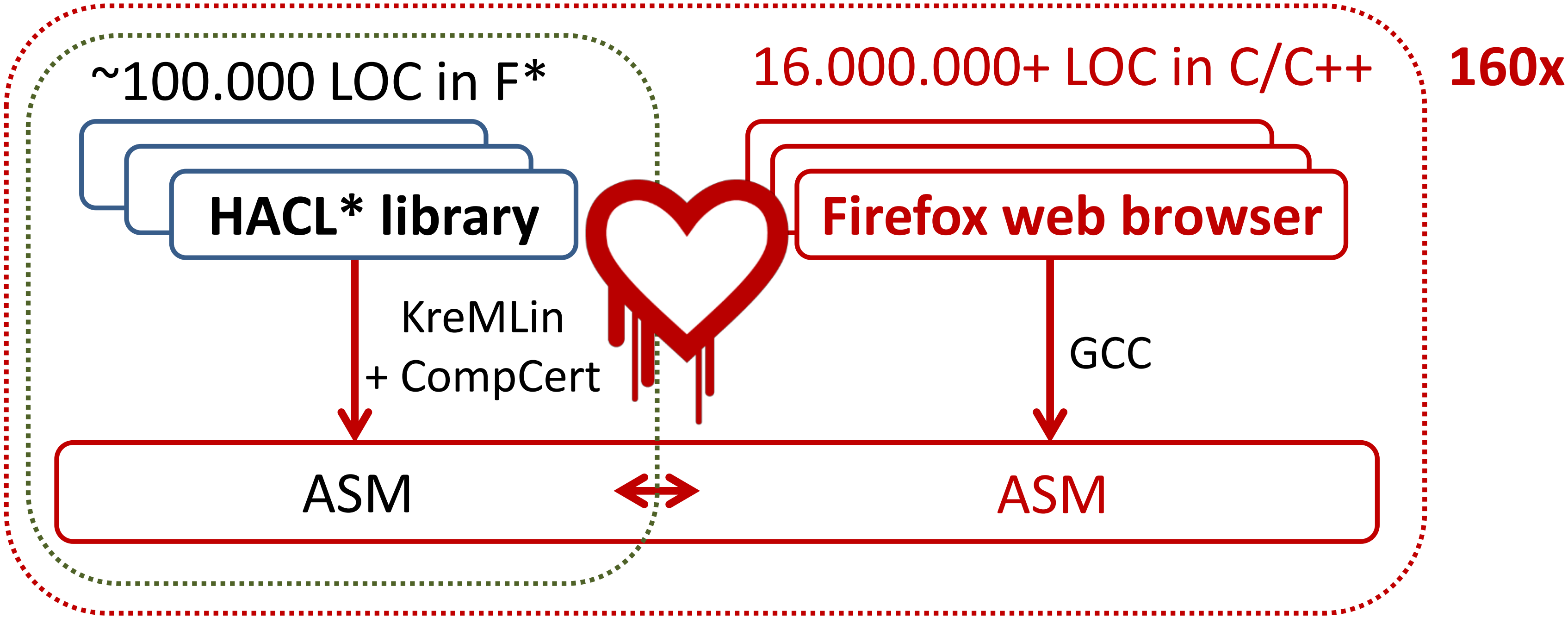
HACL* verified cryptographic library, in practice



HACL* verified cryptographic library, in practice

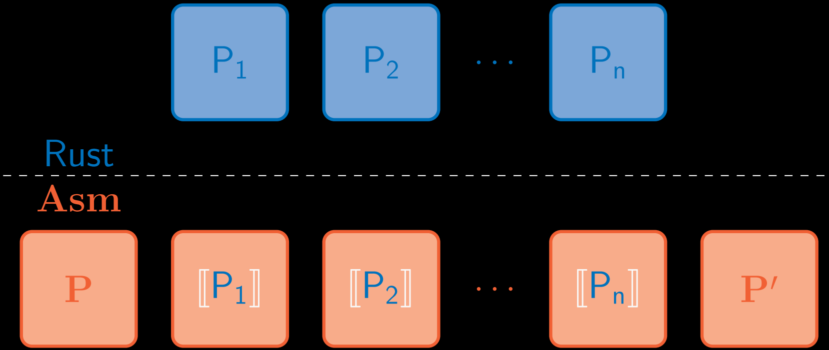


HACL* verified cryptographic library, in practice

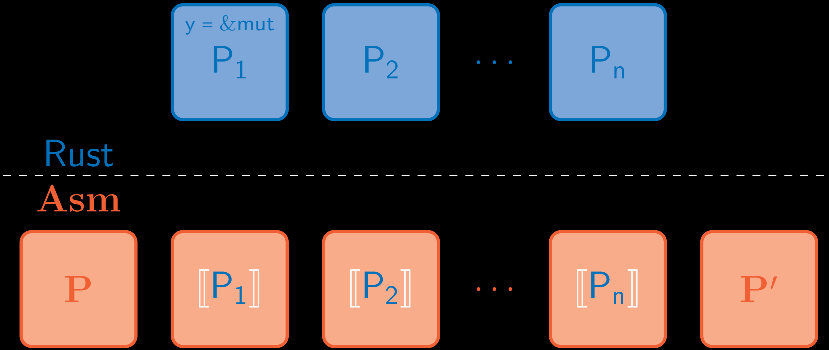


Insecure interoperability: linked code can read and write data and code, jump to arbitrary instructions, smash the stack, ...

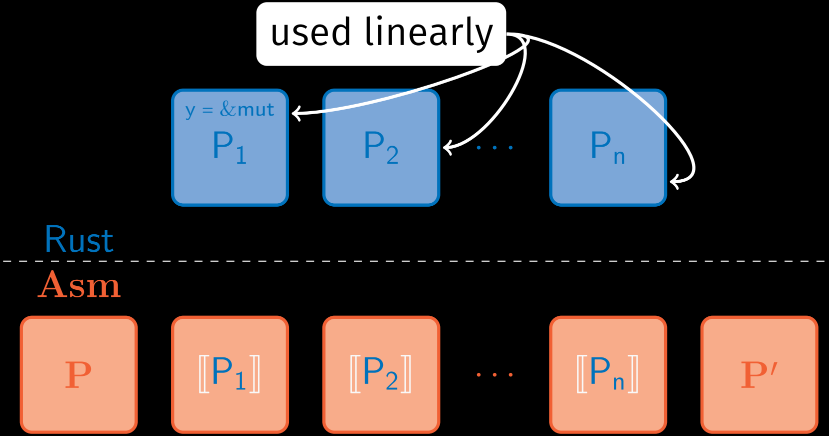
A Second Example



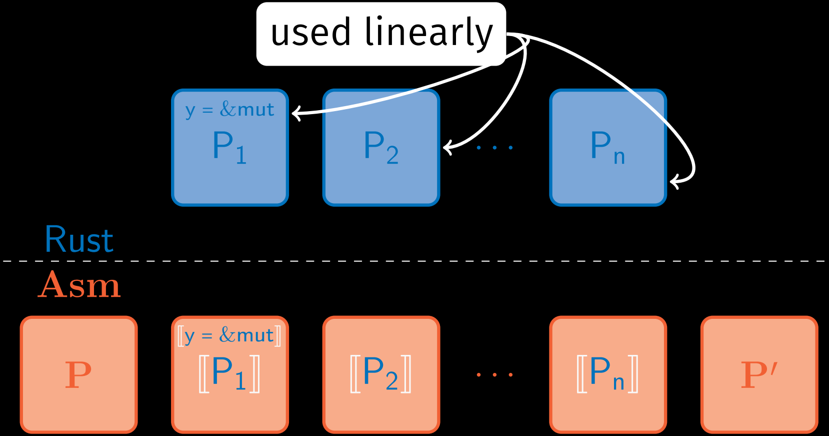
A Second Example



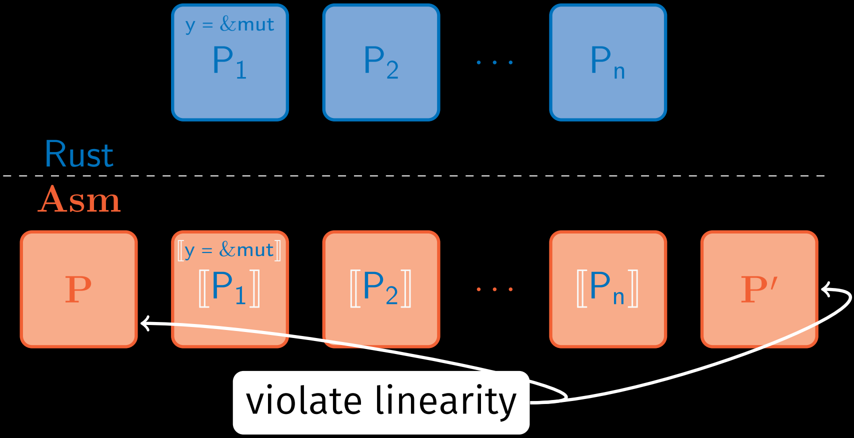
A Second Example



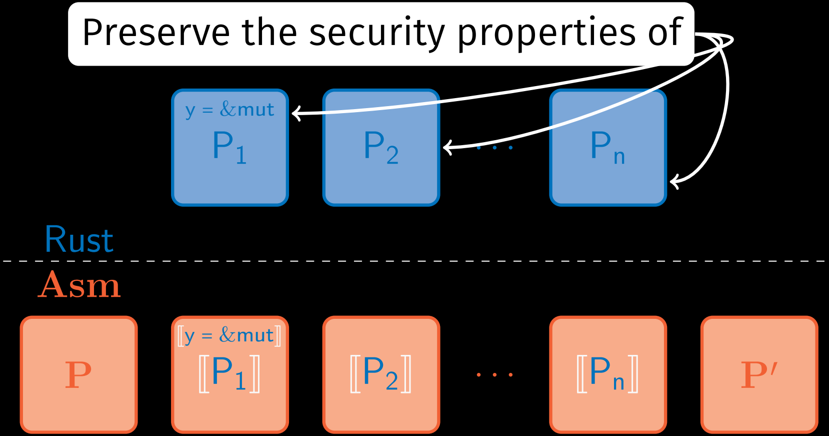
A Second Example



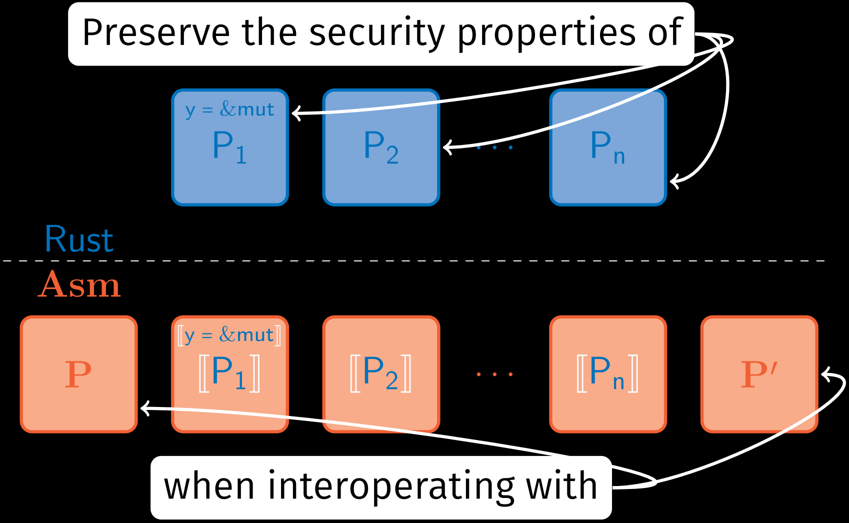
A Second Example



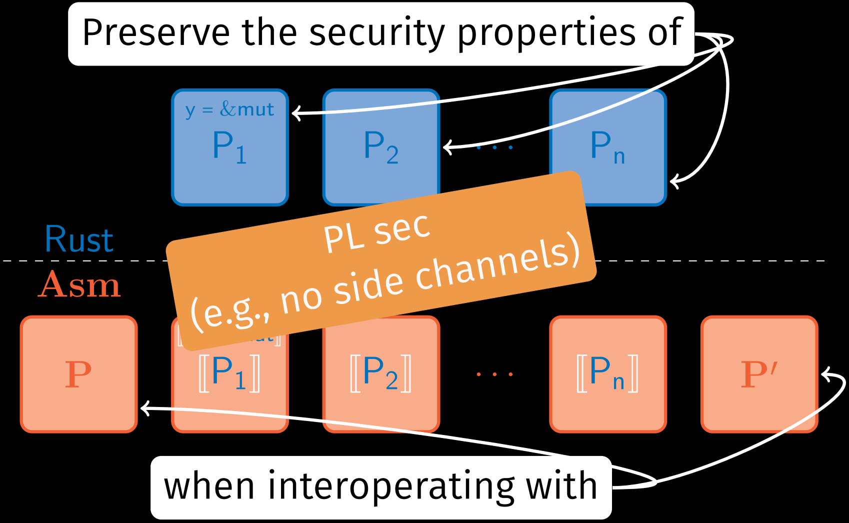
A Second Example



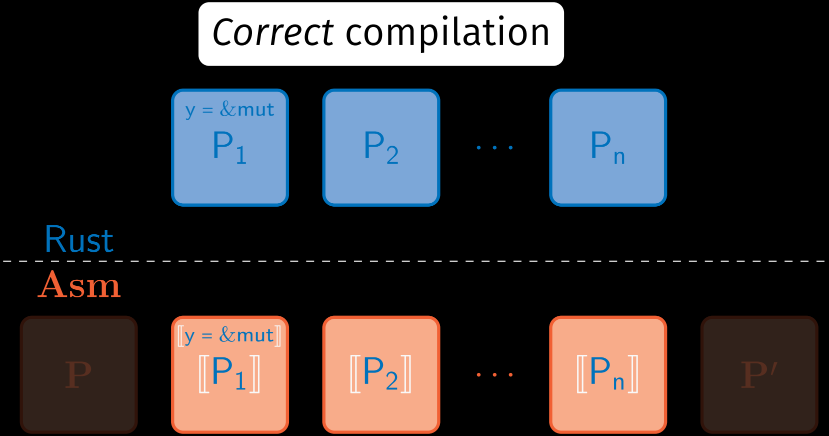
A Second Example



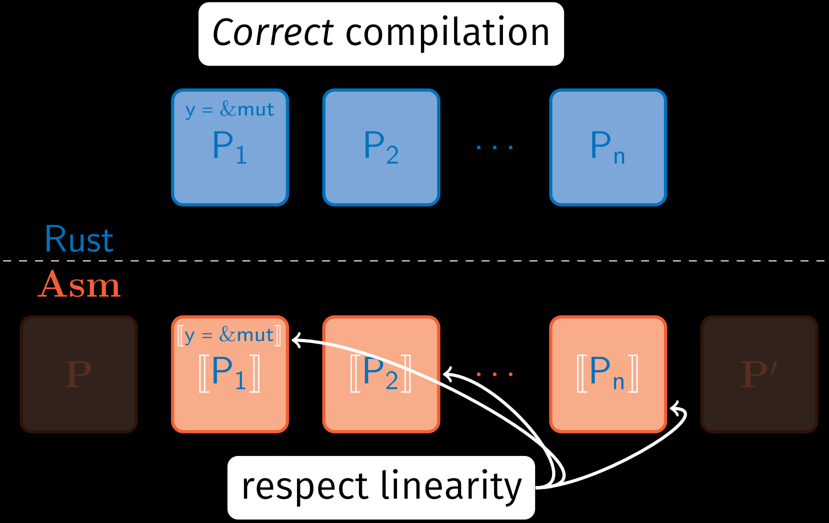
A Second Example



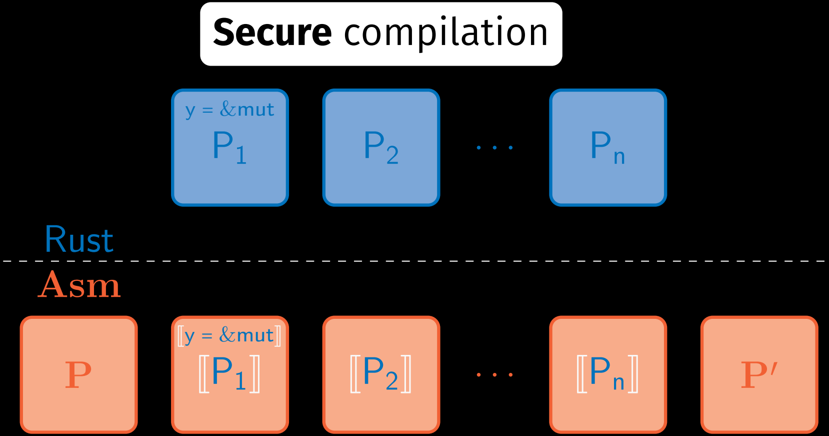
A Second Example



A Second Example



A Second Example



A Second Example

Enable source-level security reasoning