Formal methods and security

What to expect when proving?

Security motivates verification.

How do proofs influence security?

Overlap w/ prior paper

- Empirical study.
- Amazon.

Why proofs for security?

Negative goal: no way to compromise.

Corner cases matter.

Proofs: consider all cases.

Machine-checked.

Code irrelevant; consider spec.

Why not proofs for security?

- Lots of effort → alternatives?
  - Fuzzing
  - Audits
  - Refactor / Isolation
- Idiosynchratic code.
  - Less clean.
  - Error-prone in ways that bypass the spec.
- Threat model.
  - ↳ Pedantic + incorrect threat model.
- Wrong assumptions.
  - Imprecise CPU model.
  - Missing timing. ←
  - Missing non-det.
  - Missing features.
  - Physical attacks → Rowhammer.

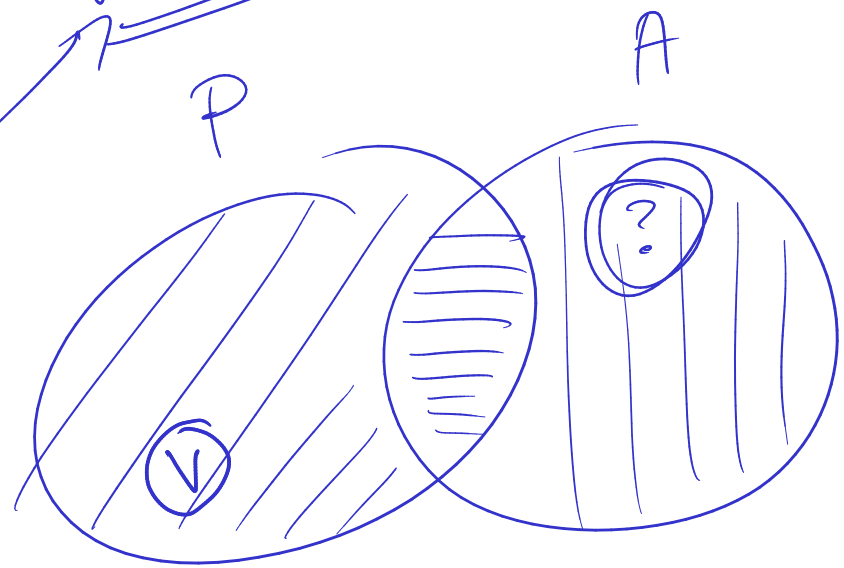
  } Underlying assumptions.
- Attackers bypass known defenses.
  - Phishing.
  - Side channels ←
  - Debug / mgmt interfaces.
  - Physical attacks. ←

  } spec
- Hard to interpret theorem.
  - bcpierce: 1 week to read the seL4 thm.

Code changes

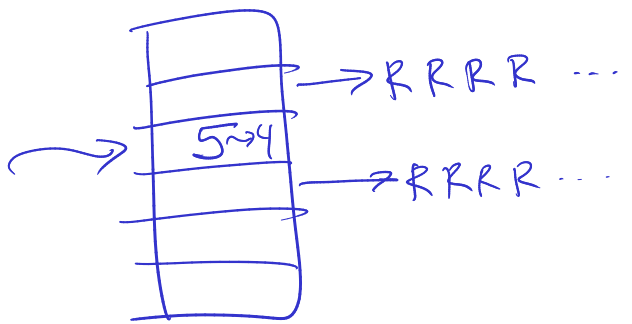
To make the sys. secure? \ To verify?	No	Yes. (P)
No	mundane.	overhead of proofs (broke security? V).
Yes (A)	misguided theorem?	great!

unreadable code?



Breakout Q: experience w/ verif (obs?)

Bowhammer



Qualified guarantees

Forces precise threat model.
Conditional correctness / security.

→ Hard to extract assumptions.
→ Hard to understand implications.

ARM ISA.
(Over-simplified: UNSPEC.

Facebook Infer.
IronFleet
Everest

Structured exploration.

Forces dev. to think precisely.

Butler's steps:

1. State –
2. Spec –
3. Abstraction –
4. Proof. ←

Diminishing returns.

Amazon.
NetCore.
I4.

Side effects

Significant effort?
Awkward code?
Awkward shim layer?

Over-reliance on proven defenses

Java VM / Javascript engine
↳ Isolation : mem. safety.

Measures: ASLR, canaries, overflow checks, ..

"Defense in depth":
Mitigation of wrong threat model.