Correctness of verified sys?

Verif. => no bugs?
Empirical study: dist. sys.
IronFleet, Verdi, Chapar
Bugs in shims, specs, tools
Violate guarantees, crash

How to find bugs?
Study code, design, ...
Fuzzing shim
Cross-checking
Serious effort!

Verified sys.
User/developer

Tools
Verifier core
Build system

App
Exec code
shim
OS
HW
spec
Verified code
axiom
Iron Fleet
---
RSM Counter

IronFleet code

Dafny

Boogie

Z3

Shim X
Spec ✓ Tooling ✓

Tooling bugs (I2, I3, I4)

Errors ignored.

→ NuBuild parses output, check for errors
→ Only checks first msg.

Dafny: ignores Z3 signals.

How to avoid?

Positive checks instead of errors
Test tooling for proof failures.
Make proof checks more integral.
Iron Fleet spec ambiguity

Challenge: replica fails to reply? retransmit \rightarrow duplicate exec?

Goal: exactly-once execution \rightarrow dedup

Iron Fleet: code for filtering dupes \rightarrow spec did not promise dedup.

Other spec bugs

NetCore: no output packets.

\rightarrow FSCQ: spec vs error codes vs SibylFS.

How to avoid spec bugs?

Spec \rightarrow prove an app using spec

\rightarrow prove properties

\rightarrow test/cross-check SibylFS.

Non-determinism

Simple

Performance.
Verdi: RSM
Coq $\rightarrow$ Ocaml
+ shims
TCP, FS.

V3: large writes not atomic

V4: incorrect snapshot write order

V5: failed to read snapshot

V8: out of stack space
find 6T Index.
Coq $\rightarrow$ Ocaml
Recursive, inefficient

V2: incorrect marshaling
client

V2: recv() complete msgs.

send (8KB)

GET k5 hello
world
Chapar
KV store
Causal consistent
UDP-based
Shim bugs!

C3: assumed no dups.
C2: assumed no drops.
C3: odd Ocaml net library.

Avoid shim bugs?
Expand verify boundary.
Narrow wait/stop API -> simpler spec.
Spec includes errors, resource use...
Shim: clean API
   well-designed libraries.
Test shim (Sibyl/FS)
Test corner cases: out of resources.
Lessons learned?

- Verified code → no bugs.
- Relies on assumptions
  - Spec → use it
  - Tools → test
  - Shims → fuzz