Co-evolving tracing and fault injection with Box of Pain

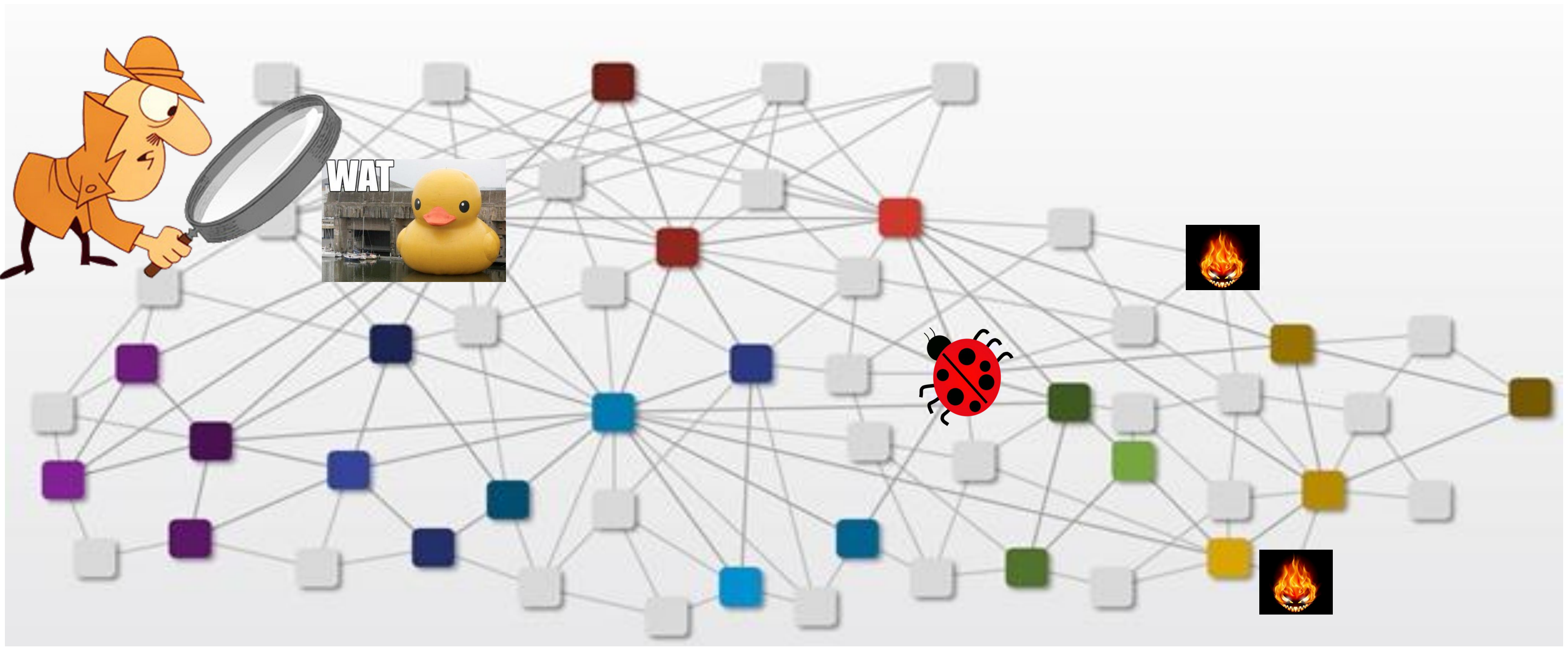
Daniel Bittman **Ethan Miller** **Peter Alvaro**

Center for Research in Storage Systems

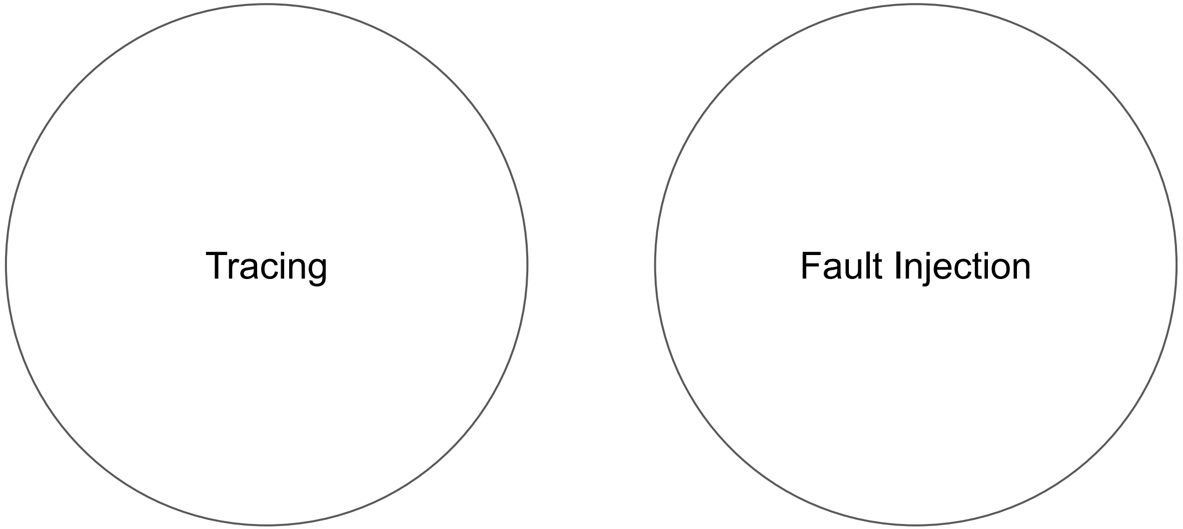
Disorderly Labs

UC Santa Cruz

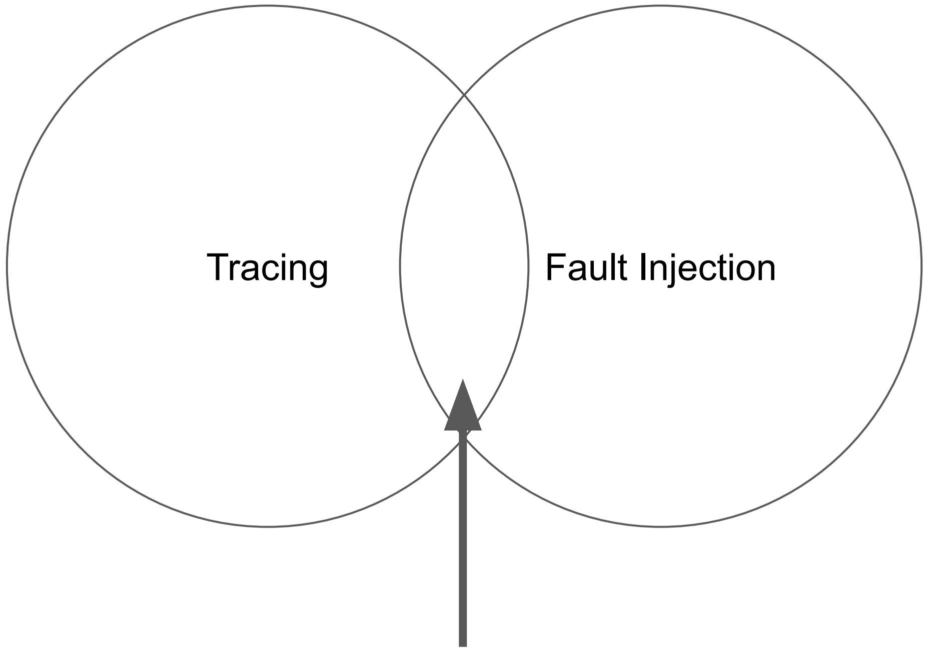
"Time of fault" bugs



Box of pain



Coevolution



Philosophy

Focus: rare but catastrophic "time of fault" bugs

Coarse tracing: communication across failure boundaries

Simulation: *Effects* of faults rather than their causes

Pragmatism: many runs are possible; few are likely

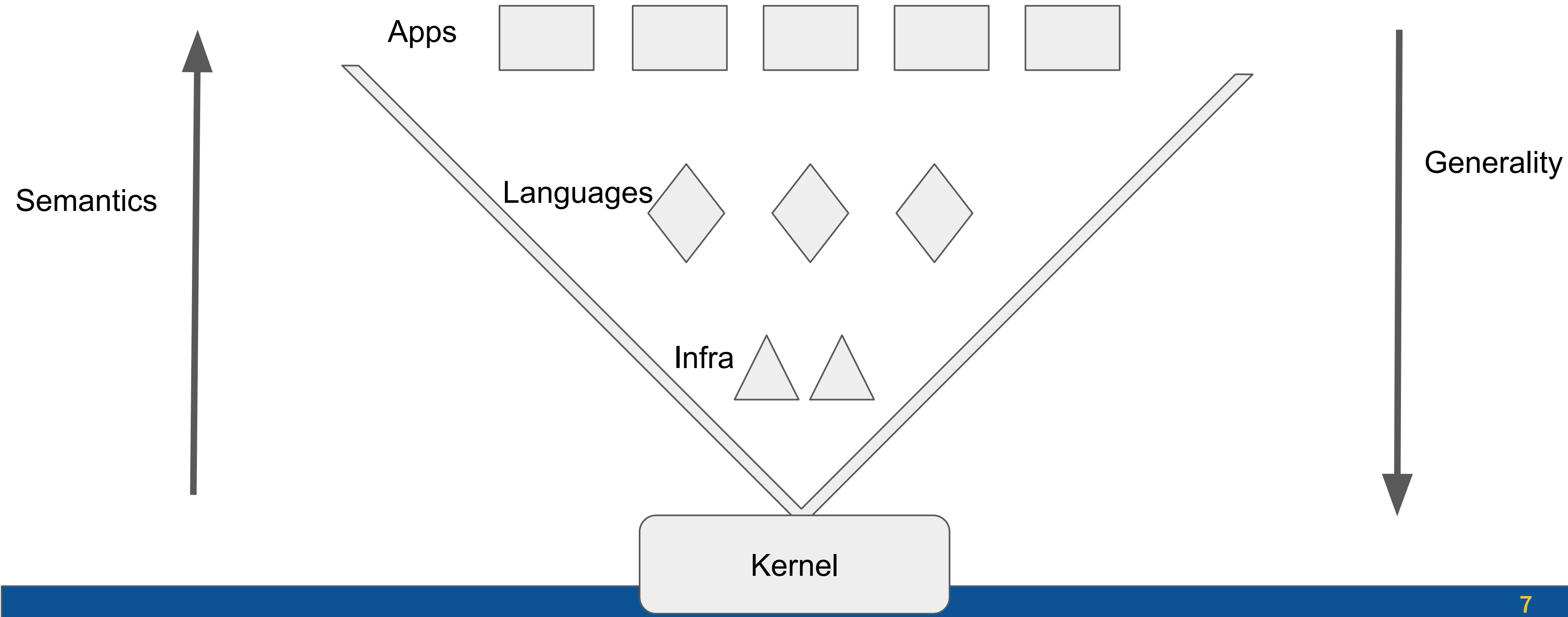
Box of Pain - architecture

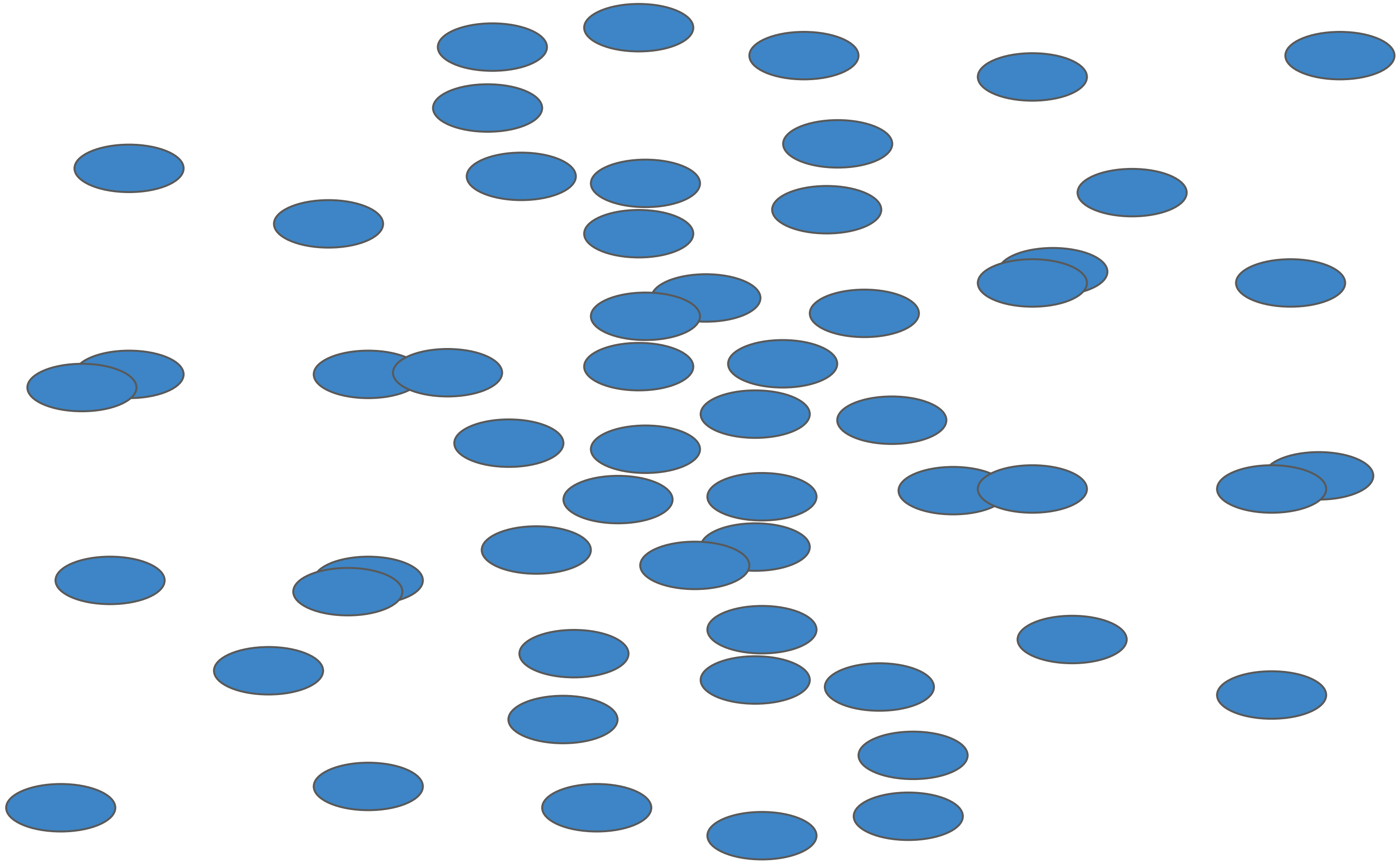
Tracer

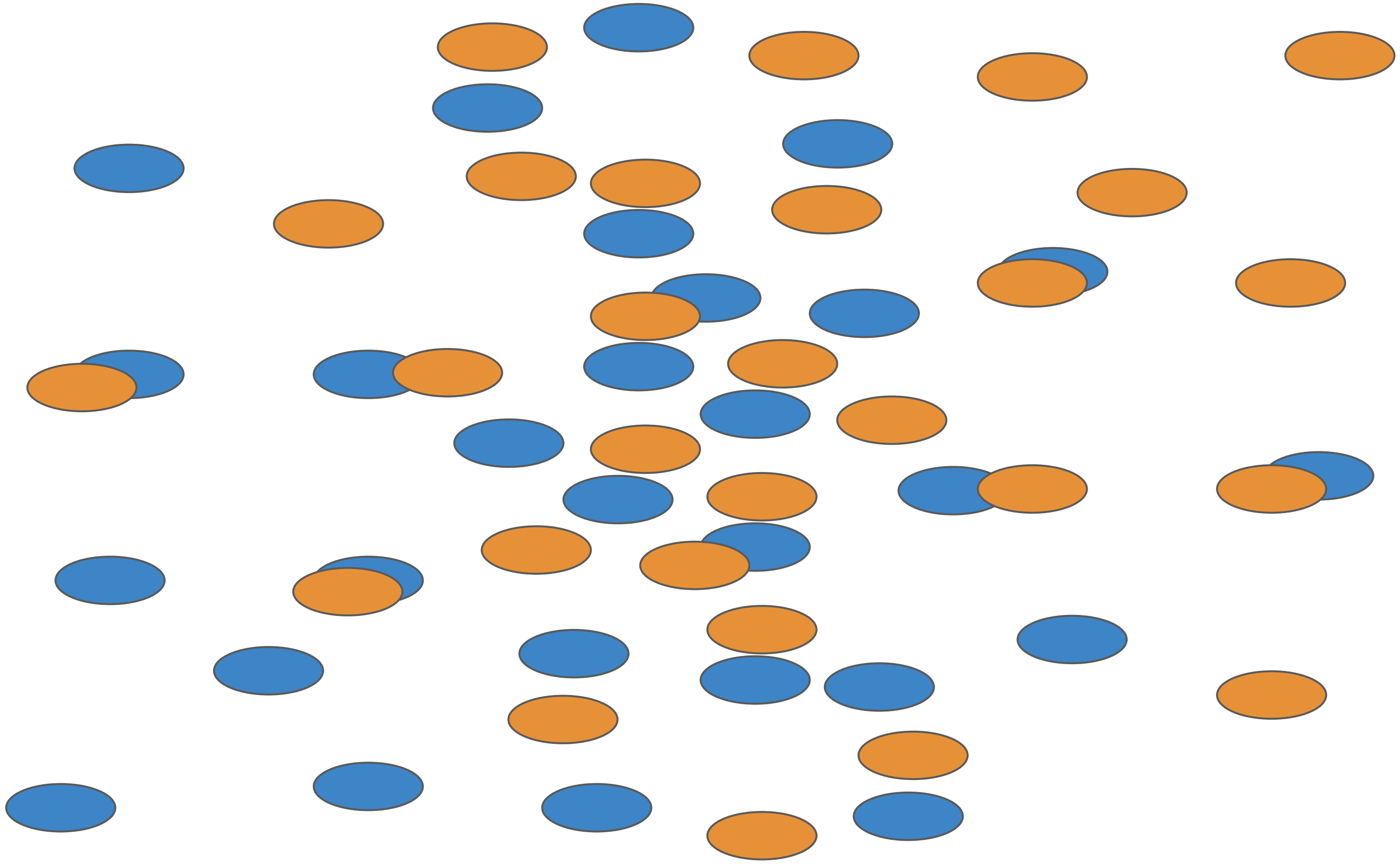
Tracker

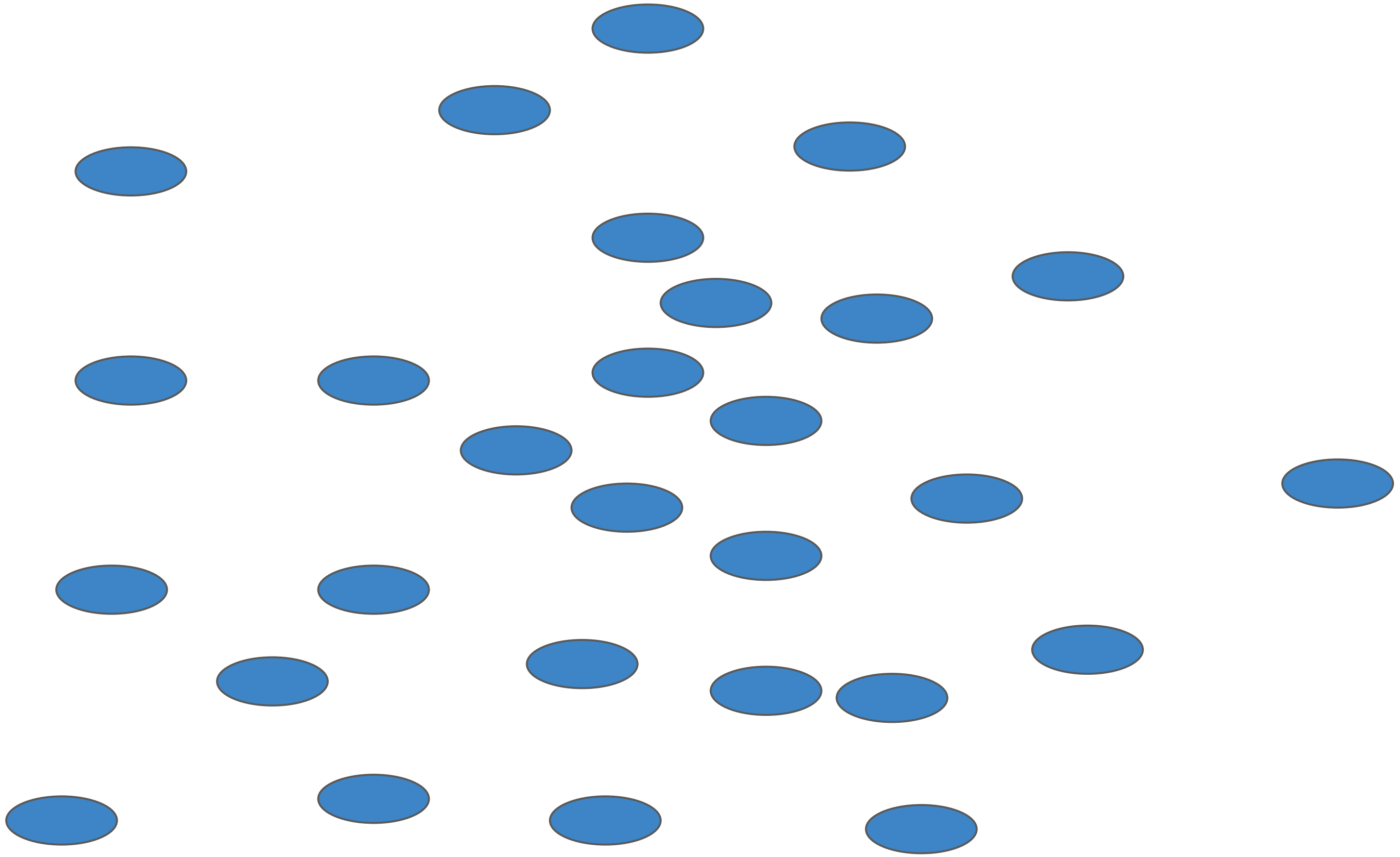
Fault Injector

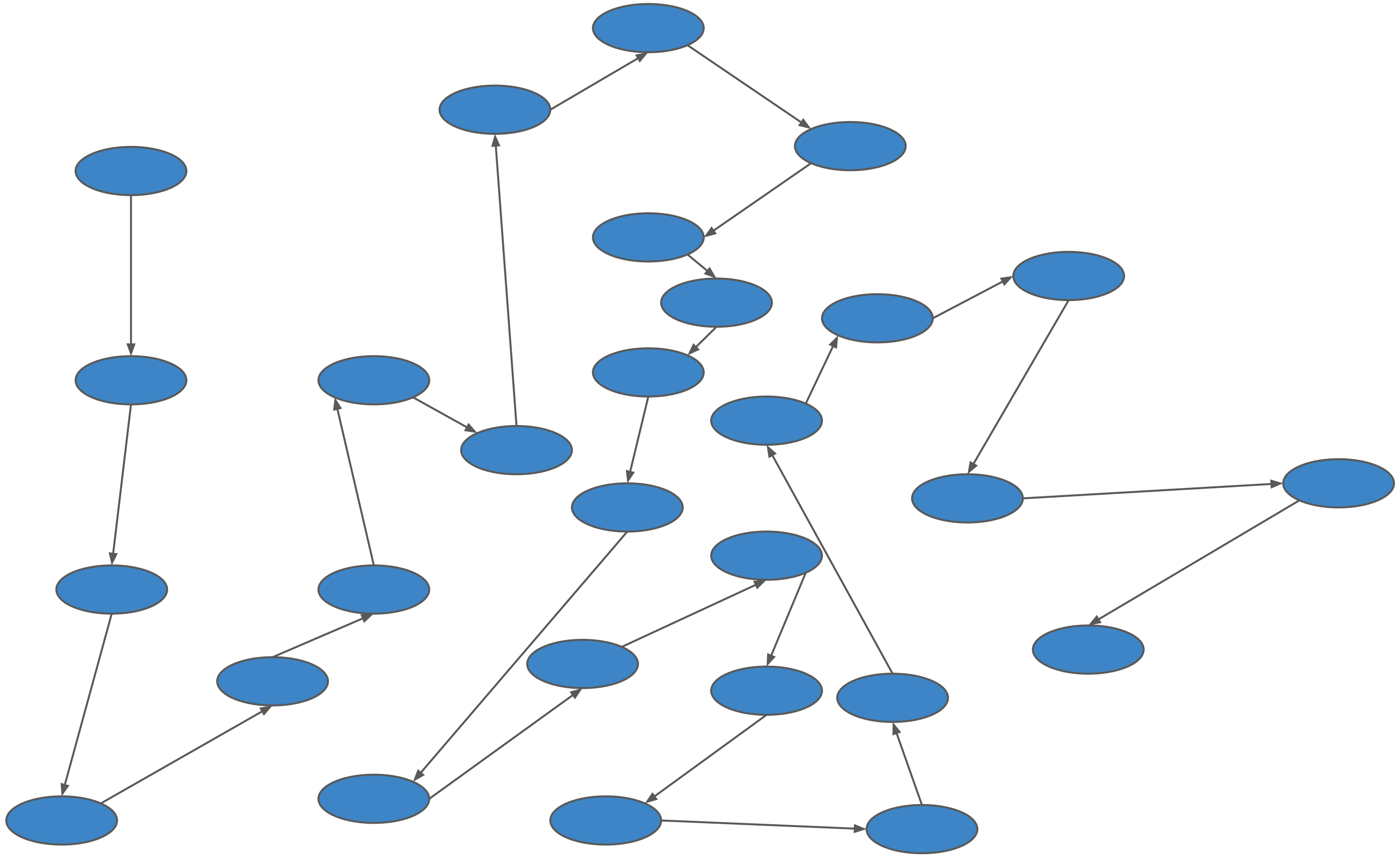
Interpose. But where?

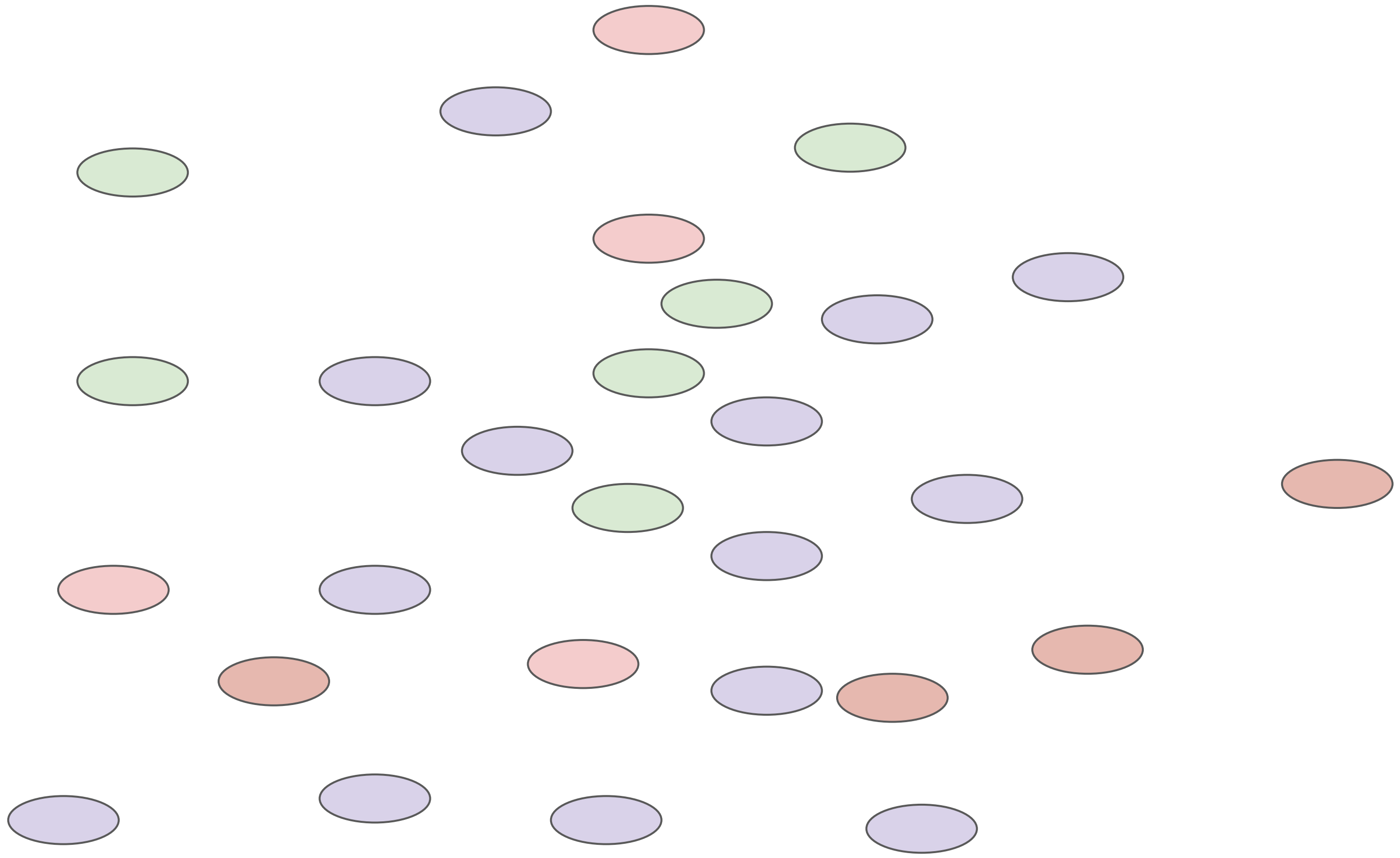


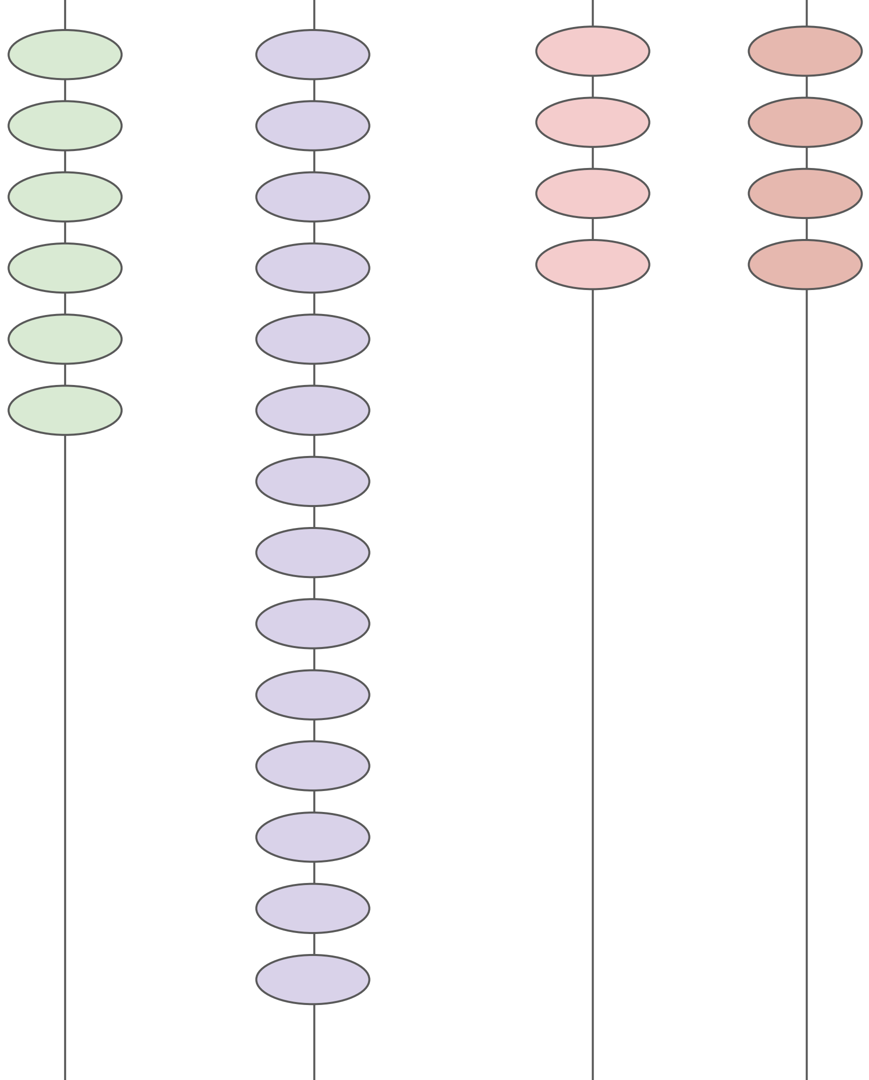


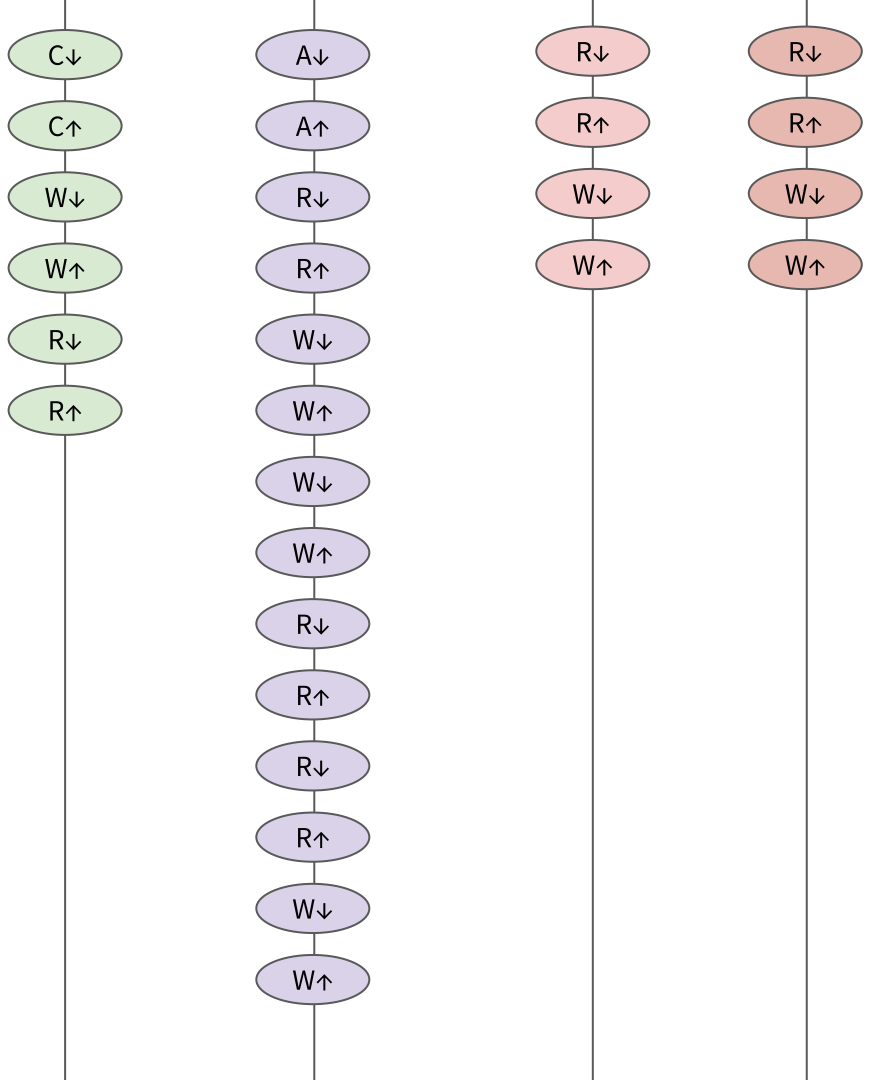








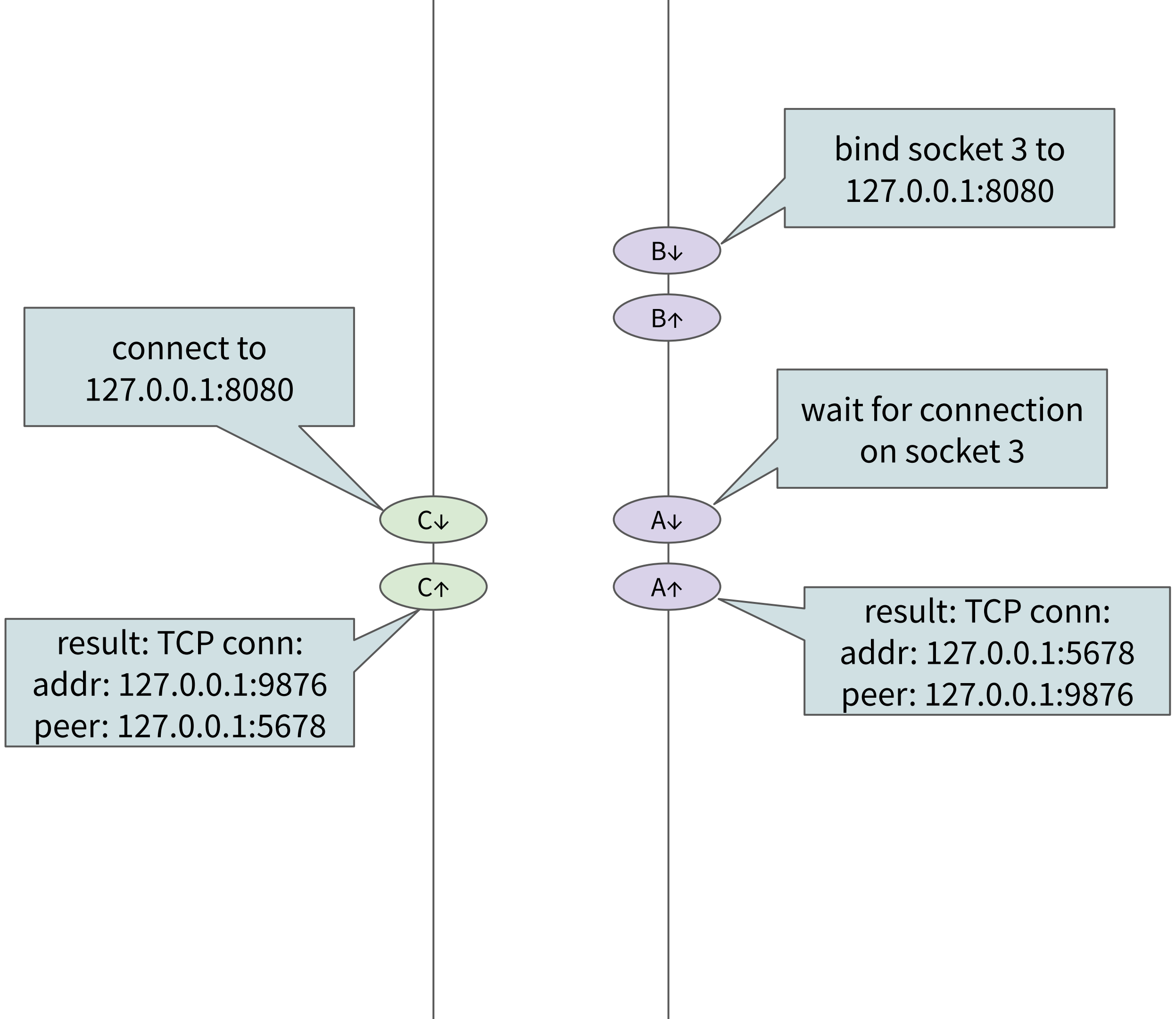


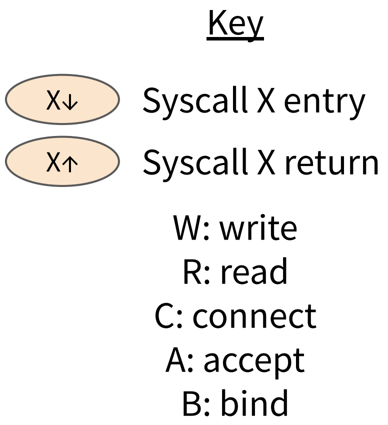


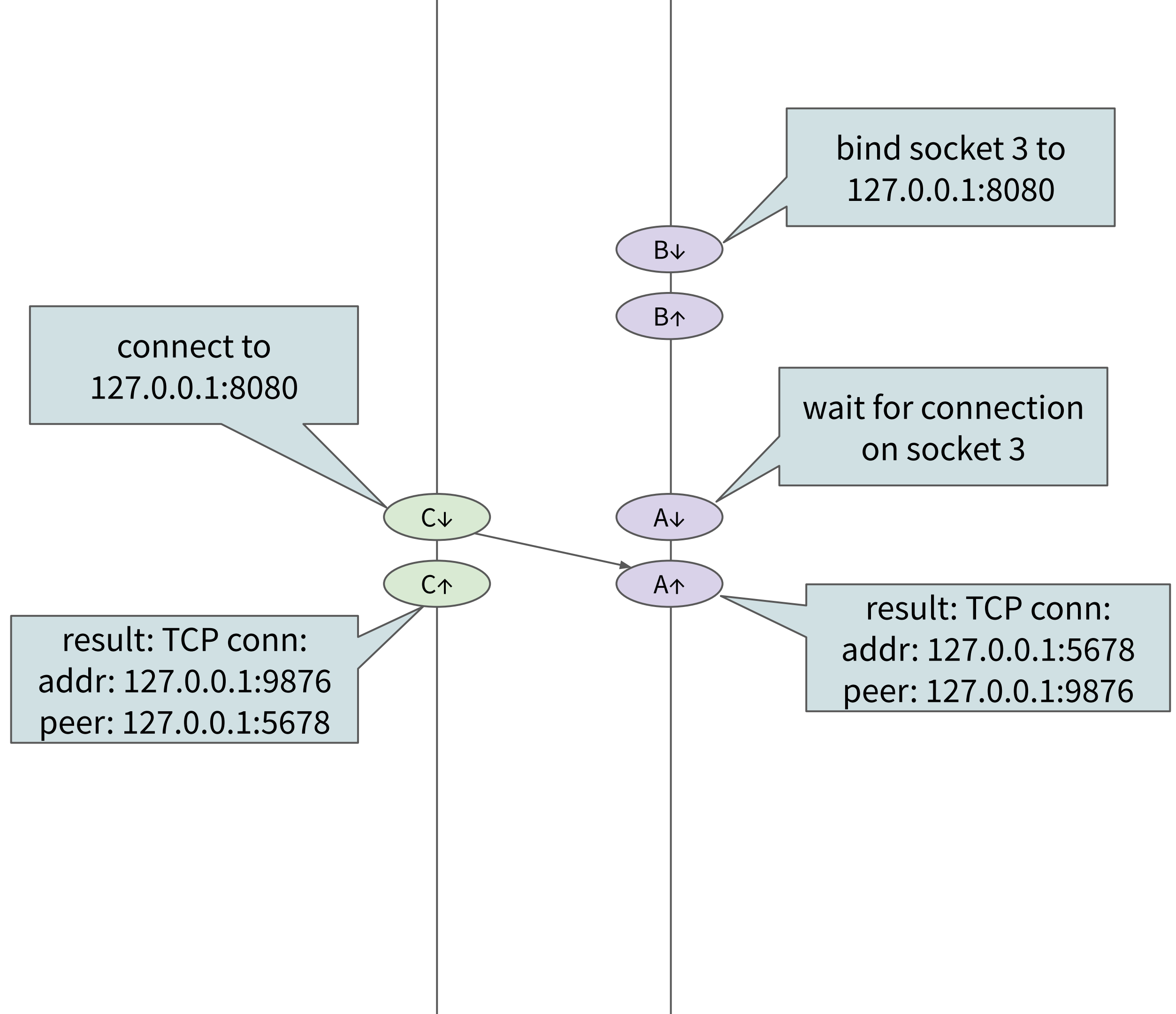
Key

X↓	Syscall X entry
X↑	Syscall X return

W: write
R: read
C: connect
A: accept


bind socket 3 to 127.0.0.1:8080
B↓
B↑
connect to 127.0.0.1:8080
wait for connection on socket 3
C↓
A↓
C↑
A↑
result: TCP conn: addr: 127.0.0.1:9876 peer: 127.0.0.1:5678
result: TCP conn: addr: 127.0.0.1:5678 peer: 127.0.0.1:9876


Key
X↓
Syscall X entry
X↑
Syscall X return
W: write
R: read
C: connect
A: accept
B: bind

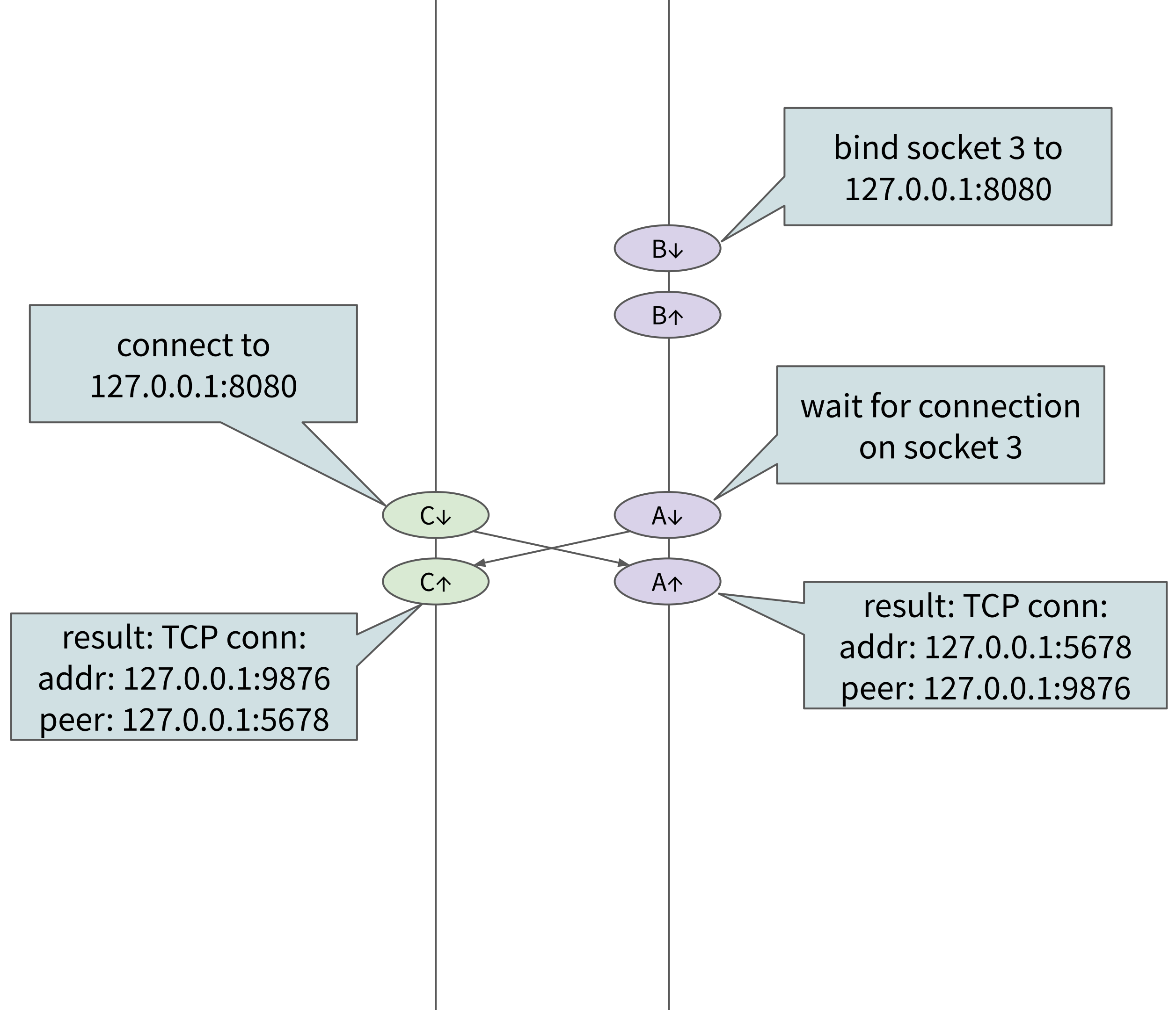

bind socket 3 to 127.0.0.1:8080
B↓
B↑
connect to 127.0.0.1:8080
wait for connection on socket 3
C↓
A↓
C↑
A↑
result: TCP conn: addr: 127.0.0.1:9876 peer: 127.0.0.1:5678
result: TCP conn: addr: 127.0.0.1:5678 peer: 127.0.0.1:9876

Key

X↓	Syscall X entry
X↑	Syscall X return

W: write
R: read
C: connect
A: accept
B: bind

A $\longrightarrow$ B: A *happens-before* B


bind socket 3 to 127.0.0.1:8080
B↓
B↑
connect to 127.0.0.1:8080
wait for connection on socket 3
C↓
A↓
C↑
A↑
result: TCP conn: addr: 127.0.0.1:9876 peer: 127.0.0.1:5678
result: TCP conn: addr: 127.0.0.1:5678 peer: 127.0.0.1:9876

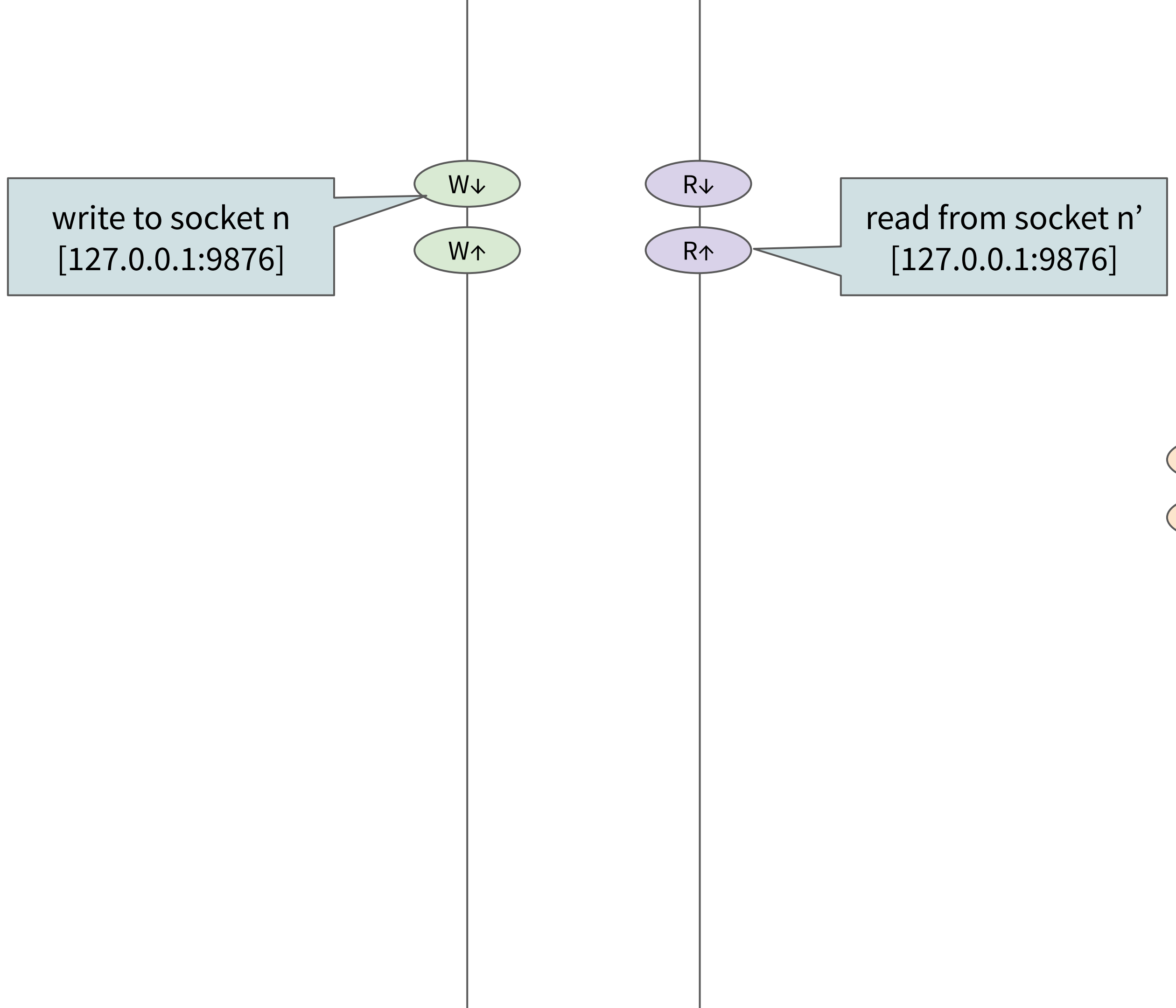
<u>Key</u>

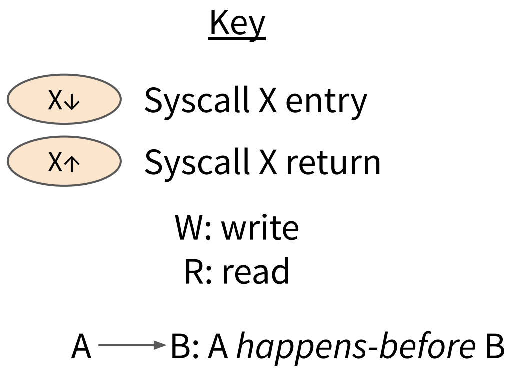
X↓ Syscall X entry

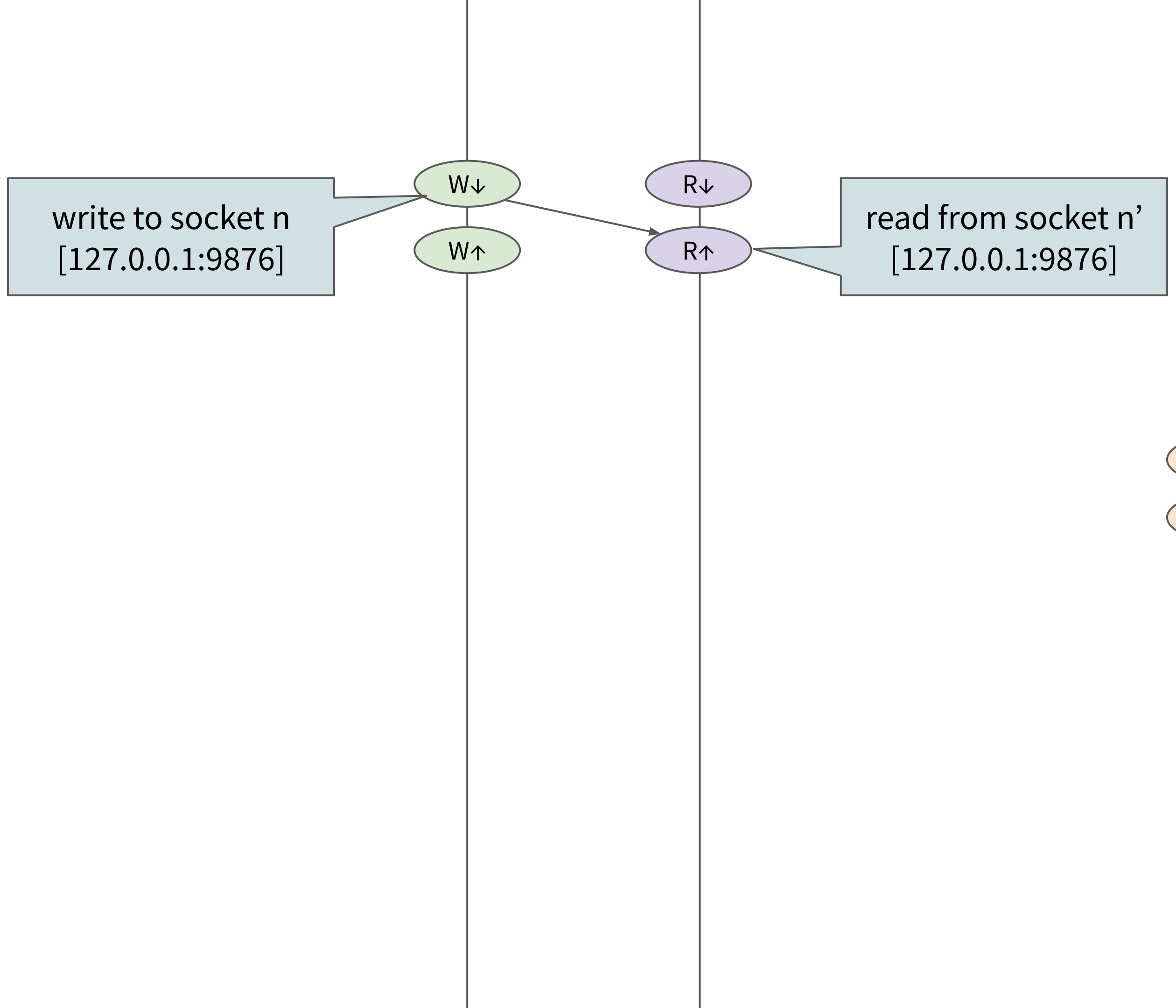
X↑ Syscall X return

W: write
R: read
C: connect
A: accept
B: bind

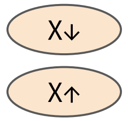
A ⟶ B: A *happens-before* B


write to socket n
[127.0.0.1:9876]
W↓
W↑
R↓
R↑
read from socket n’
[127.0.0.1:9876]


Key
X↓
Syscall X entry
X↑
Syscall X return
W: write
R: read
A
B: A *happens-before* B



Key

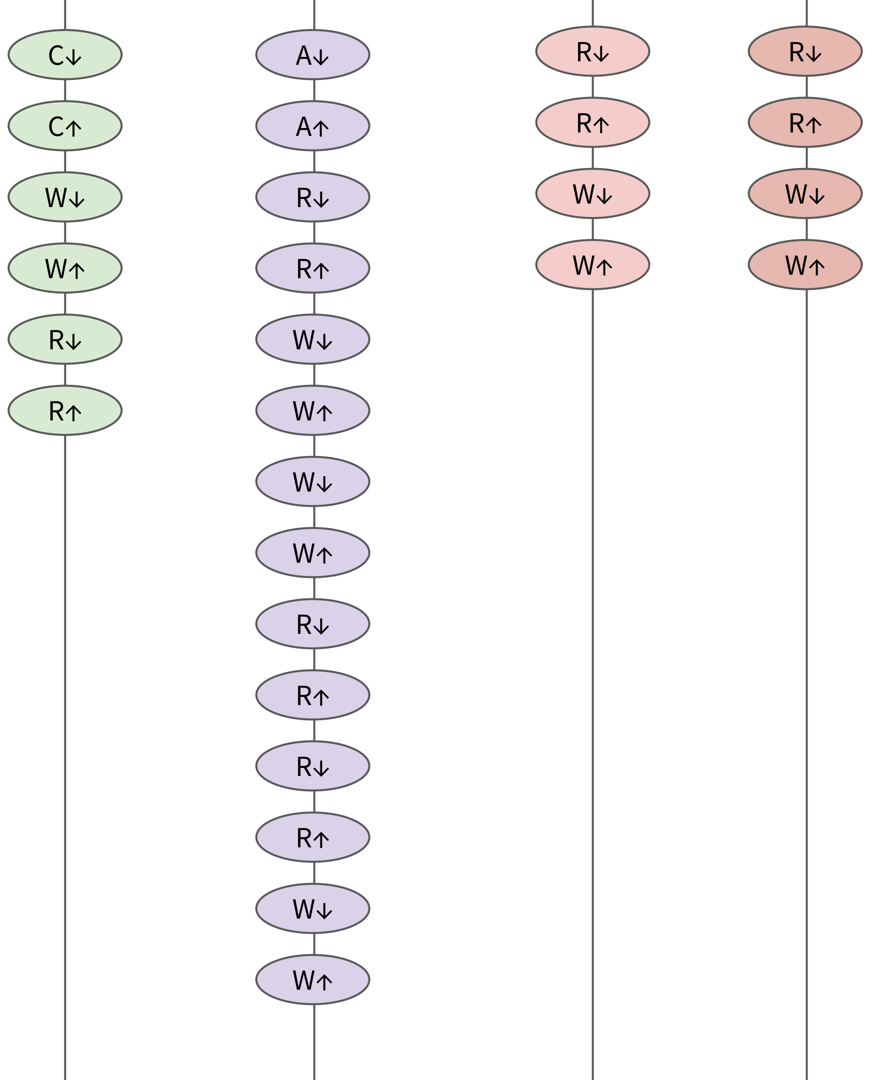


Syscall X entry

Syscall X return

W: write
R: read

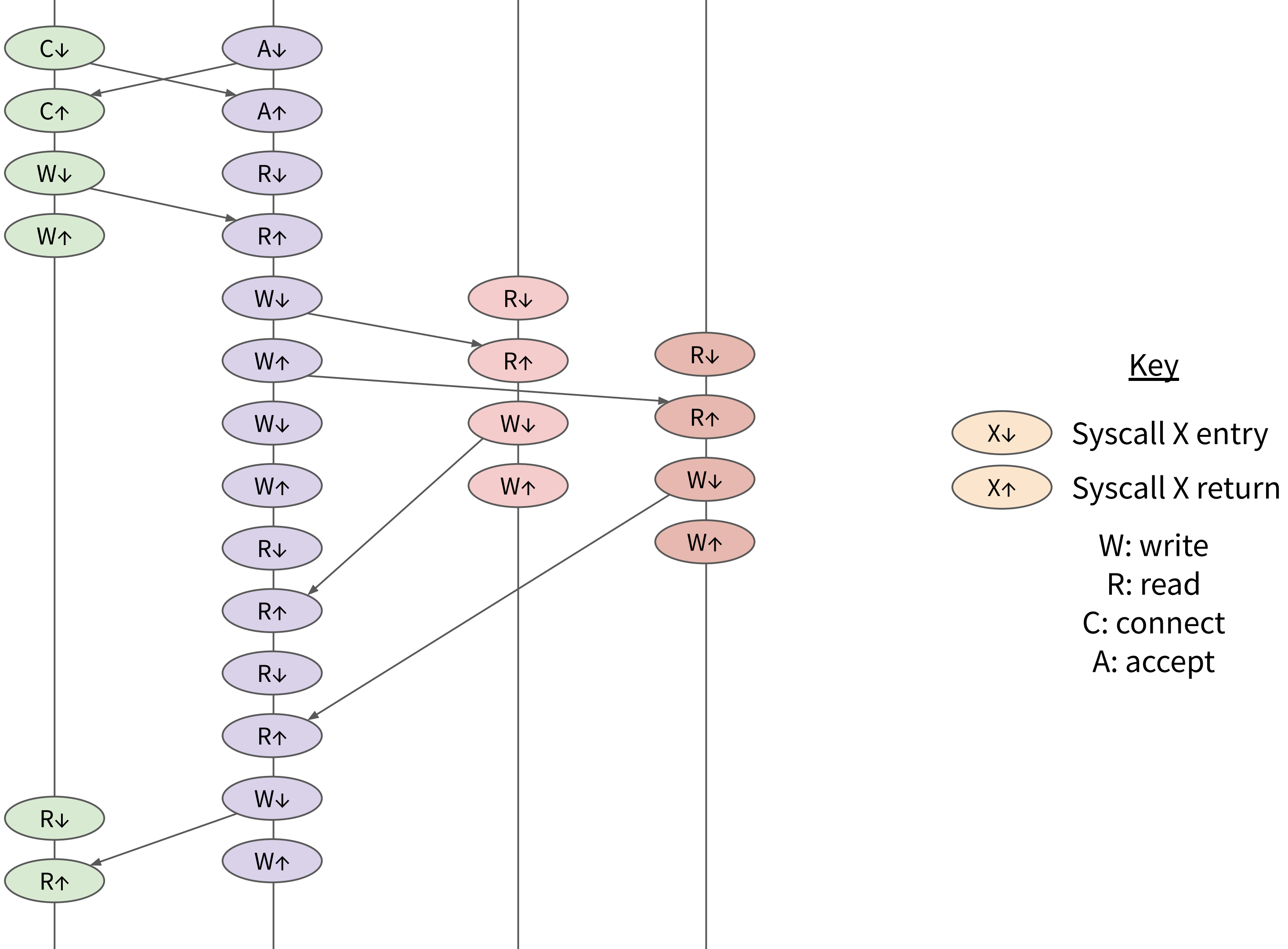
A ⟶ B: A *happens-before* B

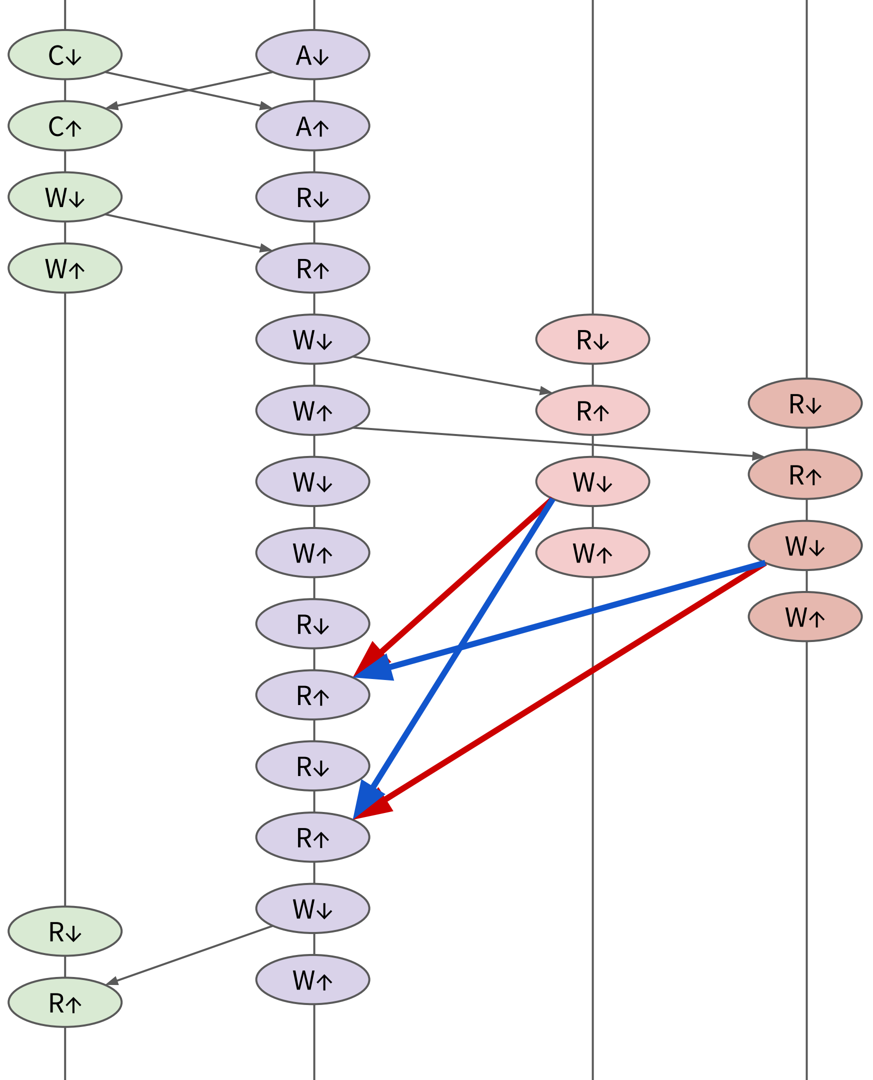


Key

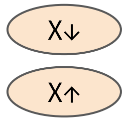
X↓	Syscall X entry
X↑	Syscall X return

W: write
R: read
C: connect
A: accept


C↓
C↑
W↓
W↑
R↓
R↑
A↓
A↑
R↓
R↑
W↓
W↑
W↓
W↑
R↓
R↑
R↓
R↑
W↓
W↑
R↓
R↑
W↓
W↑
R↓
R↑
W↓
W↑
Key
X↓ Syscall X entry
X↑ Syscall X return
W: write
R: read
C: connect
A: accept



Key



Syscall X entry

Syscall X return

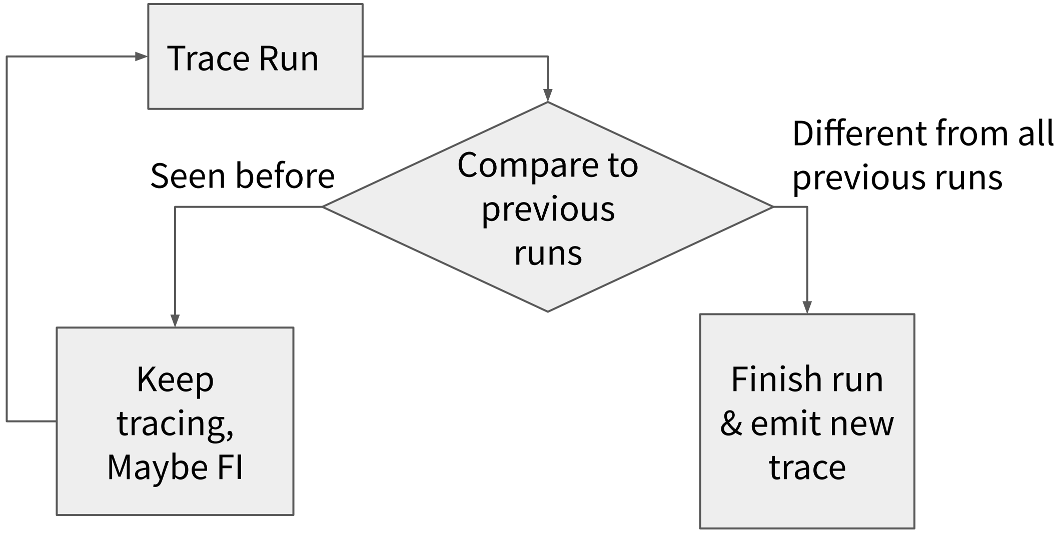
W: write
R: read
C: connect
A: accept

Box of Pain

Tracer

Tracker

Fault Injector



Box of Pain

Tracer

Tracker

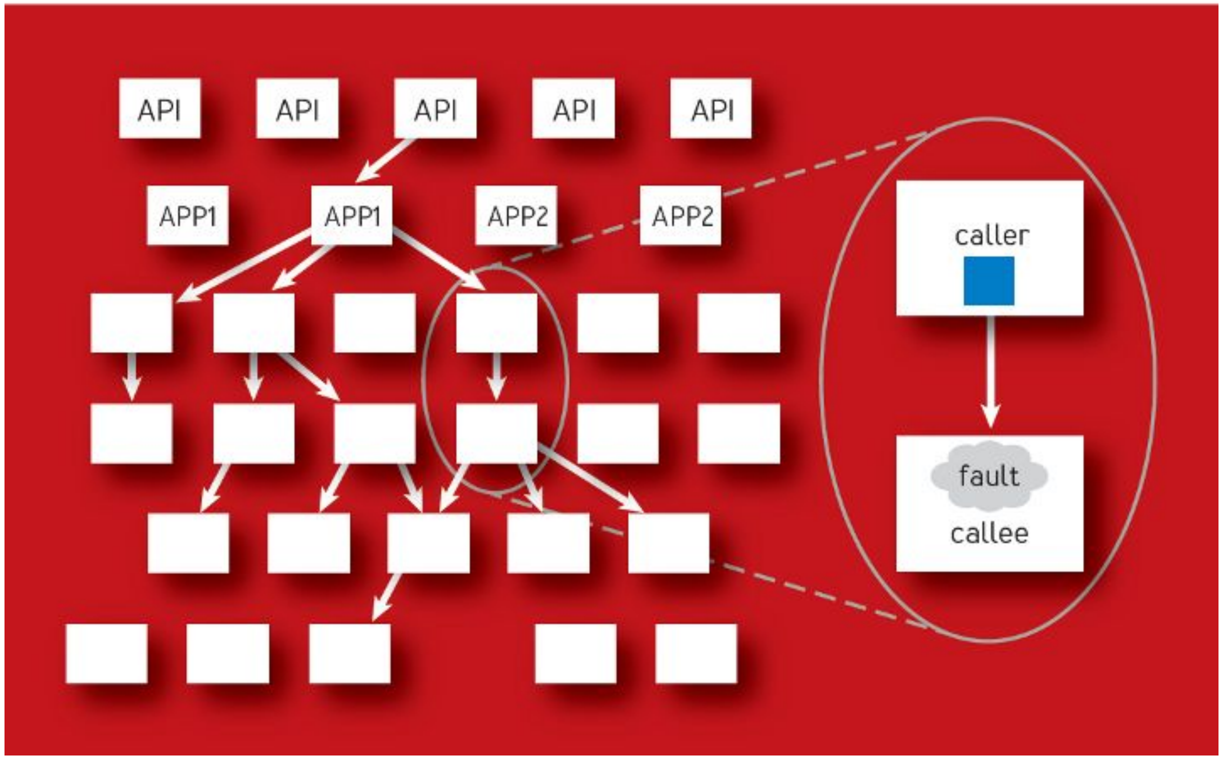
Fault Injector

Fault Injection

Philosophy:

Don't overthink it

Fault Injection



Fault Injection

Primitives:

(possibly infinite) delay

Explicit errors

Simulating real-world faults

Process crash:

prevent delivery of outgoing messages

Network partition:

prevent delivery of messages between pairs of processes

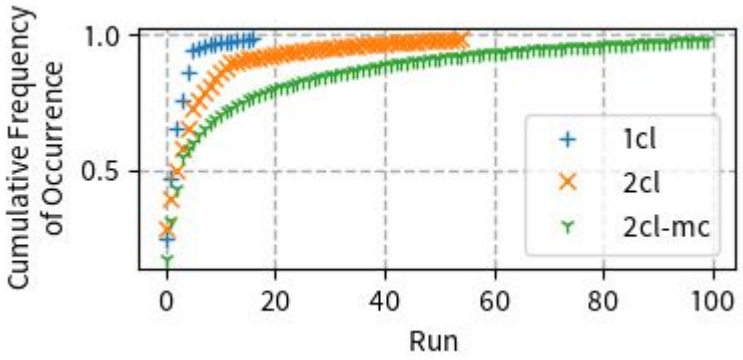
Nondeterminism and runs

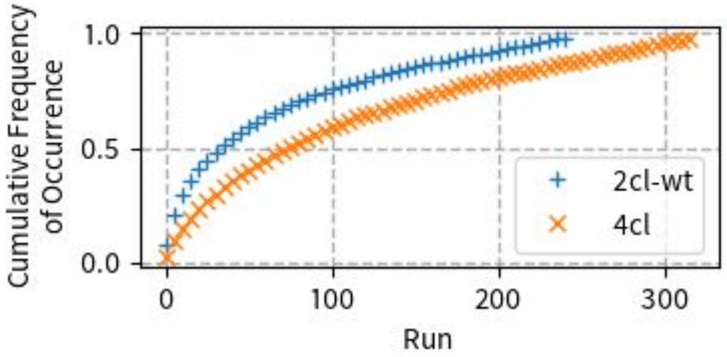
Philosophy:

Arbitrarily many possible traces

Some are much more common than others!

Preliminary Results: the tails are long





Next Step

Integrate with Lineage-driven Fault Injection

Key challenge: abstract representation of traces

Remember

Focus: rare but catastrophic "time of fault" bugs

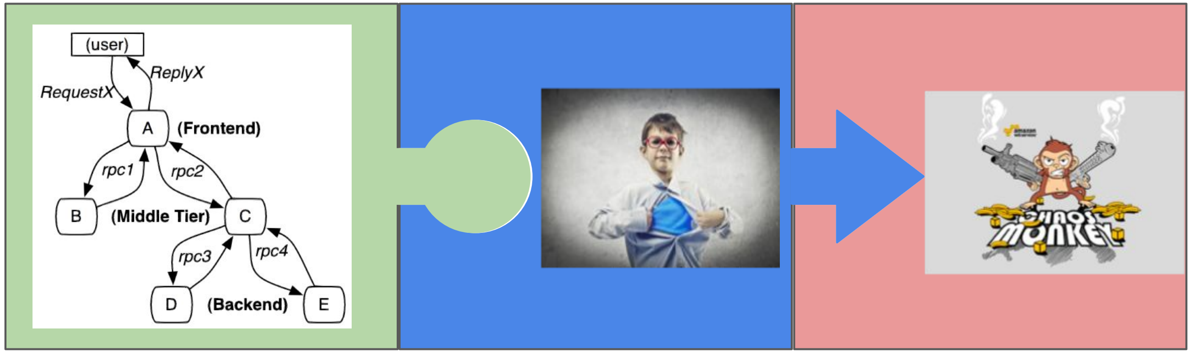
Coarse tracing: communication across failure boundaries

Simulation: *Effects* of faults rather than their causes

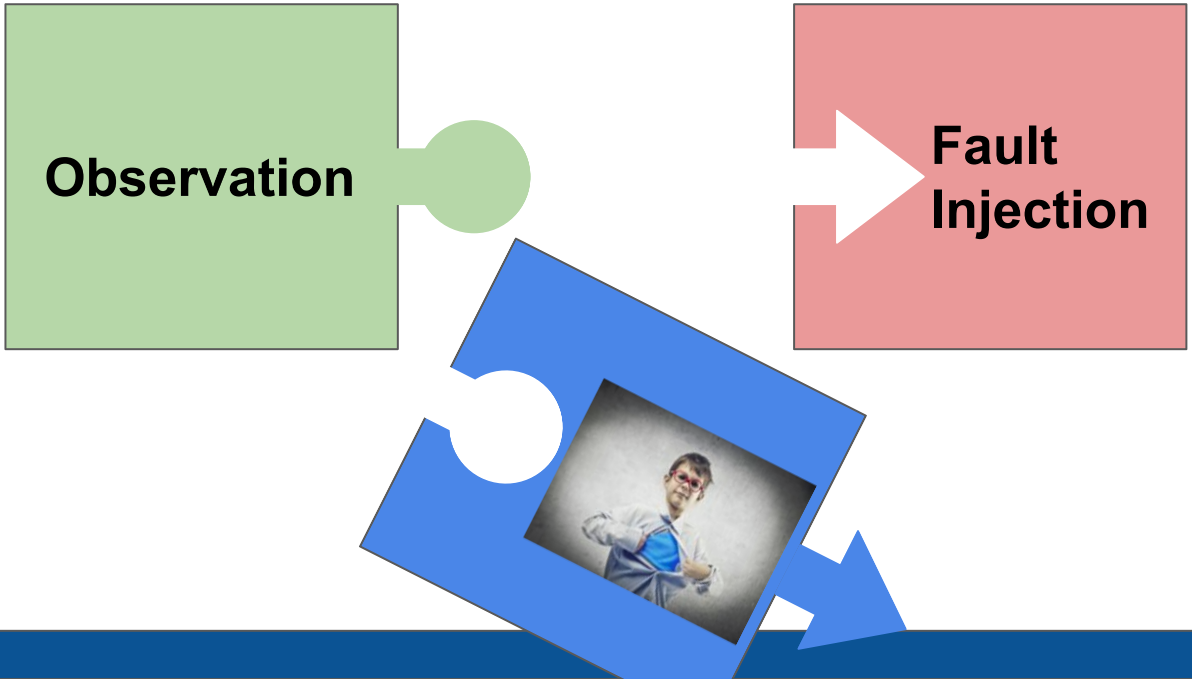
Pragmatism: many runs are possible; few are likely

Backup slides

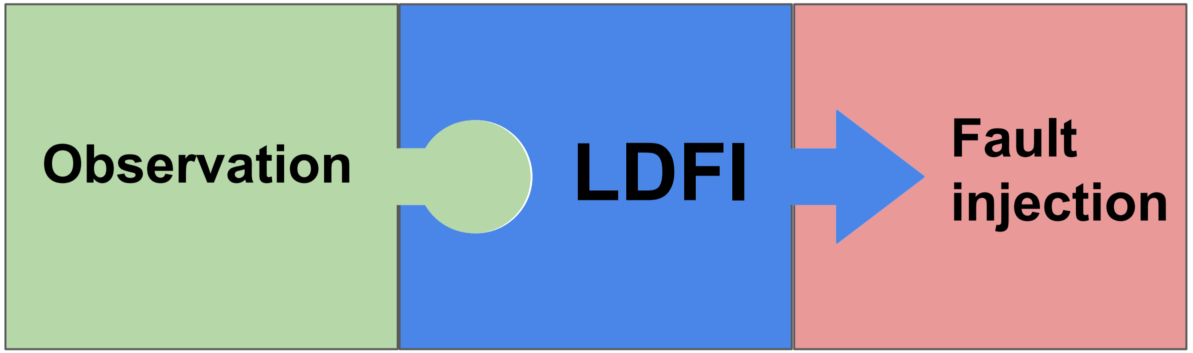
Real motivation



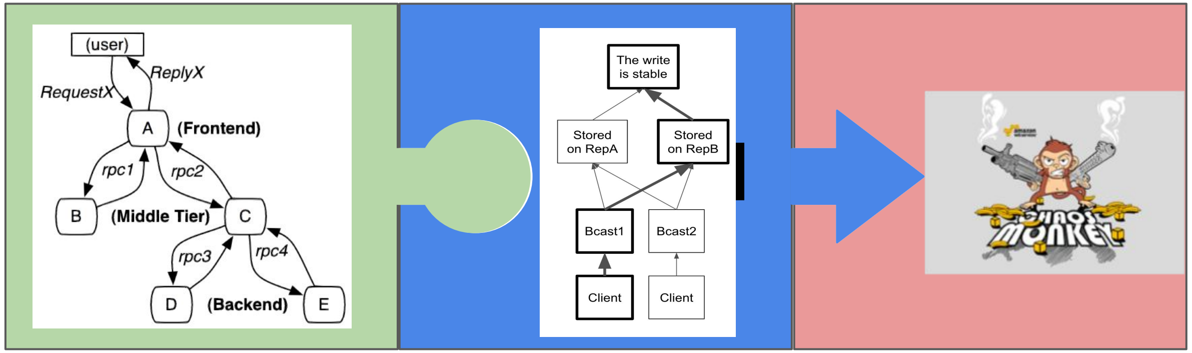
Real motivation



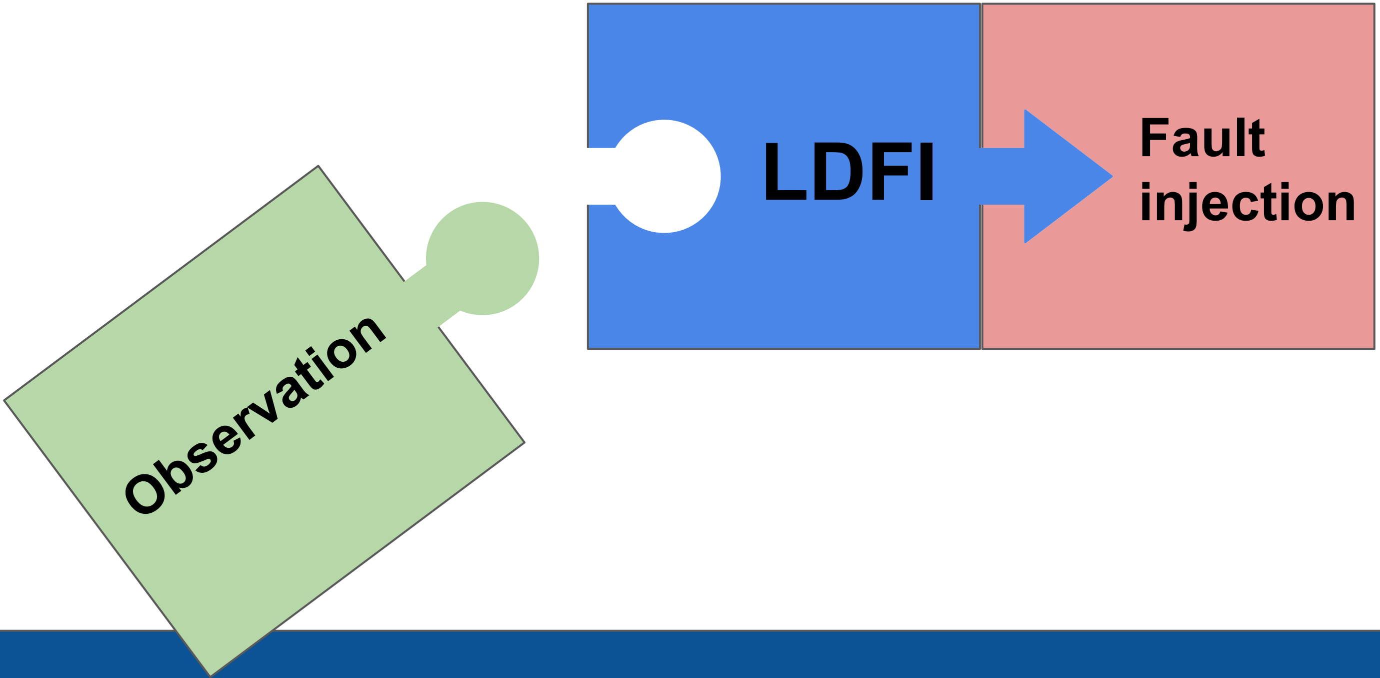
Real motivation



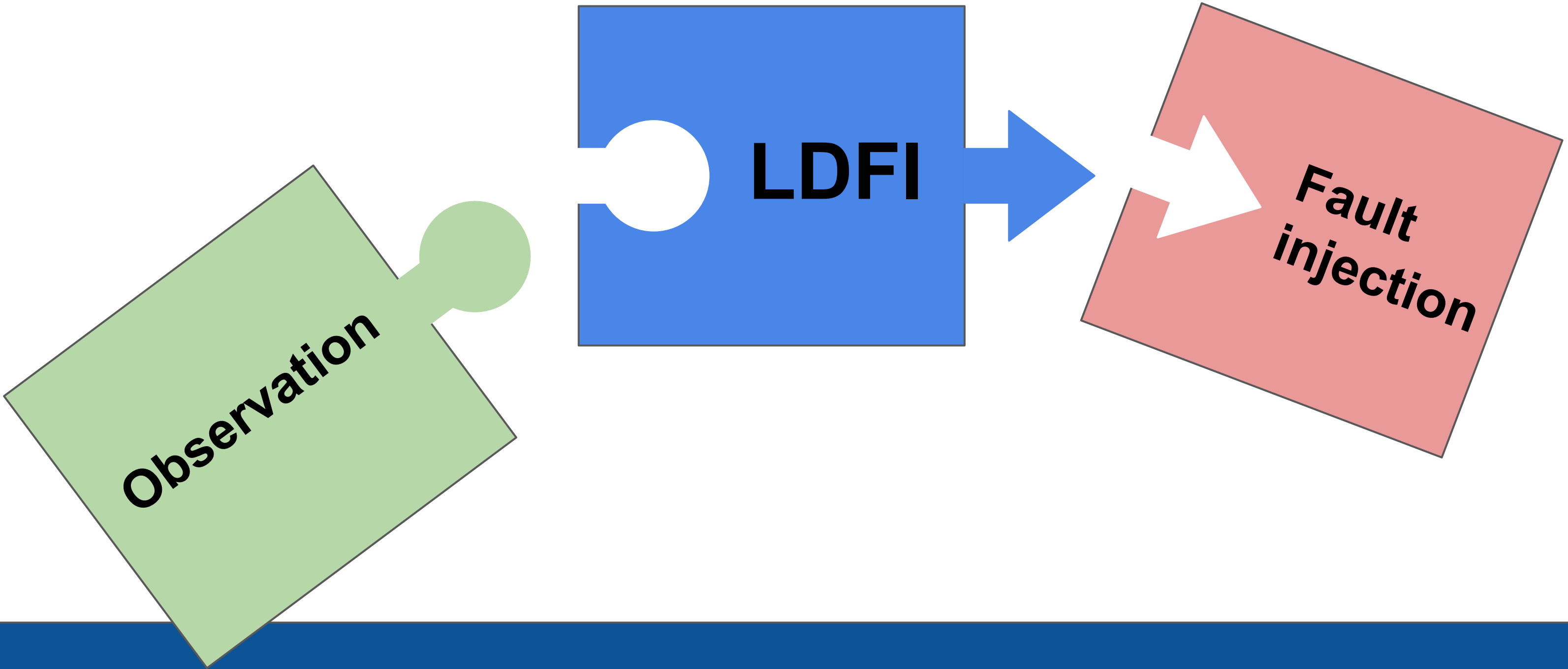
Real motivation



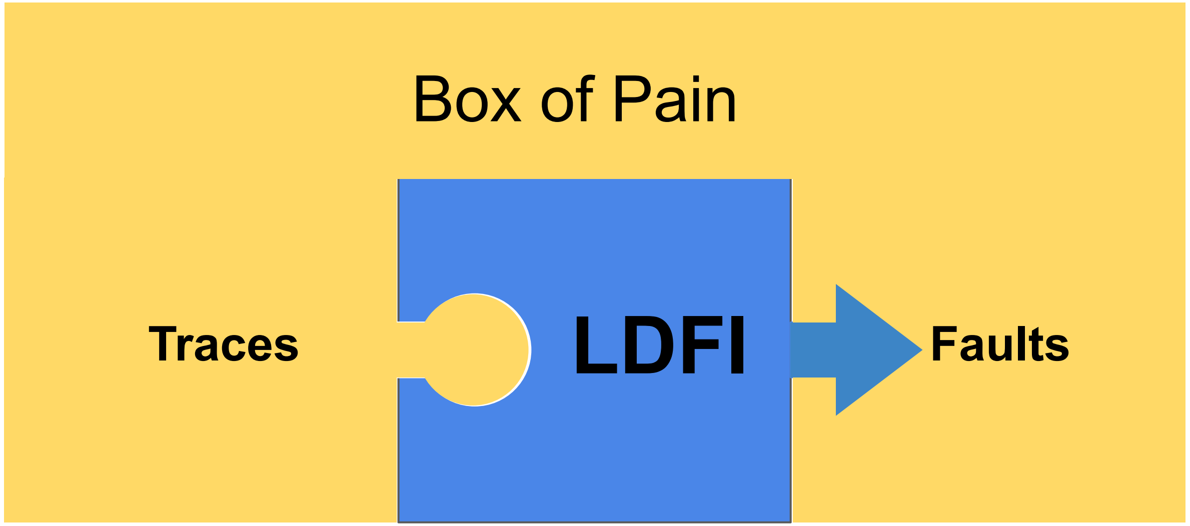
Real motivation



Real motivation



Real motivation


Box of Pain
Traces
LDFI
Faults