



Horapha – your personal warm bazel
hanwen@engflow.com

Who am I ?

Han-Wen Nienhuys – build system nerd

Worked on Bazel at Google (2013–2015)

Working at EngFlow now.



{fast, correct} – choose two

fast?

```
hanwen@hanwen-flow:~/vc/engflow/engflow$ time bazel build //dev/local
```

- Who wants to wait 15s?
- AI: many checkouts = many bazels
- CI: always runs from scratch

Can I fix this?

(CPU pinned @ 2Ghz)

What is taking time?

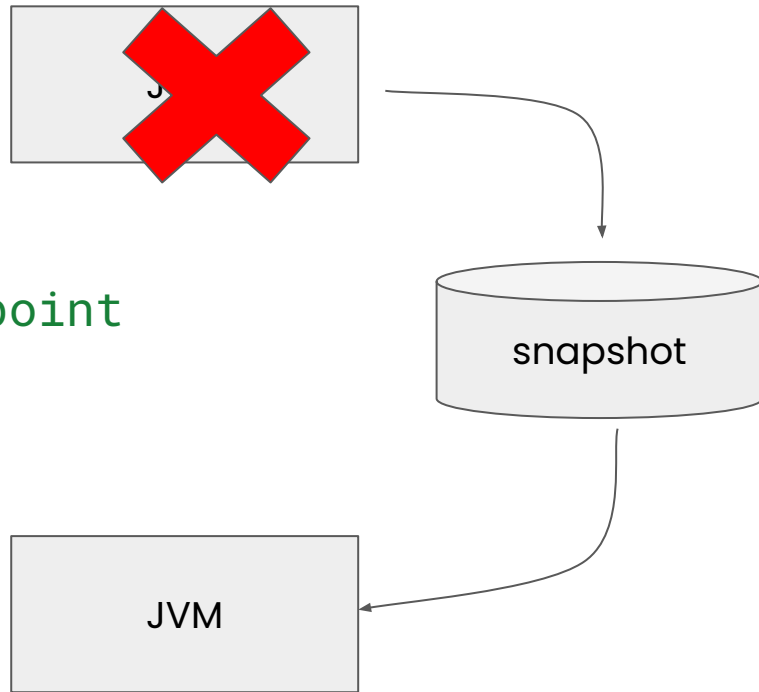
Download bazel	network , storage	
Unpack bazel	network , storage	these are on my machine already
Start JVM, JIT	CPU, RAM	
- read source - fetch deps - fetch rules	network, storage	
- interpret/hash	CPU, RAM	remember this from last time
compute target graph	CPU, RAM	
compute action graph	CPU, RAM	

Idea: speed up by dump memory

1. `bazel build //dev/local`
(slow)

2. `bazel checkpoint`
(slowish)

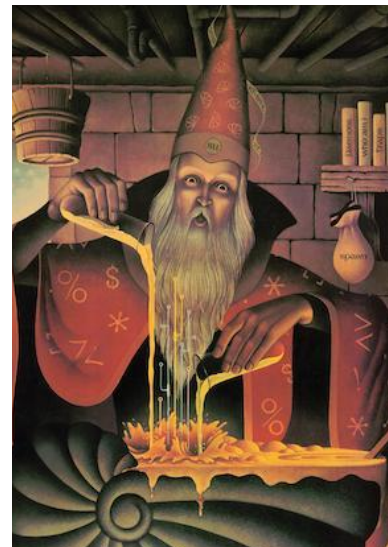
3. `bazel build ..`
(fast, frictionless)



Enter CRIU

- **C**heckpoint/**R**estore **I**n **U**erspace
- Linux-only technology
- Unix wizardry, but works well
- Lightweight (contrast: VM)
- **C**heckpoint a process to disk
- **R**estore a process from disk
 - create a specific PID
 - PID must be free
 - control PIDs: privileged
 - solution: namespaces

Conclusion: run bazel in namespace (“container”)



First try: horap'hā

A wrapper that

- sets up a container
 - user ns = root privileges
 - pid ns = target PIDs available
- starts Bazel in the container
- checkpoints/restores containers



Works! (<https://github.com/hanwen-flow/horapha/>)

Problem: Bazel & Horapha both want to manage the Bazel server

Conclusion: put into bazel launcher

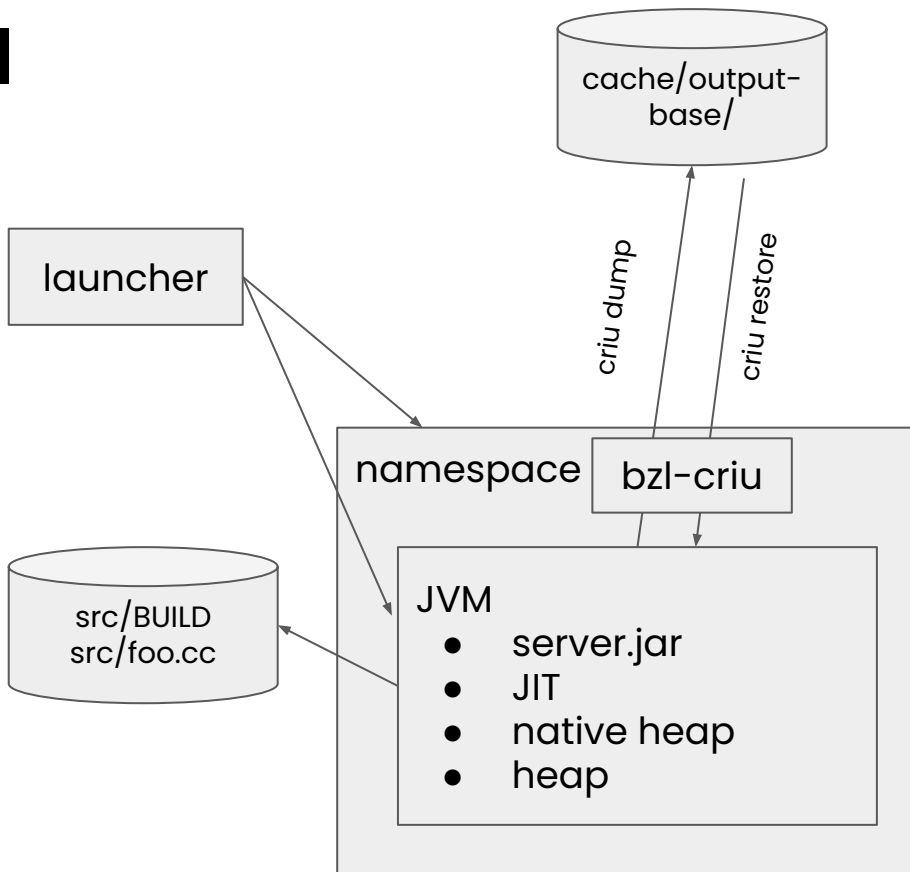
Second try: CRIU-bazel

Bazel:

- launcher
- server
- **output-base** admin area

CRIU integration:

- launcher creates namespace
- inside namespace, daemon calls CRIU
- snapshots into **output-base**
- Bazel server is untouched (*)



Results

State (early sept)

Bazel	20260727
CRIU	v4.2
Cold build:	~14s + 1s
Heap size:	5 Gbyte
with CRIU:	5s = 3x (~1 Gb/s)

Latest news (early Oct)

CRIU	4.3-dev (parallel lz4)
Bazel	20260902

```
hanwen@hanwen-flow:~/vc/engflow/engflow$ time bazel-dev --criu build //dev/local
Extracting Bazel installation...
Starting local Bazel server (no version) and connecting to it...
INFO: Invocation ID: ab682eeb-186c-4430-86d4-ce0c81358a73
INFO: Analyzed target //dev/local:local (1460 packages loaded, 28,690 targets configured, 2
479 aspect applications).
INFO: Found 1 target...
Target //dev/local:local up-to-date:
  bazel-bin/dev/local/local
INFO: Elapsed time: 16.274s, Critical Path: 0.51s
INFO: 1 process: 4969 action cache hit, 1 internal.
INFO: Build completed successfully, 1 total action

real    0m16.483s
user    0m2.443s
sys     0m0.932s
hanwen@hanwen-flow:~/vc/engflow/engflow$ time bazel-dev --criu checkpoint
```

restore:	425ms = 33x faster
heap size:	~1 Gb
	changed GC behavior (5x)
speed	~2 Gb/s
	CRIU 4.3 parallel lz4 (2x)

Evaluation/Conclusions

- Does it work?
 - yes: **up to 30x faster**
 - CRIU cost = $O(\text{memory size})$
- Can we put this in Bazel?
 - Client/server management is messy. refactor?
 - Linux specific - maybe not?
- Code
 - github.com/hanwen-flow/horapha
 - github.com/hanwen-flow/bazel (horapha branch)
- Doc:
 - ["Horapha" - Bazel with CRIU snapshotting](#)