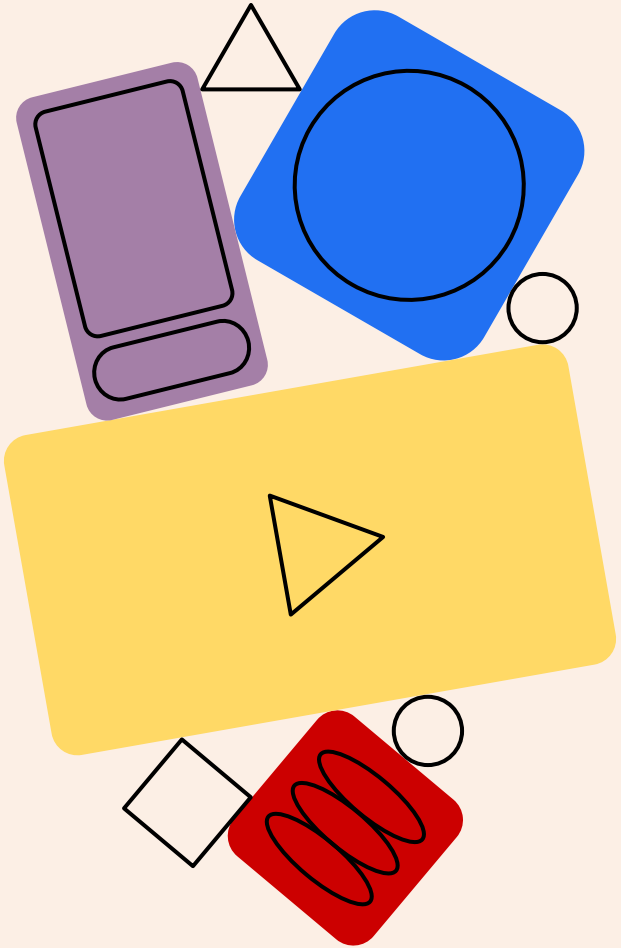
Advanced Build Tools & Remote Build Execution API



About me

Son Luong Ngoc

- From Vietnam
- Based in The Netherlands

OSS contributions

- Git, Gitlab, Gitaly
- Bazel, rules_go, buck2

Work

- Current:
  - Solution Engineer at BuildBuddy
- Past:
  - SRE at Lazada, Alibaba
  - Developer Productivity at Booking.com

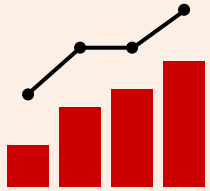
About me (excuses)

- I have traveled to 6-7 cities in the last 1 week.
- Extreme Jetlag
- Completed this slides at 4-6AM this morning

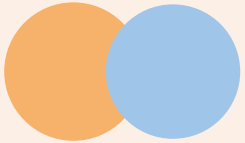
Contact me:

- @sluongng on most social media
- sluongng <at> <google mail service> dot com
- BazelBuild Slack

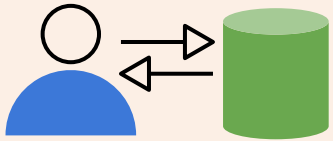
Table of Contents



CI Challenges at Scale

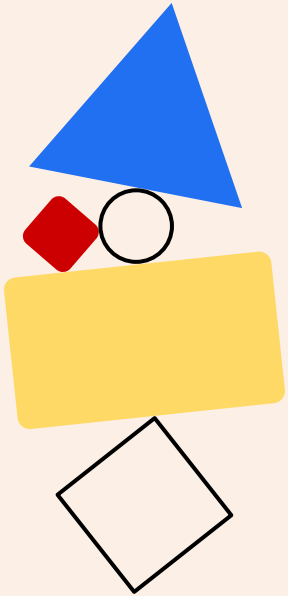


Artifact-First and Hermetic Builds



Distributed Build Systems and Remote Build Execution APIs

CI at Scale



What is “Scale”?

1. **Having X * 10^{2,3,4} of employees.**

Slower builds cost more in working hours.

2. **Having “Big Code” issues**

Forking VCS / Forges to solve performance issues.

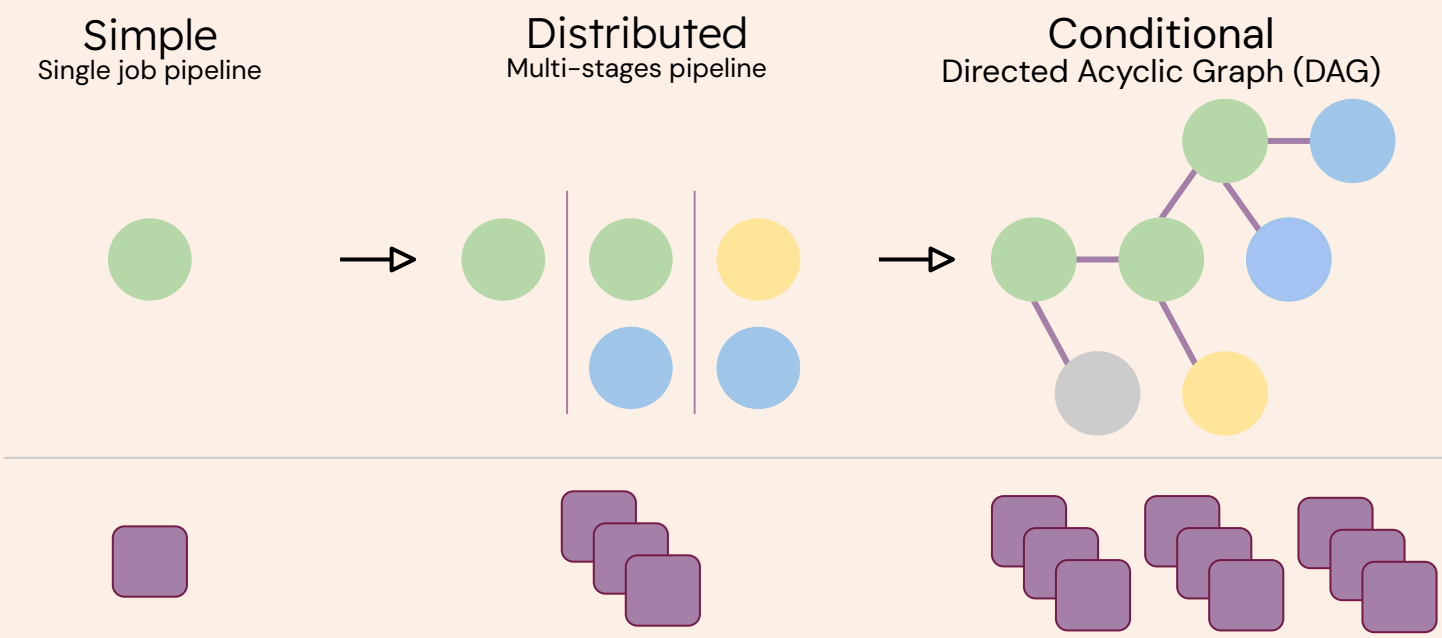
3. **Lowering risk tolerance**

Wanting to improve reliability, security, shift left.

➔ Increase in testing, verification between components and modules.

Challenges of CI at Scale

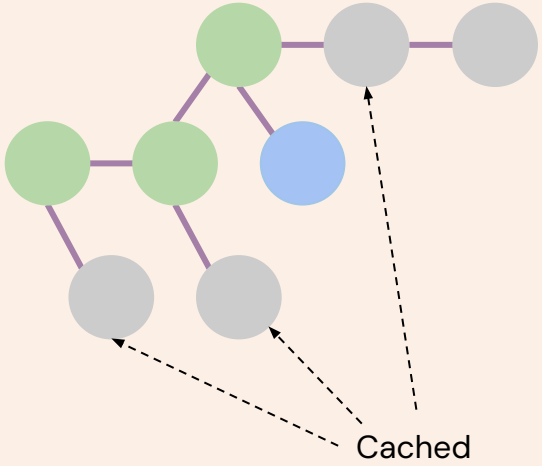
It’s a Compute problem



Challenges of CI at Scale

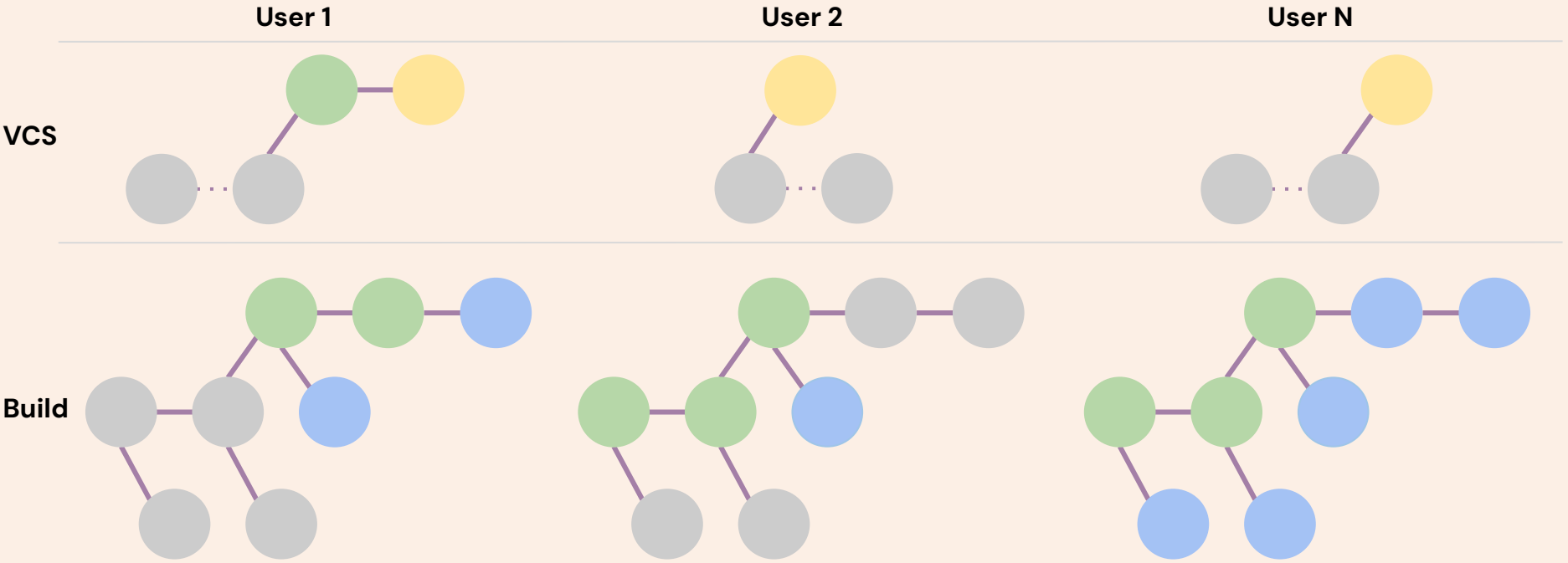
It’s a Cache (Invalidation) problem

```
docs.go                  | 2 +-
cmd/cli/main.go          | 2 +-
pkgs/hash/hasher.go      | 1 +
pkgs/hash/hasher_test.go | 1 +
pkgs/cast/cast.go        | 2 --
pkgs/cast/cast_test.go   | 9 ---------
```



Challenges of CI at Scale

It’s still a compute problem with Cache



Challenges of CI at Scale

Recap:

Big Code

Many code drives up the CI compute need

Cache Invalidation

Caching can help you speed up CI
But cache invalidation is a hard problem

Dependency Tracking

Helps drive better cache invalidation and selective building/testing
Requires a fine-grained graph

Distributed Computing

Multiple users creating multiple compute graphs require deduplication between graph to help save compute.
Smart scheduling can help maximizing local cache hits.

Morden Build Tools

Matured:

- Bazel
- Buck(2)
- Pants(2)

Getting there:

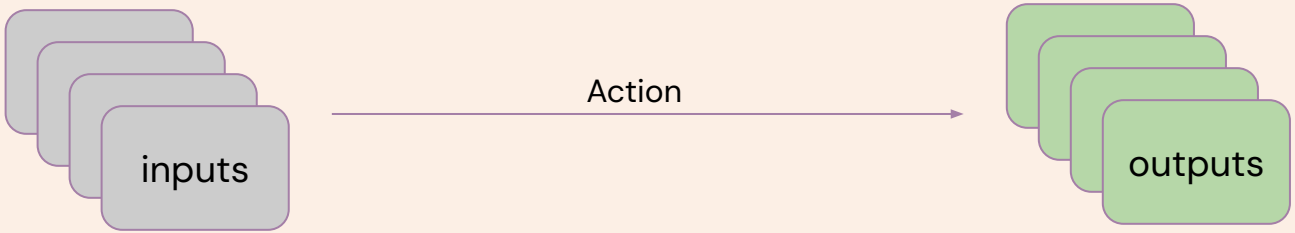
- Nix, Mill, Dune, JustBuild, distcc, ReClient, ...

Artifact-first Build Tools

- Build should be used for producing artifacts.
- If the correct artifacts already exist, don't run.

- Good: make caching easier
- Bad: "side effect" tasks such as releases, deployments are not handled.

Action and Result



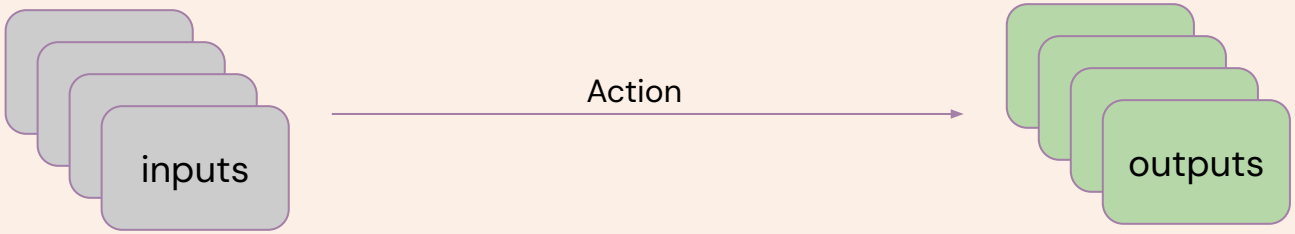
Input digests (<sha256>/<size>):
- Source files:
  - a.cc
  - b.cc
- Tool files:
  - gcc
  - ld

Command:
- “gcc ... a.cc b.cc”
- Environment variables

Platform information:
- Target CPU, OS...

Outputs:
- Stdout, stderr
- Build artifacts (directories, files)

Action and Result (cont.)



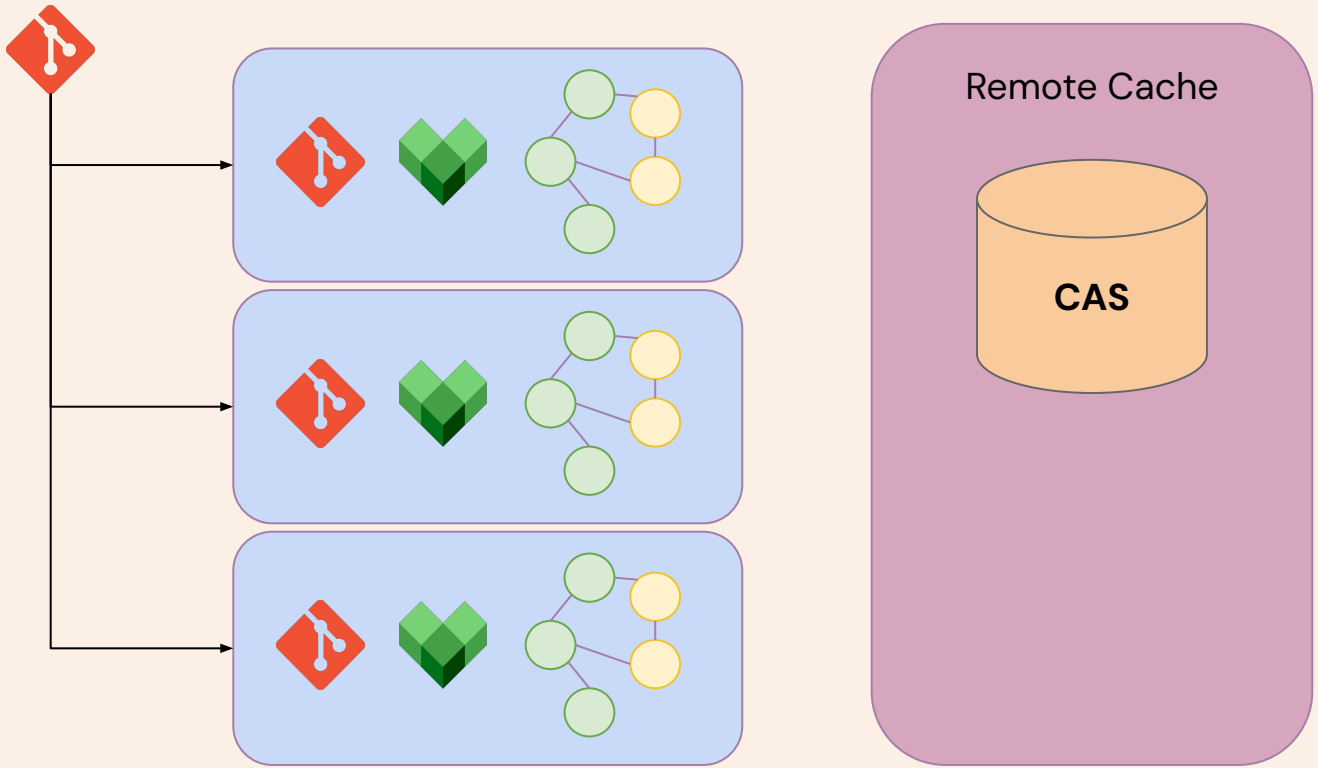
Input digests (<sha256>/<size>):
- Source files:
  - a.cc: aaabbb/12345
  - b.cc: cccddd/21345
- Tool files:
  - gcc: eeeggg/21234
  - ld: hhhccc/23231

Command: ➡️bbbeee/1234
- “gcc ... a.cc b.cc”
- Environment variables

Platform information:
- cpu, os...

Outputs:
- Stdout a1eefb/7789
- Stderr b12acx/321
- Build artifacts (directories, files)

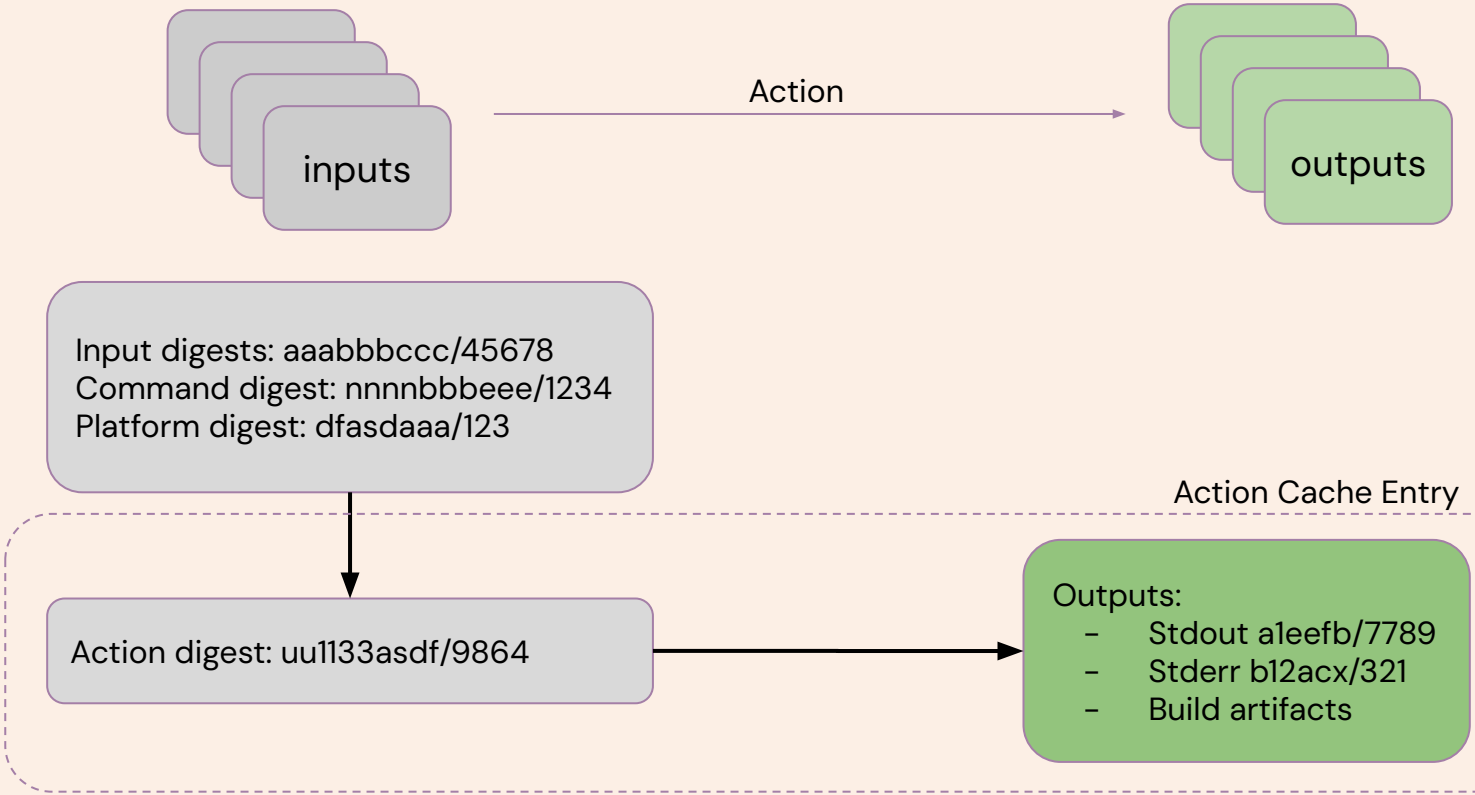
Example CI with Build Tool



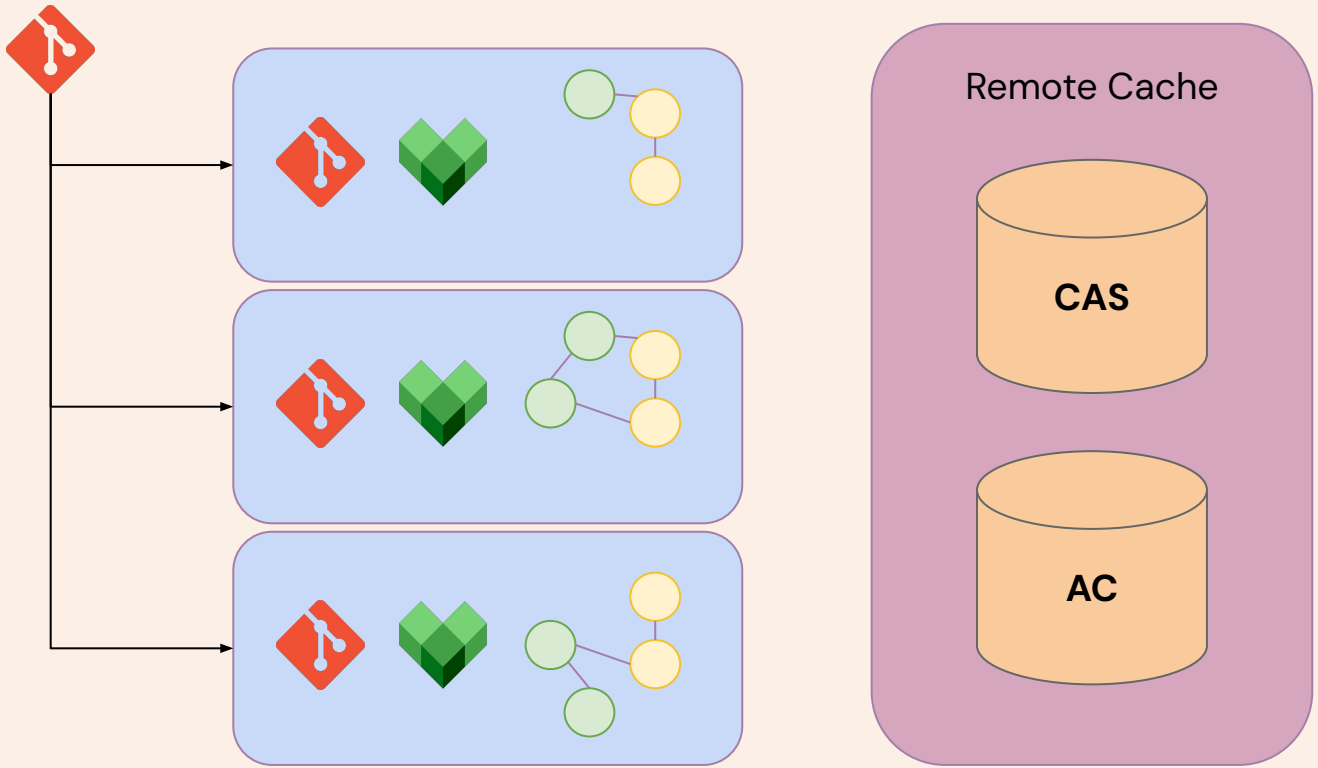
Hermetic Actions

- Isolate execution environments improve action reproducibility
- With source identity (hashes), changes can be accurately identify
- Greatly improve Reproducibility and Caching

Action Caching



Example CI with Action Cache

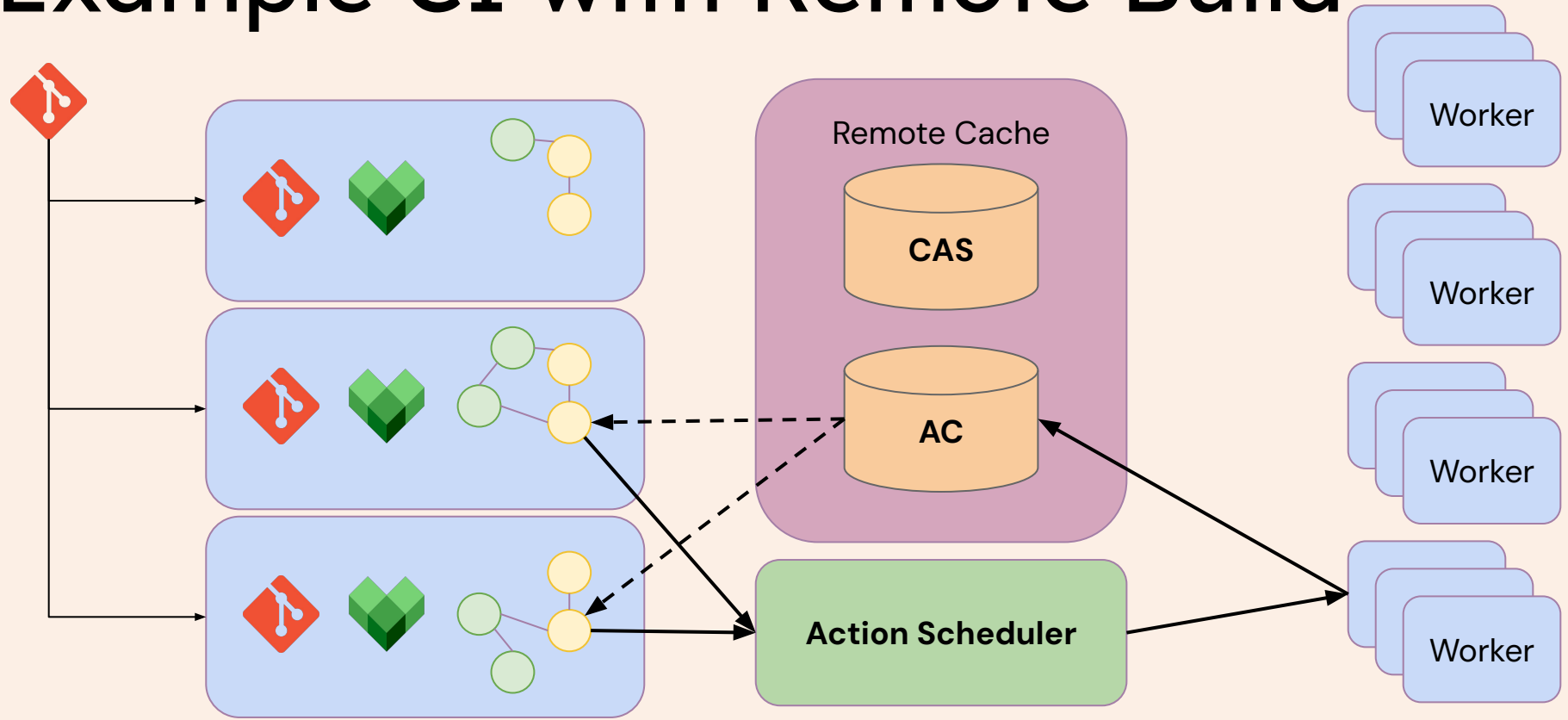


Hermetic Actions (cont.)

- Hermeticity not a binary property but more like a spectrum
- Network, CPU, Memory, filesystem, OS isolation can be applied as needed

- Observation: Reproducible Actions can run anywhere

Example CI with Remote Build



Remote Build Execution API

- Define how different build tools (clients) can interact with Remote Cache and Remote Build systems (server)
- Remote Cache API is HTTP and GRPC
- Remote Build API is GRPC

Remote Build Execution API

```
service ByteStream {
  rpc Read(...) returns (stream ...)

  rpc Write(stream ...) returns (...)
}
```

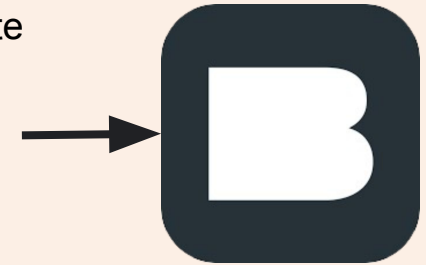
```
service ContentAddressableStorage {
  rpc BatchUpdateBlobs(...) returns (...)

  rpc BatchReadBlobs(...) returns (...)

  rpc FindMissingBlobs(...) returns (...)

  rpc GetTree(...) returns (stream ...)
}
```

```
service ActionCache {
  rpc GetActionResult(...) returns (...)

  rpc UpdateActionResult(...) returns (...)
}
```

```
service Execution {
  rpc Execute(...) returns (stream ...)

  rpc WaitExecution(...) returns (stream ...)
}
```

Remote Build API impls

- Clients:
  - Bazel
  - Buck2a
  - Pants
  - BuildStream
  - Recc
  - PleaseBuild
  - JustBuild
  - Siso
  - ReClient:
    - Replaces Goma
    - GN/Ninja integration
    - CMake integration
  - Dune (planned)
- Servers:
  - Self-service
    - BuildBarn
    - BuildFarm
    - BuildGrid
    - Bazel-Remote
  - Commercial
    - BuildBuddy
    - Aspect Build
    - Bitrise
    - EngFlow
    - NativeLink
    - Develocity



Remote Build API: TODOs

- Do not capture Windows well:
  - Still *work,* just not popular and may come with edge cases
  - Microsoft BuildXL uses a deviation of Remote APIs
  - Unreal Engine is creating their own Remote Build system
- Support for massive blobs is WIP
- Missing standardization for Build Telemetry.