Remote Build Execution for kernel developers

David Brazdil <david@source.dev>

Linux Plumbers Conference

Vienna, Sep 2024

Who am I

Previously Engineer @ Google

- Linux contributor: pKVM for arm64
- Android Platform Security
- ART compiler

Now CTO @ source.dev

- Tooling for software maintenance of Android OS
- Driven by upcoming EU regulation - 7 years of support
- Exploring integration with KernelCI
- Fast, cheap builds enabler for automation

What is RBE

Bazel

- Open-source build system
- Scalable to very large codebases
- Hermetic and sandboxed by default
- Caches shareable across team/org

Remote Build Execution

- gRPC protocol for distributed builds and tests
- Scalable to datacenter levels
- Offload heavy computation from laptops and workstations to servers
- Consistent execution environment for developers

What's in it for *me*?

RBE infrastructure is widely available

- Open-source backends
- Self-hosted solutions
- Fully managed commercial offerings
- Easy to switch between providers, avoiding lock-in

Non-Bazel build systems supported

- Pants, Buck2 implement RBE natively
- reclient, buildbox: wrappers for common compilers, including GCC and Clang

Performance!

- ~60% on x86 defconfig

Open-source backends

	Licence	Language	Deployment
BuildBarn[1]	Apache-2.0	Go	Kubernetes, Compose
BuildFarm[2]	Apache-2.0	Java	Kubernetes, Helm
BuildGrid[3]	Apache-2.0	Python	Compose
NativeLink[4]	Apache-2.0	Rust	Kubernetes
BuildBuddy[5]	MIT Expat *	Go	Terraform

* open-core model, remaining features under enterprise license

1 https://github.com/buildbarn

2 https://github.com/bazelbuild/bazel-buildfarm

3 https://gitlab.com/BuildGrid/buildgrid

4 https://github.com/TraceMachina/nativelink

5 https://github.com/buildbuddy-io/buildbuddy

more: https://bazel.build/community/remote-execution-services

RBE protocol

Basic operations

```
service ContentAddressableStorage {
  rpc FindMissingBlobs(FindMissingBlobsRequest)
    returns (FindMissingBlobsResponse) {...}

  rpc BatchUpdateBlobs(BatchUpdateBlobsRequest)
    returns (BatchUpdateBlobsResponse) {...}

  rpc BatchReadBlobs(BatchReadBlobsRequest)
    returns (BatchReadBlobsResponse) {...}
}
```

https://github.com/bazelbuild/remote-apis

RBE protocol

Describing a build step

```
message Command {
  repeated string arguments;
  repeated EnvironmentVariable environment_variables;
  repeated string output_paths;
  ...
}

message Directory {
  repeated FileNode files;             // list of blobs
  repeated DirectoryNode directories; // list of blobs
  repeated SymlinkNode symlinks;       // list of paths
  ...
}

message Action {
  Digest command_digest;     // Command blob
  Digest input_root_digest;  // Directory blob
  ...
}
```

https://github.com/bazelbuild/remote-apis

RBE protocol

Caching build steps

```
message ActionResult {
  int32 exit_code;
  Digest stdout_digest;
  Digest stderr_digest;
  repeated OutputFile output_files;
  ...
}

service ActionCache {
  rpc GetActionResult(GetActionResultRequest)
    returns (ActionResult) {...}

  rpc UpdateActionResult(UpdateActionResultRequest)
    returns (ActionResult) {...}
}
```

https://github.com/bazelbuild/remote-apis

RBE protocol

Remote execution

```
message ExecuteRequest {
  Digest action_digest;
  bool skip_cache_lookup;     // force execution
  ...                         // hints to the backend
}

message ExecuteResponse {
  ActionResult result;
  ...
}

service Execution {
  // Returns a message stream that will eventually
  // yield an ExecuteResponse message.
  rpc Execute(ExecuteRequest)
    returns (stream Operation) {...}
}
```

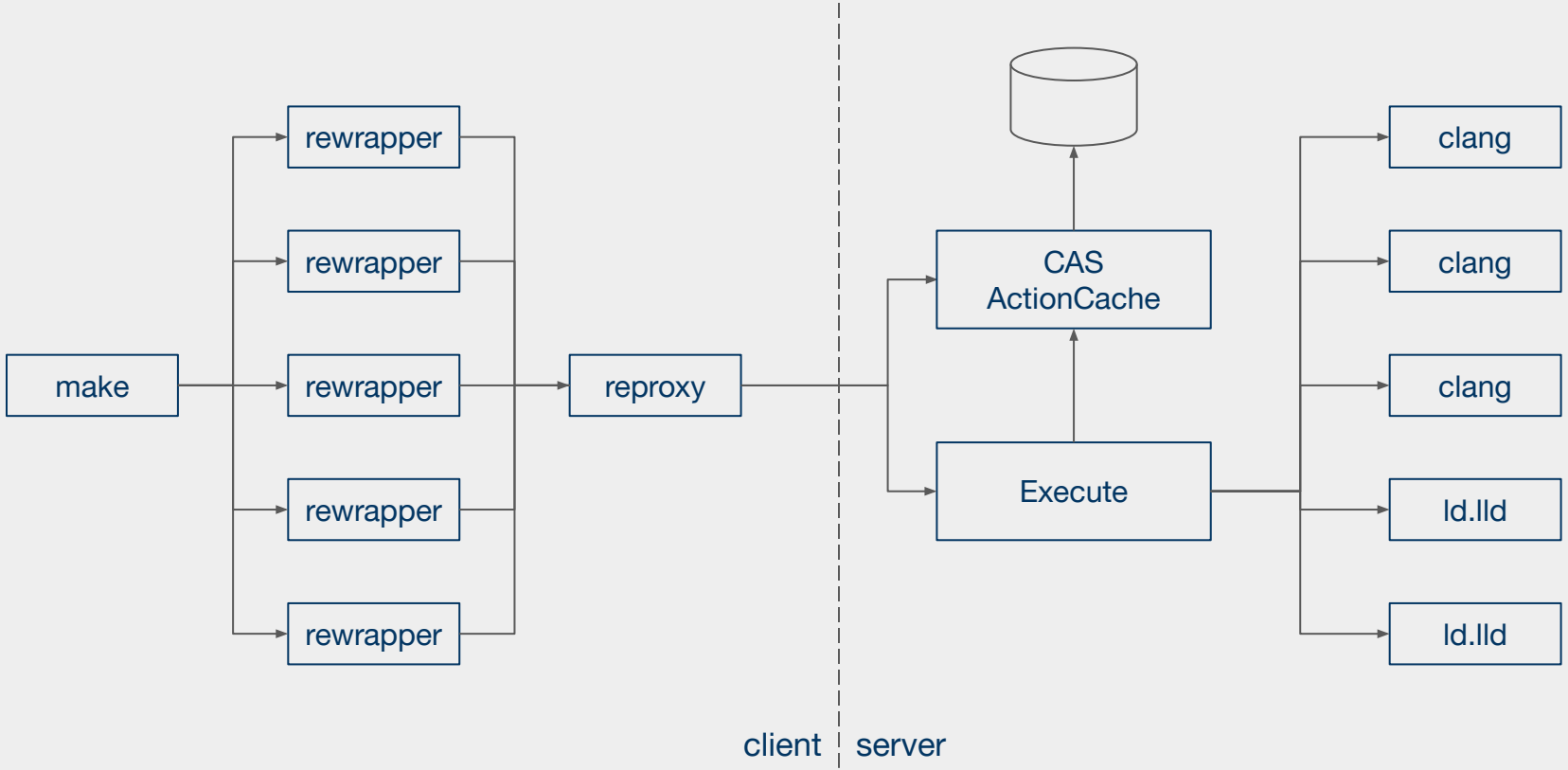
https://github.com/bazelbuild/remote-apis

reclient

- Part of the Bazel project, Apache-2.0
  - Initially developed as a replacement for Goma (deprecated)
- **rewrapper** - wrapper around common compilers
  - gcc, clang, javac, metalava, d8, r8, typescript, …
  - Determines the input/output fileset
    … by parsing the command line arguments
    … by processing dependencies (eg. header files, libs)
- **reproxy** - local / remote worker pool manager
  - Configurable strategies for optimizing performance
  - default: `remote_local_fallback`
  - good for devs: `racing`

https://github.com/bazelbuild/reclient

Architecture



Building the kernel

Start backend

Start an RBE backend on the local machine

- best to review paths/sizes in basic_cas.json
- Docker images are also provided

```
$ git clone https://github.com/TraceMachina/nativelink
$ cd nativelink
$ bazelisk run nativelink -- \
    nativelink-config/examples/basic_cas.json
```

https://github.com/bazelbuild/bazelisk

Building the kernel

Start frontend

Compile reclient

```
$ git clone https://github.com/bazelbuild/reclient
$ cd reclient
$ bazelisk run --config=clangscandeps   \
               //:artifacts_install --  \
               --destdir ./dist
```

Start reproxy

```
$ source <lpc2024>/envsetup
$ dist/bootstrap -re_proxy=$PWD/dist/reproxy \
                 -cfg=<lpc2024>/reproxy.cfg
```

https://github.com/bazelbuild/bazelisk

Building the kernel

Run build

Compile the kernel

```
$ cd linux
$ source <lpc2024>/envsetup
$ make LLVM=1 CC=<lpc2024>/cc LD=<lpc2024>/ld defconfig
$ make LLVM=1 CC=<lpc2024>/cc LD=<lpc2024>/ld -j32
```

Observe progress in reproxy

- expect to see errors from Kconfig probing

```
$ <reclient>/dist/reproxystatus
Reproxy(unix:///tmp/reproxy.sock) OK
Actions completed: 35 (10 cache hits, 25 remote executions)
Actions in progress: 32
```

The ugly

- Builds need to be reproducible-ish
  - `CONFIG_RANDSTRUCT=y` needs a constant seed
  - `Documentation/kbuild/reproducible-builds.rst`
- Kconfig passes inputs/outputs via stdin/stdout
  - This is not supported by the RBE protocol → run locally
- Dependency scanner does not recognize `.incbin` in assembly
  - Harder to fix for inline assembly
  - However, heuristic to filter out such source files is cheap
- Linux uses compiler flags not recognized by reclient (yet)
  - eg. `-Wp,-MD,<depfile>`, expects linker invoked via `clang`
  - Bit of a neverending whack-a-mole game
  - Our team is developing a frontend which doesn't suffer from this problem 🤫

Questions?

Thank you!

Try it out: https://gitlab.com/sourcedotdev/lpc2024