Zero-sysroot hermetic LLVM cross-compilation using Bazel

By Corentin Kerisit & David Zbarsky

https://github.com/cerisier/toolchains_llvm_bootstrapped

Motivation

01 cross-compilation is arcane

02 setup steps suck

03 hermeticity unlocks new capabilities

https://github.com/cerisier/toolchains_llvm_bootstrapped

Status quo

https://github.com/cerisier/toolchains_llvm_bootstrapped

Existing approaches

https://github.com/cerisier/toolchains_llvm_bootstrapped

Existing approaches

Sysroots, docker, prebuilts env

dockcross

Cross compiling toolchains in Docker images

CMake ★ 3.6k 422

osxcross

MacOS Cross-Toolchain for Linux and *BSD

C++ ★ 3.3k 350

xx

Dockerfile cross-compilation helpers

Shell ★ 535 39

cross

“Zero setup” cross compilation and “cross testing” of Rust crates

Rust ★ 8.0k 442

https://github.com/cerisier/toolchains_llvm_bootstrapped

Existing approaches

Source based

buildroot

Buildroot, making embedded Linux easy. Note that this is not the official repository, but only a mirror. The official Git repository is at https://gitlab.com/buildroot.org/buildroot/. Do not open issues or file pull requests here.

Makefile ★ 3.3k 2.7k

crosstool-ng

A versatile (cross-)toolchain generator.

Shell ★ 2.4k 729

llvm-mingw

An LLVM/Clang/LLD based mingw-w64 toolchain

C ★ 2.7k 238

https://github.com/cerisier/toolchains_llvm_bootstrapped

Existing approaches

Zig cc

https://github.com/cerisier/toolchains_llvm_bootstrapped

Existing approaches

Nix

pkgsCross.*

https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

Requirements

Cross-compiling
everything from source

The hermetic approach

Requirements

01 An LLVM prebuilt toolchain

02 The LLVM source code

03 All other runtimes' source code

04  Bazel

https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

Requirements

{ Fast, Correct } — Choose two

https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

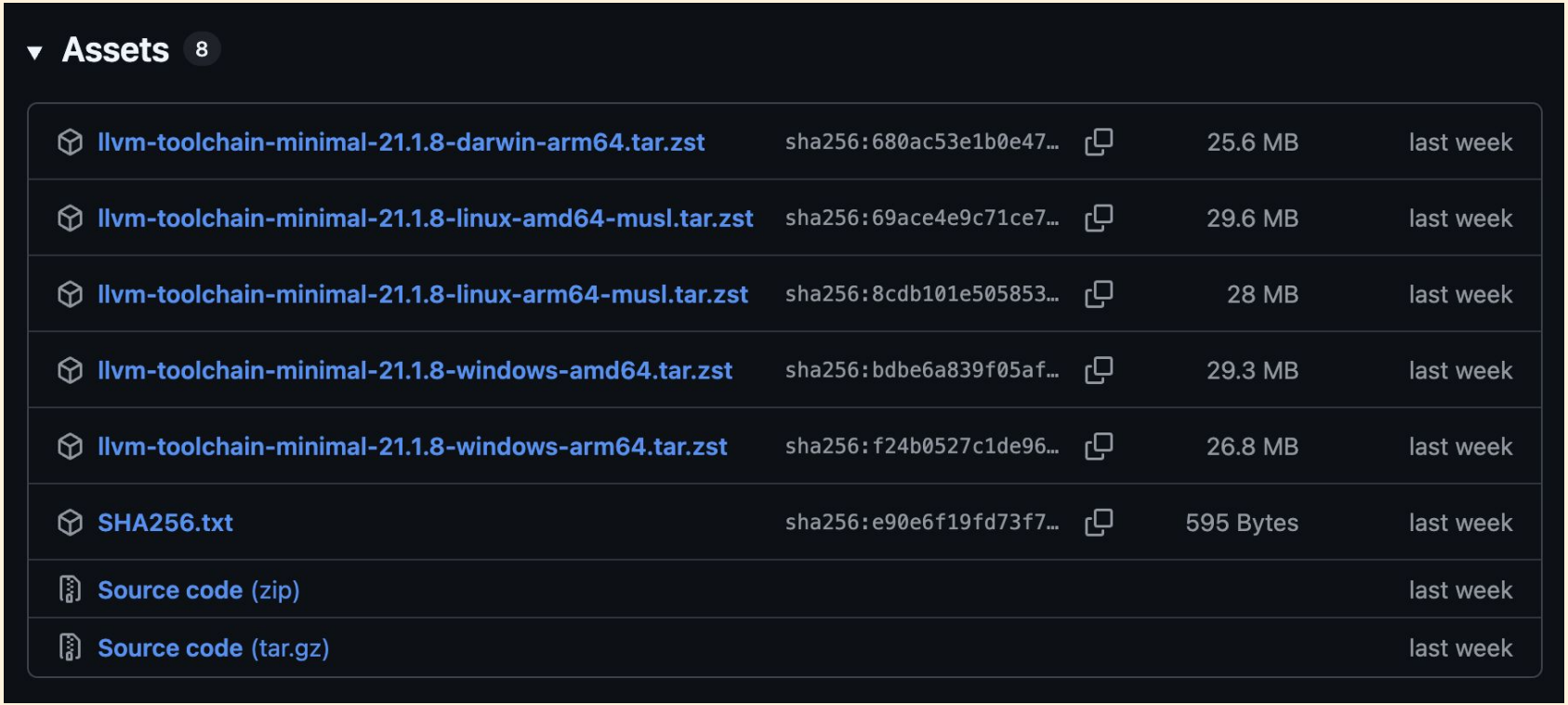
LLVM Driver

```
OVERVIEW: llvm toolchain driver

USAGE: llvm [subcommand] [options]

SUBCOMMANDS:

  objcopy
  strip
  nm
  ar
  ranlib
  lib
  dlltool
  lld
  clang
  clang++
  ...more...

  Type "llvm <subcommand> --help" to get more help on a specific subcommand

OPTIONS:

  --help - Display this message
```

https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

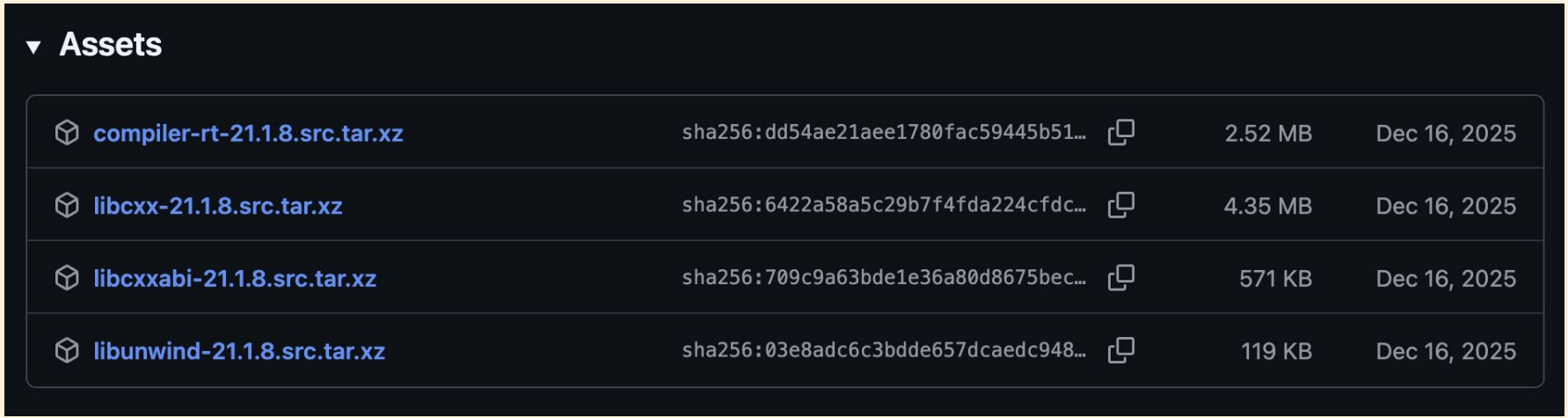
Prebuilts



https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

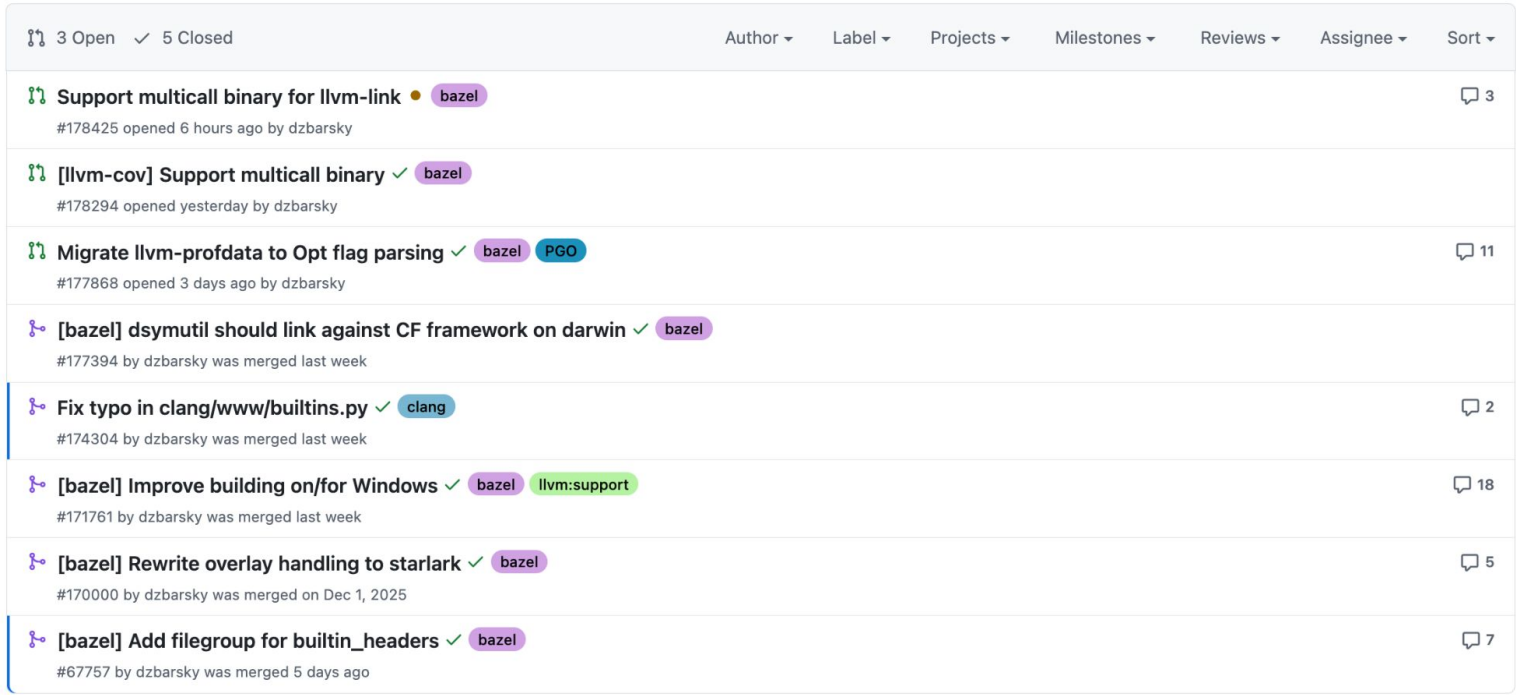
Runtimes



https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

Upstreaming changes



https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

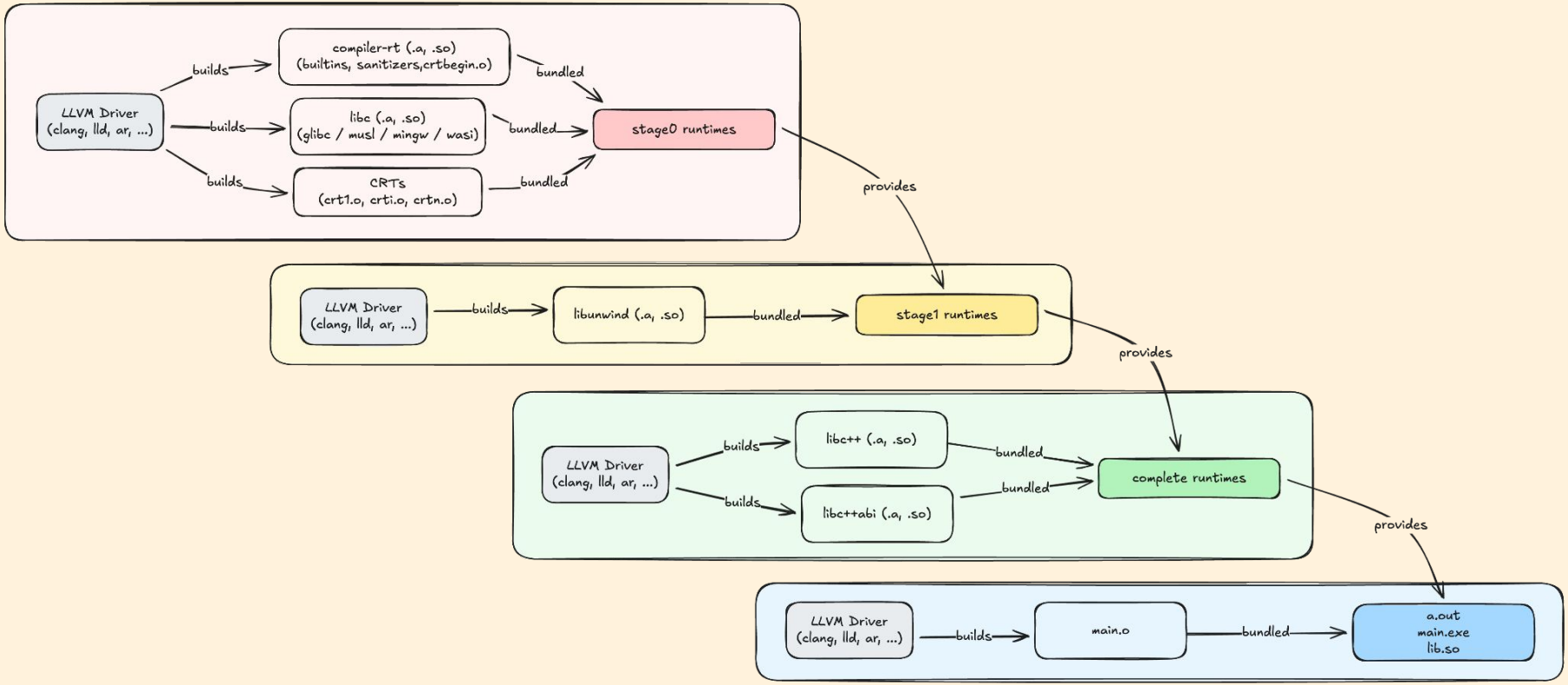
Wiring

Prebuilt toolchain
+
Runtime sources
+
BUILD.bazel files
+
Bazel `toolchain()` definitions

https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

Bazel transitioned toolchains



https://github.com/cerisier/toolchains_llvm_bootstrapped

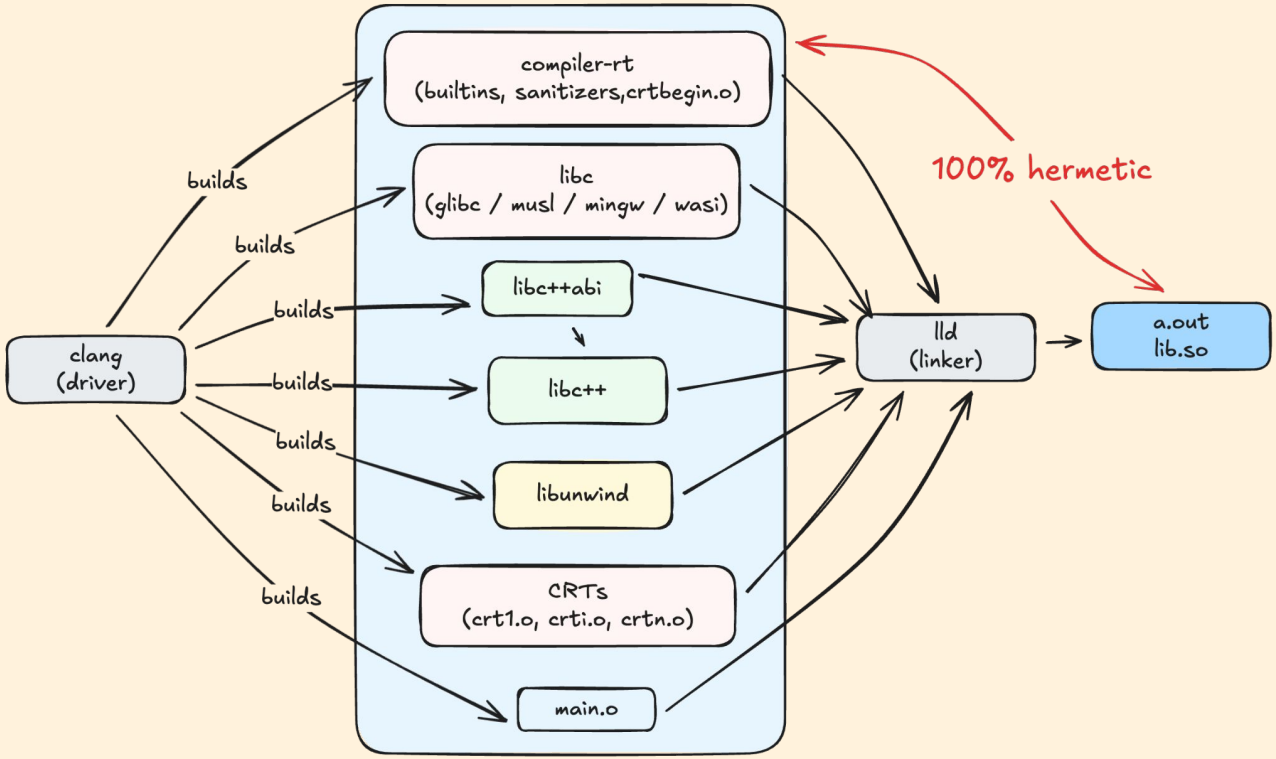
The hermetic approach

Hermetic link graph

–sysroot=/dev/null

The hermetic approach

Hermetic link graph



https://github.com/cerisier/toolchains_llvm_bootstrapped

The hermetic approach

Key consequence

any host ➡️ any target

The hermetic approach

Key consequence

Supported platforms

✅ Currently supports cross-compilation between all combinations of the following platforms:

To ↓ / From →	macOS aarch64	Linux aarch64	Linux x86_64	Windows aarch64	Windows x86_64
aarch64-apple-darwin	✅	✅	✅	✅	✅
x86_64-apple-darwin	✅	✅	✅	✅	✅
aarch64-linux-gnu [1]	✅	✅	✅	✅	✅
x86_64-linux-gnu [1]	✅	✅	✅	✅	✅
aarch64-linux-musl	✅	✅	✅	✅	✅
x86_64-linux-musl	✅	✅	✅	✅	✅
aarch64-windows-gnu [2]	✅	✅	✅	✅	✅
x86_64-windows-gnu [2]	✅	✅	✅	✅	✅
wasm32-unknown-unknown	✅	✅	✅	✅	✅
wasm64-unknown-unknown	✅	✅	✅	✅	✅

\+ **many more** in progress

https://github.com/cerisier/toolchains_llvm_bootstrapped

The Hard Parts

Bazelifying everything

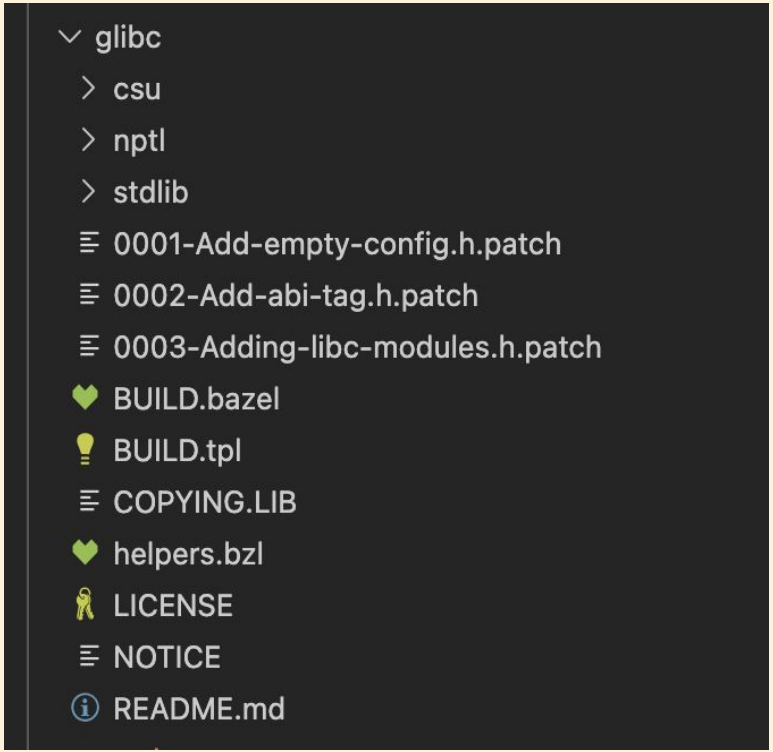
But we need to write the **BUILD.bazel** files

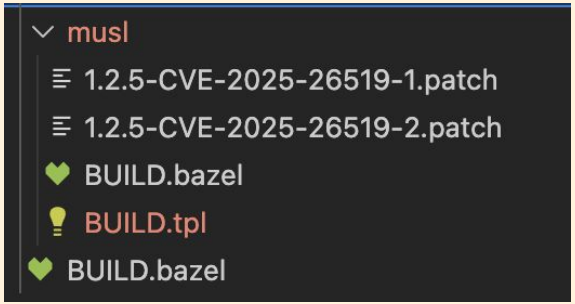
> Either adapt existing runtimes or provide new ones (wasi, ape, etc.)

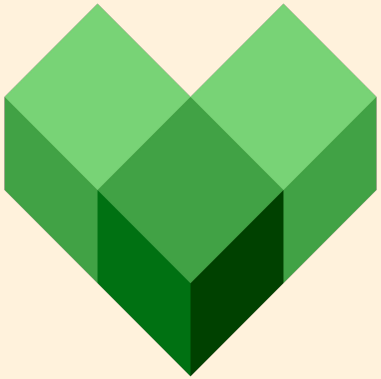
https://github.com/cerisier/toolchains_llvm_bootstrapped

The Hard Parts

Bazelifying everything







https://github.com/cerisier/toolchains_llvm_bootstrapped

The Hard Parts

The glibc case

https://github.com/cerisier/toolchains_llvm_bootstrapped

The glibc case

The issue

01 Complex makefile and custom scripts

02 Requires a gcc cross compiler

03 Clang support since 2.43 (Jan 2026)

04 Support needed for **many** versions

https://github.com/cerisier/toolchains_llvm_bootstrapped

The glibc case

The solution

Generate a **stub library** and link against it

aka interface library (.ifso)

https://github.com/cerisier/toolchains_llvm_bootstrapped

The glibc case

Thank you Zig

c.s

```
.text
.balign 8
.globl __strtoull_l
.type __strtoull_l, %function;
.symver __strtoull_l, __strtoull_l@@GLIBC_2.2.5
__strtoull_l: .quad 0
.balign 8
.globl __wcsftime_l
.type __wcsftime_l, %function;
.symver __wcsftime_l, __wcsftime_l@@GLIBC_2.3
__wcsftime_l: .quad 0
.balign 8
.globl __xmknodat
.type __xmknodat, %function;
.symver __xmknodat, __xmknodat@@GLIBC_2.4
__xmknodat: .quad 0
```

```
$ clang -nostdlib -shared c.s -o libc.so
$ nm -D libc.so
000000000001b9f0 T __strtoull_l@@GLIBC_2.2.5
000000000001c780 T wcsftime_l@@GLIBC_2.3
000000000001ba00 T __xmknodat@@GLIBC_2.4
```

```
$ clang -L. -lc main.c
$ readelf -d a.out
Dynamic section at offset 0xaeac0 contains 28 entries:
  Tag                Type           Name/Value
  0x0000000000000001 (NEEDED)       Shared library: [libc.so.6]
  0x000000000000001e (FLAGS)        BIND_NOW
  0x00000006fffffffb (FLAGS_1)      NOW PIE
```

https://github.com/cerisier/toolchains_llvm_bootstrapped

The glibc case

Thank you Zig

01 Abilists to the rescue

02 Symbols index per **lib**, **version** and **arch**

03 Stub generator

https://github.com/cerisier/toolchains_llvm_bootstrapped


$ demo

Sanitizers

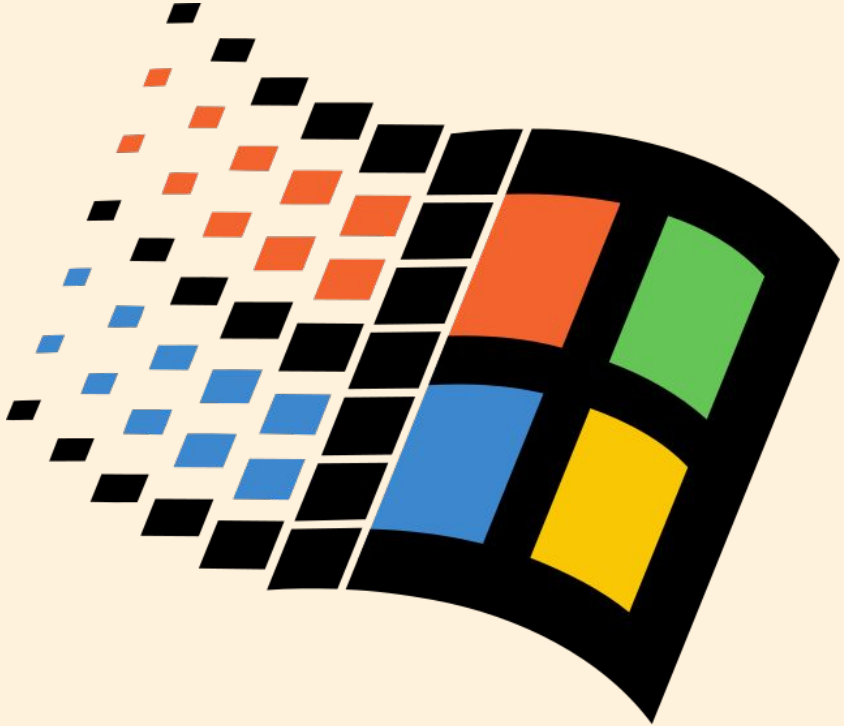
01 Built from LLVM sources

02 Support fully static or dynamic linking

03 Composable with the rest of the build.
Build a UBSAN’ed LLVM?

https://github.com/cerisier/toolchains_llvm_bootstrapped

Windows



https://github.com/cerisier/toolchains_llvm_bootstrapped

Windows

01 Build **mingw** CRTs

02 Build **mingwex** and other extra C libs

03 Build import libs from DLL .defs

04 Assemble umbrella import libs from MRI recipes

https://github.com/cerisier/toolchains_llvm_bootstrapped


$ demo

The perks

Bazel enabled

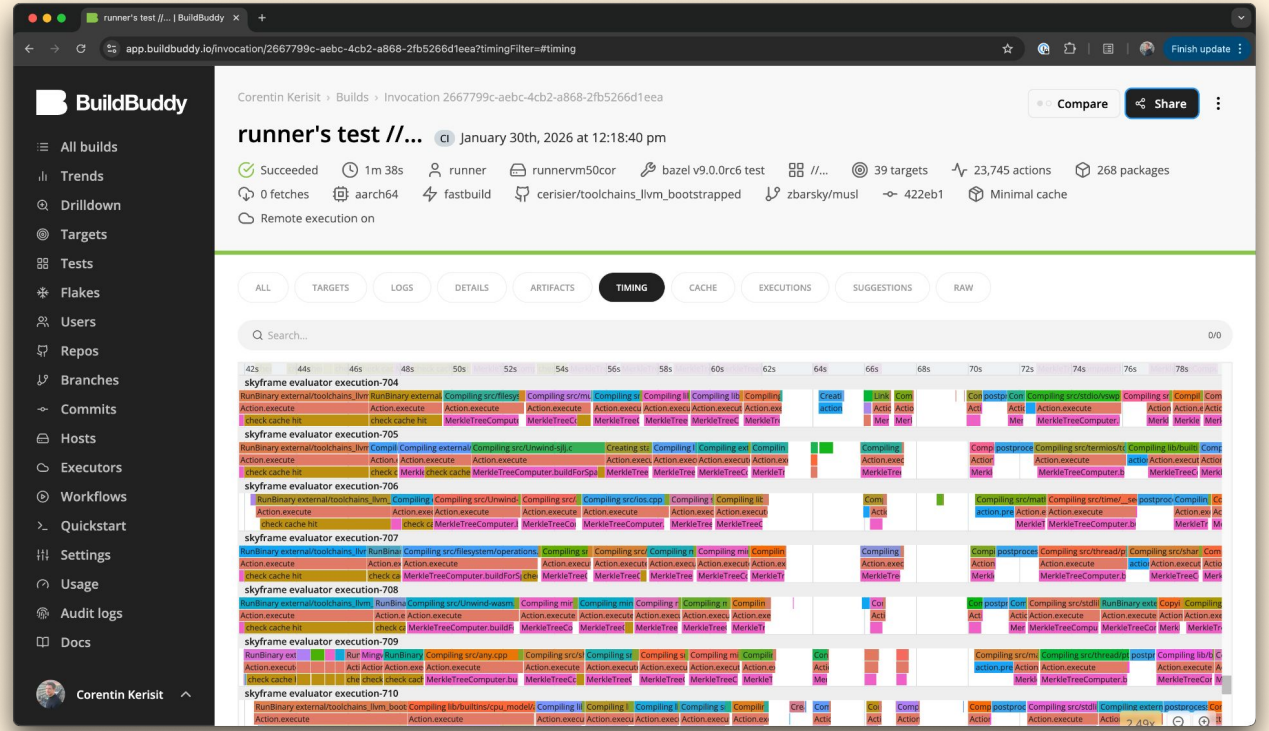
01 Zero-setup remote builds

02 Zero-setup C cross compilation and cross link to other languages (Rust, Go, …)

https://github.com/cerisier/toolchains_llvm_bootstrapped

The perks

Thank you BuildBuddy



https://github.com/cerisier/toolchains_llvm_bootstrapped

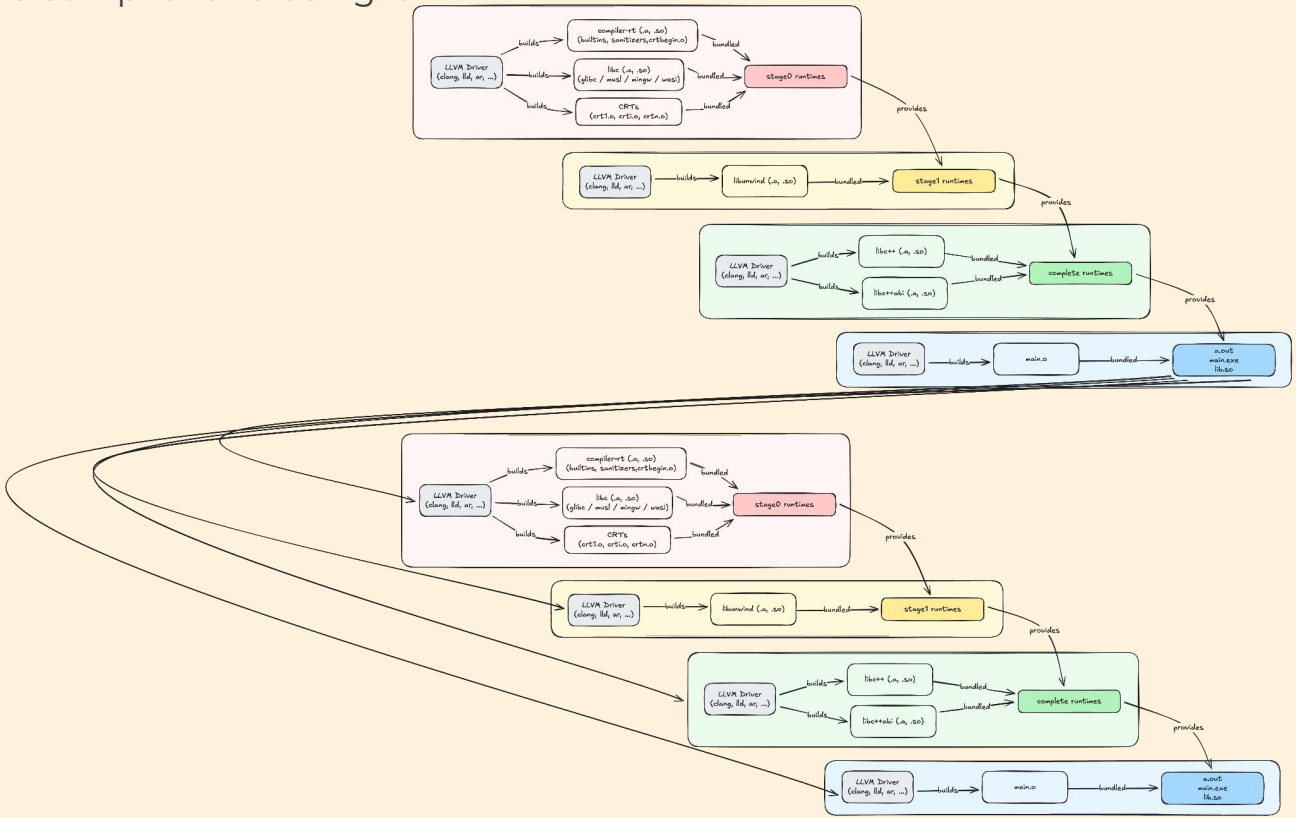

$ demo

Bootstrapping

https://github.com/cerisier/toolchains_llvm_bootstrapped

Bootstrapping

Compiling the compiler and using it



https://github.com/cerisier/toolchains_llvm_bootstrapped

Bootstrapping

Compiling the compiler and using it

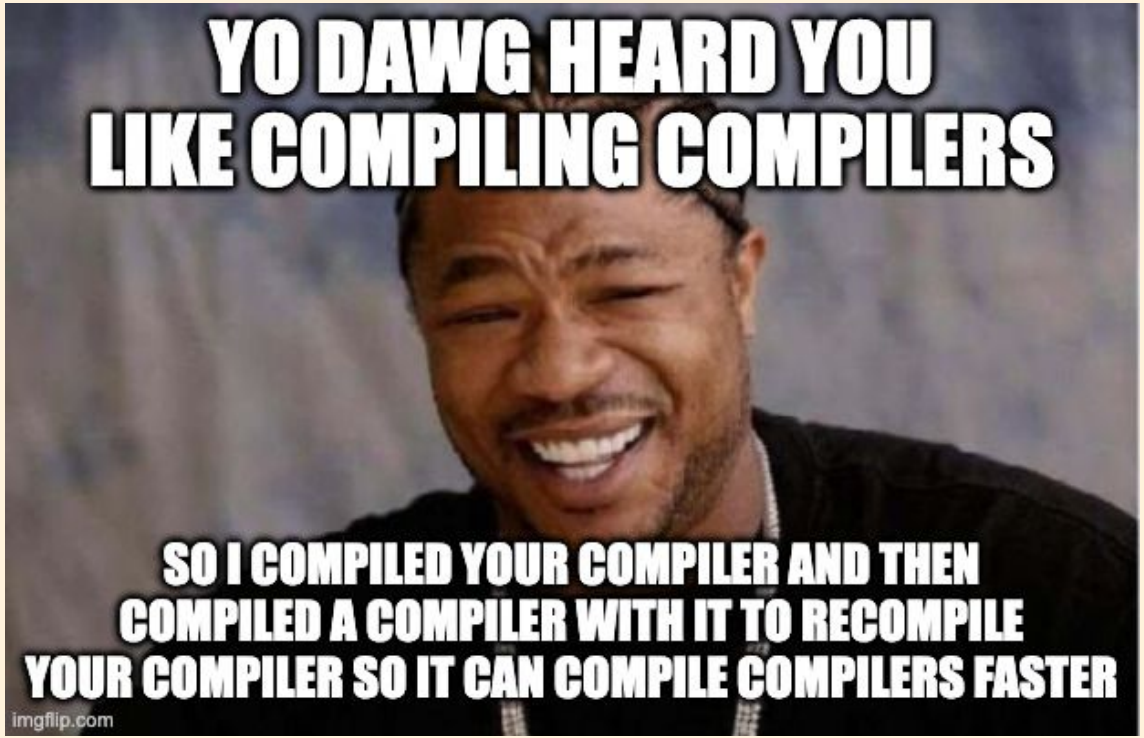
In a single build.

https://github.com/cerisier/toolchains_llvm_bootstrapped


$ demo

What’s next?

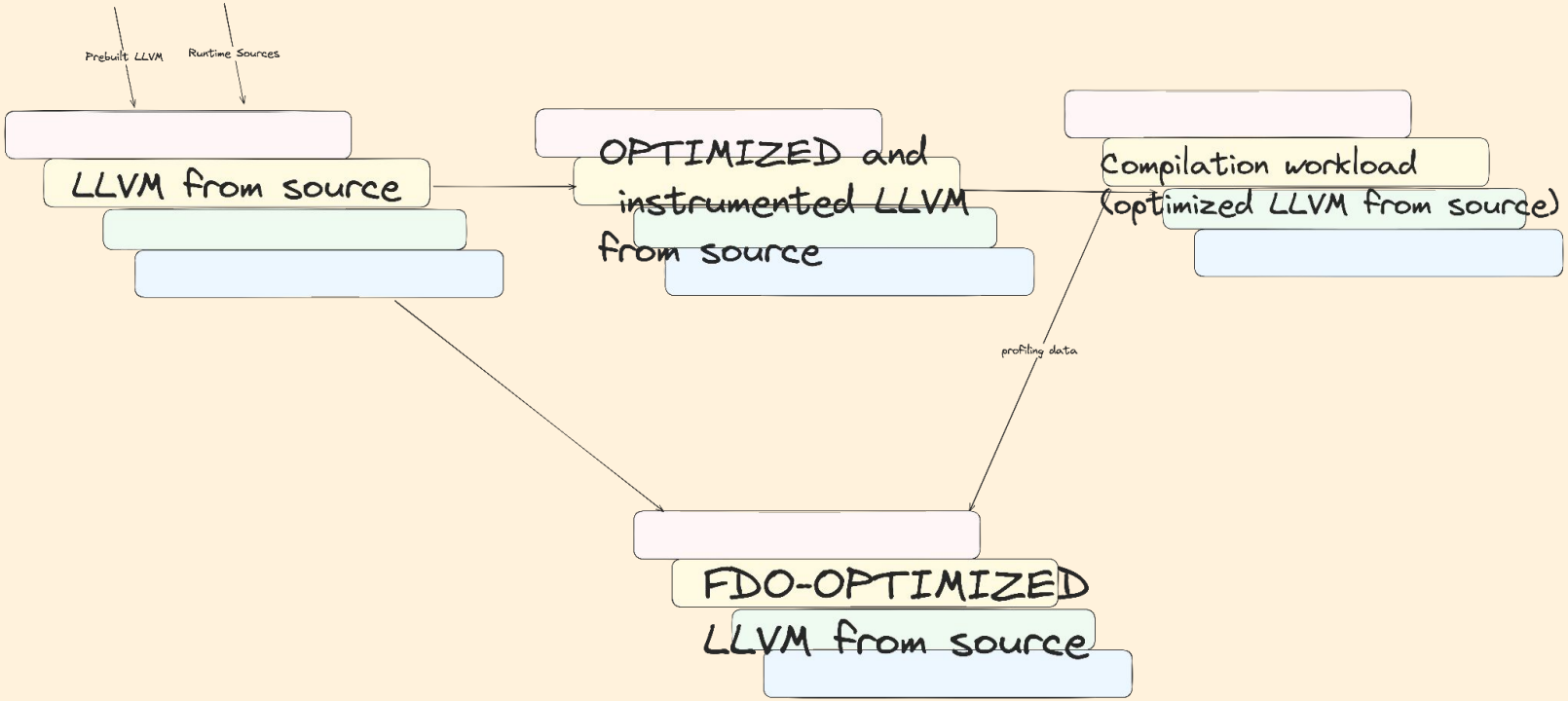
FDO



https://github.com/cerisier/toolchains_llvm_bootstrapped

What's next?

FDO



https://github.com/cerisier/toolchains_llvm_bootstrapped

What's next?

Exhaustive targets and runtimes support

- armbe
- thumb
- spirv
- risc

Your favorite **target**?
Send us a PR!

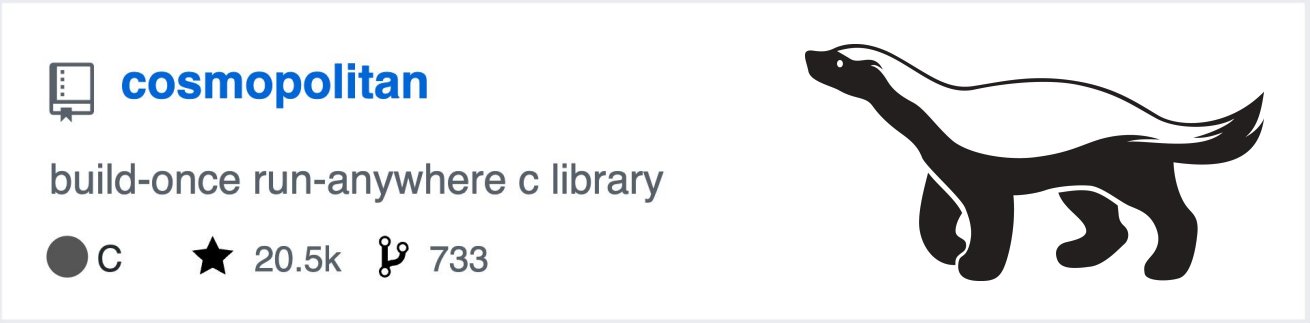
- wasi
- cygwin
- ape
- ???

Your favorite **runtime**?
Send us a PR!

https://github.com/cerisier/toolchains_llvm_bootstrapped

What’s next?

Cosmopolitan libc/APE – build once, run everywhere



https://github.com/cerisier/toolchains_llvm_bootstrapped

What's next?

Make this useful outside of Bazel

```
bazel build //:crossenv --platforms=//:linux_arm64_musl
make CC="./bazel-bin/crossenv/clang-wrapper"
```

https://github.com/cerisier/toolchains_llvm_bootstrapped

What's next?

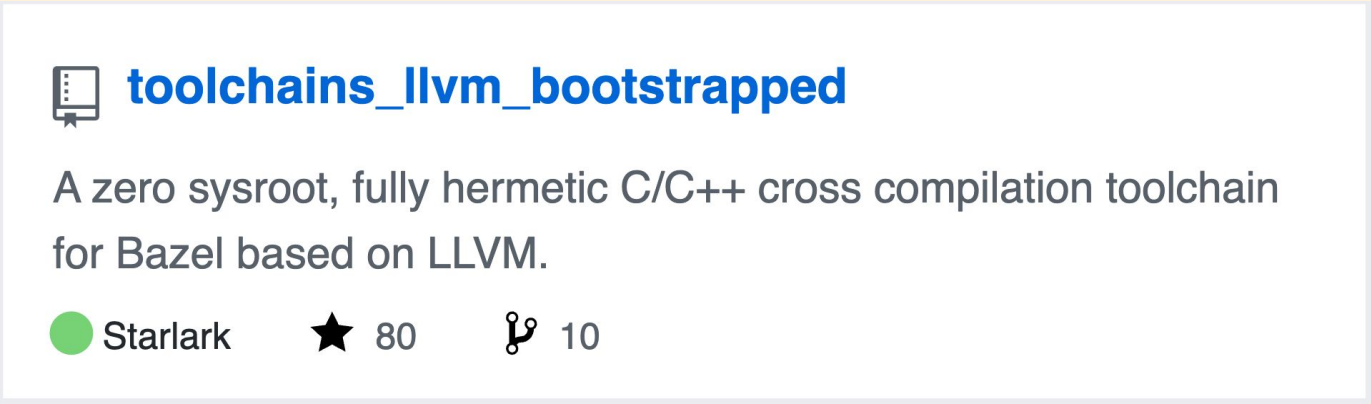
Make this useful outside of Bazel

```
# bazel-crossenv wraps bazel invocation
make CC="./bazel-crossenv --target armeb-linux-musleabi"
```

https://github.com/cerisier/toolchains_llvm_bootstrapped

Thank you

Feedback and contributions welcomed



https://github.com/cerisier/toolchains_llvm_bootstrapped