A test suite for packagers

https://github.com/opencollab/llvm-toolchain-integration-test-suite/
2021-09-15

Who are we?

Serge

- LLVM developer
- Member of LLVM packaging team at RedHat

Sylvestre

- Main maintainer of the llvm toolchain for +10 years on Debian & Ubuntu
- Maintainer of apt.llvm.org
- Paid job: director of engineering at Mozilla

How it started

- Debian and Ubuntu have an hugly 2000 lines of code single file
  - Verifies integration, common issues, non regression, etc
- Lot of valuable checks but too *deb specific*
- Fedora also has its own set of checks for each LLVM sub-package
- Serge and Sylvestre were chatting during one of the French covid lockdown and a *what if* moment happened

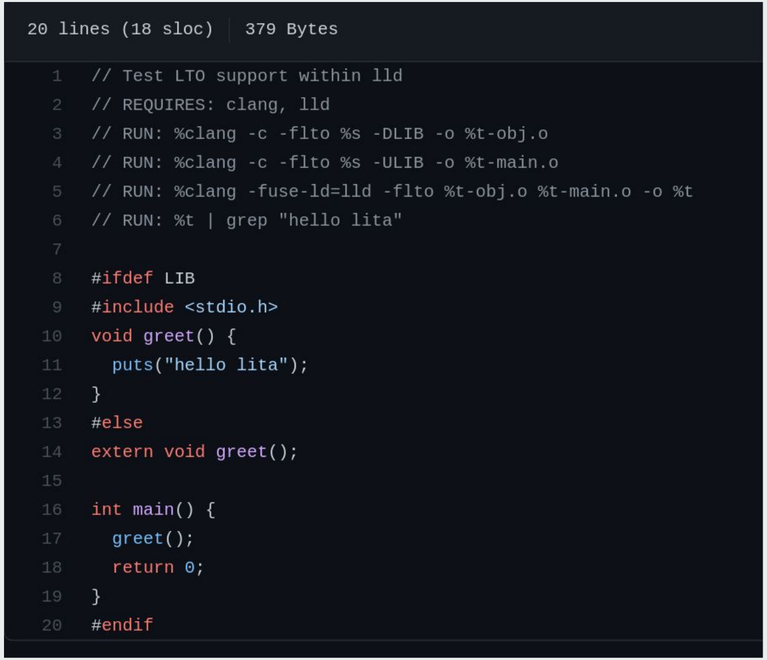
Goals of this project

- Make sure that the various tools provided by the llvm toolchain work well together
  - Trivial example: *clang -fuse-ld=lld -flto*
- Work on the packages, without source or build directories available
- Consistency over package practices and user behavior
- Make sure we don’t regress
- Share the effort across packagers
- Provide a behavior close to the *make distcheck* rule from automake

How it works

```
> cmake ..
-- Checking for lit
-- Checking for clang++
-- Checking for clang
-- Checking for clang-tidy
[...]
-- Checking for scan-build-py
-- Disabling tests related to scan-build-py
-- Configuring done
-- Generating done
> make check
-- Testing: 40 tests, 8 workers --
PASS: LLVM regression suite :: basic_openmp.c (1 of 40)
PASS: LLVM regression suite :: atomic_fetch.c (2 of 40)
[...]
PASS: LLVM regression suite :: plugins.cpp (39 of 40)
PASS: LLVM regression suite :: llvm_config_libs.cpp (40 of 40)
```

How it works - 2



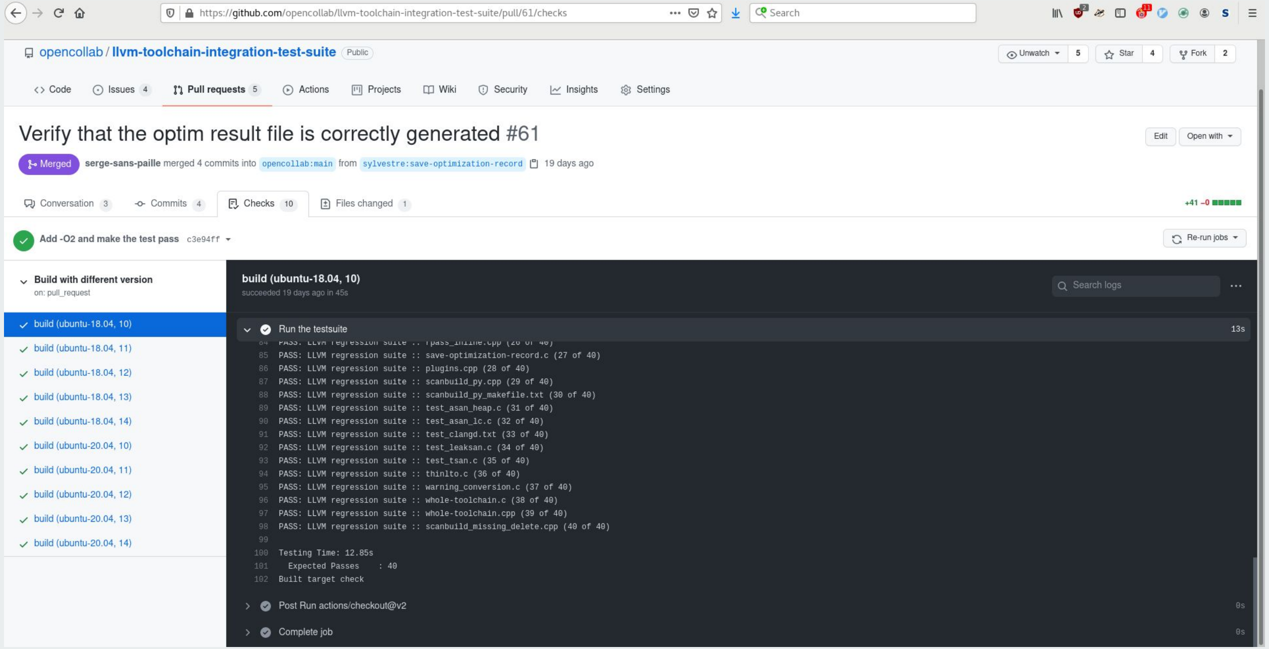
20 lines (18 sloc) | 379 Bytes

```c
// Test LTO support within lld
// REQUIRES: clang, lld
// RUN: %clang -c -flto %s -DLIB -o %t-obj.o
// RUN: %clang -c -flto %s -ULIB -o %t-main.o
// RUN: %clang -fuse-ld=lld -flto %t-obj.o %t-main.o -o %t
// RUN: %t | grep "hello lita"

#ifdef LIB
#include <stdio.h>
void greet() {
  puts("hello lita");
}
#else
extern void greet();

int main() {
  greet();
  return 0;
}
#endif
```

How it works - 3



https://github.com/opencollab/llvm-toolchain-integration-test-suite/pull/61/checks

opencollab / llvm-toolchain-integration-test-suite Public

Verify that the optim result file is correctly generated #61

Merged serge-sans-paille merged 4 commits into `opencollab:main` from `sylvestre:save-optimization-record` 19 days ago

Add -O2 and make the test pass c3e94ff

Build with different version
on: pull_request

- ✓ build (ubuntu-18.04, 10)
- ✓ build (ubuntu-18.04, 11)
- ✓ build (ubuntu-18.04, 12)
- ✓ build (ubuntu-18.04, 13)
- ✓ build (ubuntu-18.04, 14)
- ✓ build (ubuntu-20.04, 10)
- ✓ build (ubuntu-20.04, 11)
- ✓ build (ubuntu-20.04, 12)
- ✓ build (ubuntu-20.04, 13)
- ✓ build (ubuntu-20.04, 14)

build (ubuntu-18.04, 10)
succeeded 19 days ago in 45s

Run the testsuite 13s

```
PASS: LLVM regression suite :: rpass_inline.cpp (26 of 40)
PASS: LLVM regression suite :: save-optimization-record.c (27 of 40)
PASS: LLVM regression suite :: plugins.cpp (28 of 40)
PASS: LLVM regression suite :: scanbuild_py.cpp (29 of 40)
PASS: LLVM regression suite :: scanbuild_py_makefile.txt (30 of 40)
PASS: LLVM regression suite :: test_asan_heap.c (31 of 40)
PASS: LLVM regression suite :: test_asan_lc.c (32 of 40)
PASS: LLVM regression suite :: test_clangd.txt (33 of 40)
PASS: LLVM regression suite :: test_leaksan.c (34 of 40)
PASS: LLVM regression suite :: test_tsan.c (35 of 40)
PASS: LLVM regression suite :: thinlto.c (36 of 40)
PASS: LLVM regression suite :: warning_conversion.c (37 of 40)
PASS: LLVM regression suite :: whole-toolchain.c (38 of 40)
PASS: LLVM regression suite :: whole-toolchain.cpp (39 of 40)
PASS: LLVM regression suite :: scanbuild_missing_delete.cpp (40 of 40)

Testing Time: 12.85s
  Expected Passes    : 40
Built target check
```

Post Run actions/checkout@v2 0s

Complete job 0s

What do we support/test?

- llvm tools
- clang
- some clang tools
- polly
- lldb
- lld
- libomp
- libunwind
- compiler-rt
- LLVM plugins

What do we not test?

- distribution-specific behavior
- language-specific details
- behavior specific to a given LLVM version

Successes

- Show some recent regressions (LLVMgold)
- Detected an installation issue with scan-build-py, that we fixed upstream

Current status

- Used in Debian & Ubuntu
- Used by apt.llvm.org
- Used by Fedora and RHEL

Next steps

- Make it official and integrated (ninja check-install ?)
- Get more distros
- Test LLVM runtimes builds
- Add support for MLIR, libc, etc
- Debian/Ubuntu: make progress on decreasing the g++ dependencies