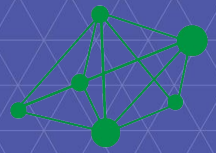
HKG18-118: OpenDataPlane Testing in Travis and Shippable

Dmitry Eremin-Solenikov, Cavium
Maxim Uvarov, Linaro



Linaro connect
Hong Kong 2018

What is ODP (OpenDataPlane)

- The ODP project is an open-source, cross-platform set of APIs for the networking data plane
- Project hosted on github:

  https://github.com/Linaro/odp
- ODP is written using C99
  - We support x86, Arm, MIPS64 and PowerPC platforms
- We use Travis CI and Shippable as our CI systems

Linaro connect
Hong Kong 2018

What is Travis CI

- Travis CI is free continuous integration platform, tightly integrated with GitHub hosting system
- It features rich set of features, which we exploited for our development
  - Scriptable and configurable task descriptions
  - Up to 5 builders running in parallel
  - Networking access, root access, etc
- Easy to start: Just put simple `.travis.yml` file into your project root

Github and Travis integration

- Travis runs on each git push to the repo once enabled (does not matter which repo is it: main or private fork)
- Validation for each pull request and PR updates
- Travis cron runs for stable branch

Applied Travis CI

- Build matrix + jobs.include: two ways to specify build tasks
  - Matrix compilers vs environment
  - Additional jobs added one by one
- Cross-compilation is possible, but required additional efforts (Ubuntu provides cross-toolchains for supported architectures).
- Each additional test is a separate job

Build Matrix in YAML

```
compiler:
    - gcc
    - clang-3.8
env:
  matrix:
    - CONF="--disable-abi-compat"
    - DPDK_VERS="17.11" CONF=""
jobs:
    include:
        - stage: test
          compiler: aarch64-linux-gnu-gcc
          env: TEST="aarch64-linux-gnu" CROSS_ARCH="arm64"
          script:
            - ./bootstrap
            - ./configure --prefix=$HOME/odp-install $CROSS --enable-debug
              --disable-test-cpp --enable-test-vald --enable-test-helper --enable-test-perf
```

Build Matrix (part)

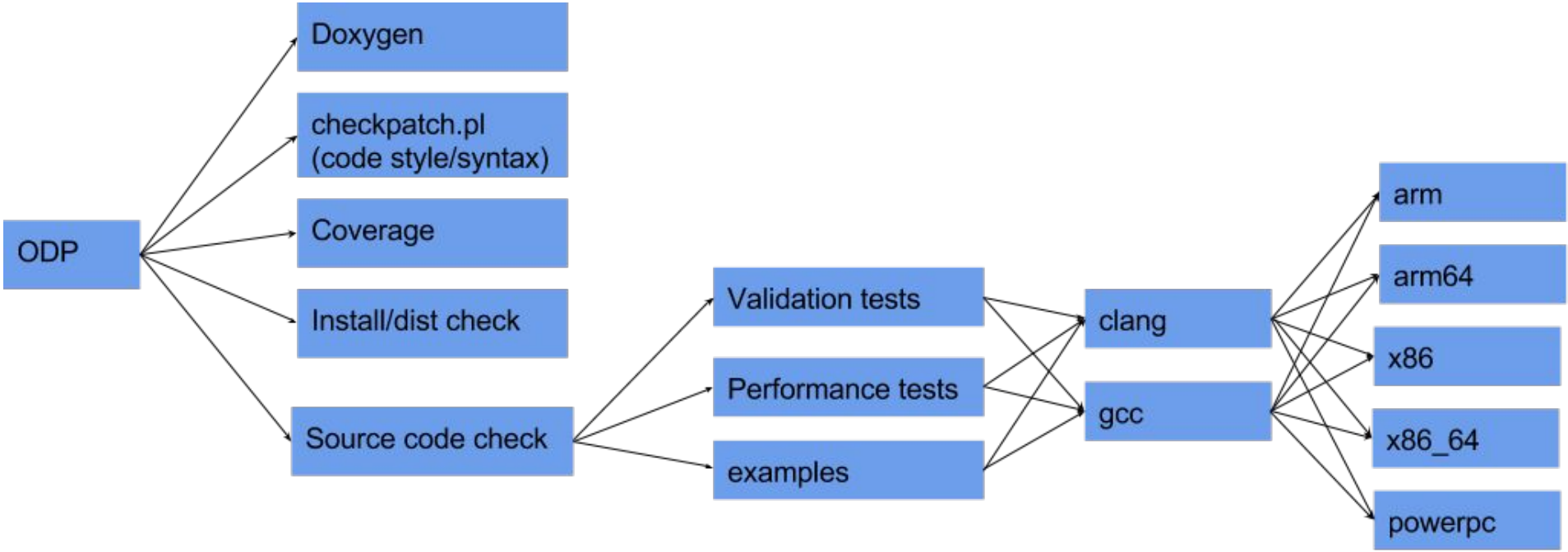
Build only — 6 min 56 sec

# 2253.1	Compiler: gcc C	TEST=doxygen	2 min 12 sec
# 2253.2	Compiler: gcc C	CONF=""	2 min 40 sec
# 2253.3	Compiler: clang-3.8 C	CONF=""	2 min 32 sec
# 2253.4	Compiler: gcc C	CROSS_ARCH="i386"	4 min 2 sec
# 2253.5	Compiler: gcc C	CROSS_ARCH="arm64"	3 min 49 sec

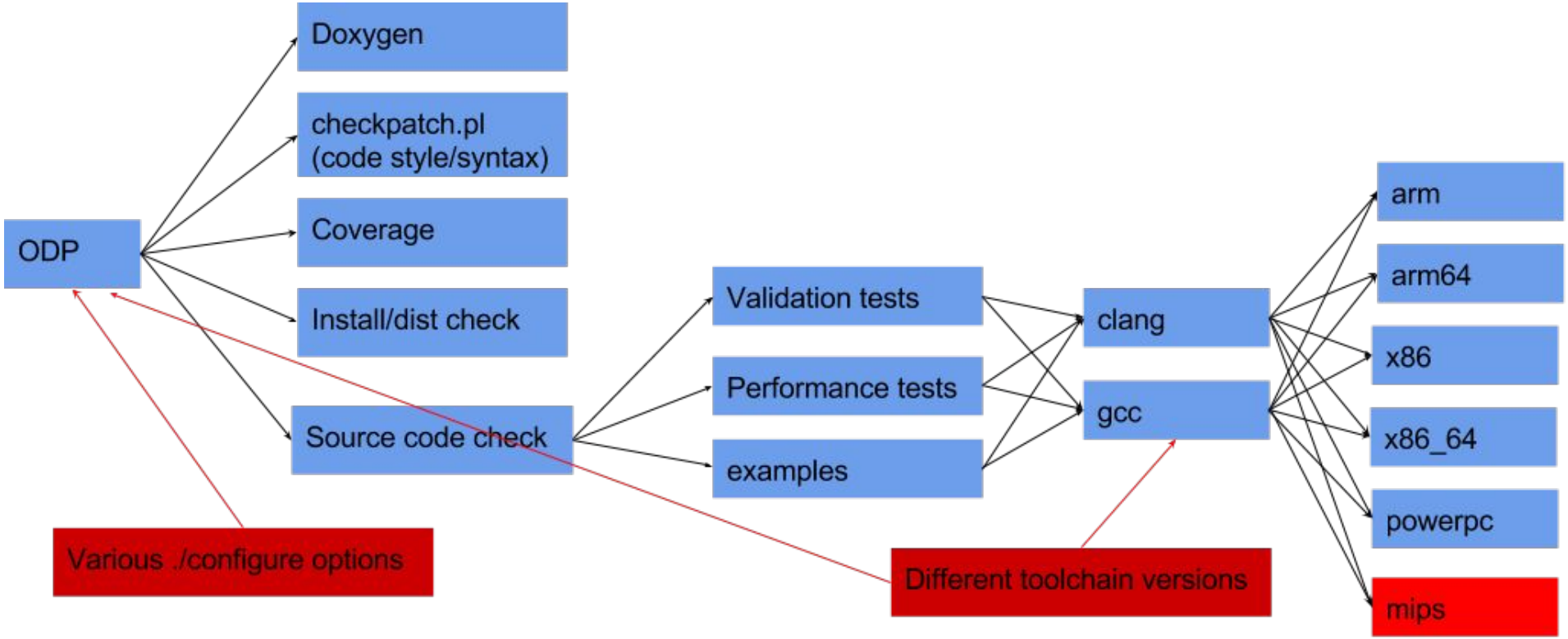
Test — 1 hr 25 min 1 sec

# 2253.6	Compiler: gcc C	CONF=""	37 min 56 sec
# 2253.7	Compiler: clang-3.8 C	CONF=""	37 min 22 sec
# 2253.8	Compiler: gcc C	CONF="--disable-abi-compat"	11 min 40 sec
# 2253.9	Compiler: clang-3.8 C	CONF="--disable-abi-compat"	11 min 52 sec
# 2253.10	Compiler: gcc C	CONF="--enable-deprecated"	11 min 47 sec
# 2253.11	Compiler: clang-3.8 C	CONF="--enable-deprecated"	11 min 40 sec
# 2253.12	Compiler: gcc C	CONF="--enable-dpdk-zero-copy"	11 min 43 sec
# 2253.13	Compiler: clang-3.8 C	CONF="--enable-dpdk-zero-copy"	11 min 59 sec
# 2253.14	Compiler: gcc C	CONF="--disable-static-applications"	11 min 27 sec
# 2253.15	Compiler: clang-3.8 C	CONF="--disable-static-applications"	13 min 1 sec
# 2253.16	Compiler: gcc C	CONF="--disable-host-optimization"	10 min 49 sec
# 2253.17	Compiler: clang-3.8 C	CONF="--disable-host-optimization"	10 min 53 sec

ODP testing looks like:



Did we forget something? YES!



Gory Details

- We started with just 2 compilers x 4 build variants matrix
- Then we had to add more build variants...
- … and cross compilation...
- … and few special tasks…
- … and it went on and on…

- Till we had single build taking up to 4 ¼ hours

Build caching

- Total build time is 8:27 h
- No need to compile dependencies on each run!

```
cache:
        ccache: true
        pip: true
        directories:
                - $HOME/cunit-install
                - $HOME/doxygen-install
                - dpdk
                - netmap
```

Code coverage

```
788  1  static inline int move_data_to_tail(odp_packet_hdr_t *pkt_hdr, int segs)
789     {
790             int dst_seg, src_seg;
791             uint32_t len, free_len;
792  1          uint32_t moved = 0;
793
794  6          for (dst_seg = segs - 1; dst_seg >= 0; dst_seg--) {
795  6                  len    = pack_seg_tail(pkt_hdr, dst_seg);
796  6                  moved += len;
797
798  6                  if (len == BASE_LEN)
799                             continue;
800
801  6                  free_len = BASE_LEN - len;
802
803  6                  for (src_seg = dst_seg - 1; src_seg >= 0; src_seg--) {
804  5                          len = fill_seg_tail(pkt_hdr, dst_seg, src_seg,
805                                                 free_len);
806  5                          moved += len;
807
808  5                          if (len == free_len) {
809                                     /* dst seg is full */
810  5                                  break;
811                             }
812
813                             /* src seg is empty */
814                             free_len -= len;
815                     }
816
817  6                  if (moved == pkt_hdr->frame_len)
818  1                          break;
819             }
820
821             /* first segment which have data */
822  1          return dst_seg;
823     }
```

Linaro connect
Hong Kong 2018

Shippable

- Travis executes all build stages on x86-64 hosts
- We would like to test our software on other architectures
- Shippable allows one to plug your own AArch64 nodes and use them to build your software

Shippable through the magnifying glass

- x86-64 or AArch64 architecture
- Docker + YAML config
- Everything is run as root
- Same build matrix as under Travis
- No extra “build stages”
- Visual representation for tests results and code coverage
- Just one build node by default, pay for more

ODP and Shippable

- For now we have simple 2x2 build matrix (Clang or GCC vs ABI-compat/non-ABI-compat)
- Just one build node, so tests are not parallelized
- Performance tests are disabled to speed up builds

Overall Status	Failed	Language	c	Commit SHA	afa6de80d5
Branch	PR #5...	Tests Passed	6091	Commit Msg	validation: ipsec: inbound TFC dummy packets check Signed-off-by: Dmitry Eremin-Solenikov <dmitry.ereminsolenikov@linaro.org> Reviewed-by: Bill Fischofer <bill.fischofer@linaro.org> Signed-off-by: Maxim Uvarov <maxim.uvarov@linaro.org>
Triggered By	muvarov	Tests Failed	1		
Started At	Last Thursday at 11:00 PM	Tests Errors	0		
		Tests Skipped	0		
Duration	58 minutes	Branch Coverage	Not Available		
Total Q Dur	54 minutes	Seq Coverage	Not Available		

#	Status	Lang Version	Started At	Duration	Environment	Tests	Coverage
1	Success	none	Last Thursday at 11:00 PM	23 minutes	CONF="--disable-test-perf --d...	P:2732 F:0 E:0 S:0	No Coverage Dat
2	Failed	none	Last Thursday at 11:23 PM	7 minutes	CONF="--disable-test-perf --d...	P:627 F:1 E:0 S:0	No Coverage Dat
3	Success	none	Last Thursday at 11:31 PM	23 minutes	CONF="--disable-abi-compat...	P:2732 F:0 E:0 S:0	No Coverage Dat
4	Failed	none	Last Thursday at 11:54 PM	4 minutes	CONF="--disable-abi-compat...	No Test Data	No Coverage Dat

Linaro connect
Hong Kong 2018

Future steps

- Improve validation time
- Although we are using both Travis and Shippable, it is not enough for our tasks

Implement a set of tests to be executed on real hardware:

- Use hardware interfaces
- Test performance
- Test implementations compatibility

Linaro connect
Hong Kong 2018

Summary

- No more ‘git bissect’. We validate each Pull Request. We know that code builds and runs!
- No last minute bug fixes before release.
- Test infrastructure can be reused by forked projects.
- Maintiners review patches _after_ tests.
- Tests are part of the project. Anyone can contribute!

Thank You

#HKG18

ODP github page https://github.com/Linaro/odp

Travis CI configuration: https://github.com/Linaro/odp/blob/master/.travis.yml

Travis: https://travis-ci.org/

For further information: www.opendataplane.org

Linaro connect
Hong Kong 2018

What is ODP Continued

- Project hosted on github:

  https://github.com/Linaro/odp
- Provides dynamic and static libraries
- Uses autotools (autoconf, automake).
- Uses Doxygen to generate API documentation