University of New Hampshire
InterOperability
Laboratory

The University of New Hampshire InterOperability Laboratory
(UNH-IOL)

DPDK Lab Summary Jan 2020

Jeremy Plsek

www.iol.unh.edu

Systems

Hardware from vendors:

- Broadcom
  - brcm_57414 25000 Mbps
- Intel (public, 11/2018)
  - XL710-QDA2 40 Gbps
  - 82599ES 10 Gbps
- Mellanox (public, 11/2018)
  - ConnectX-4 Lx 40 Gbps
  - ConnectX-4 Lx 25 Gbps
  - ConnectX-5 100 Gbps
- NXP (public, 12/2019)
  - LS2088A 10000 Mbps

Virtual Machines:

- Arch Linux (1/2020)
- CentOS 8 (10/2019)
- Fedora 31 (12/2019)
- FreeBSD 11.2 (8/2019)
- openSuse Leap 15.1 (1/2020)
- Ubuntu 18.04 x2 (7/2019)
- Windows Server 2019

Dates represent when the systems started reporting to patchworks (i.e. public)

Testing Overview

System	Performance Testing (dts)	DPDK+OVS Performance Testing (ovs_perf)	Compile Testing (make)	Compile Testing (meson)	Unit Testing	DPDK+OVS Compile Testing	DPDK+SPDK Compile Testing
Broadcom	bm						
Intel	bm	bm				bm	
Mellanox	bm	bm				bm	
NXP (arm)	bm						
Arch Linux			c	c		c	
CentOS 8			c	c		c	c
Fedora 31			c	c		c	c
FreeBSD 11.2			vm	vm			vm
openSuse Leap 15.1			c	c		c	
Ubuntu 18.04			c	c	vm	c	c
Windows Server 2019				vm			

Hardware systems are meant for running performance testing, while the virtual machines are meant to run non-hardware nic dependent testing.

bm: bare metal; c: container; vm: virtual machine

Involved Downstream Projects

- OVS (dpdk-latest branch)
- SPDK (v20.01x branch)

Test downstream projects when DPDK patches are submitted and DPDK subtrees are updated.

How Testing is Started

Runs with Jenkins.

Polls DPDK Patchwork REST API

→ Add patch series to our DB

→ Apply patch series to dpdk master (or subtree)

→ If build fails: Send email to patchwork mailing list and record to our DB

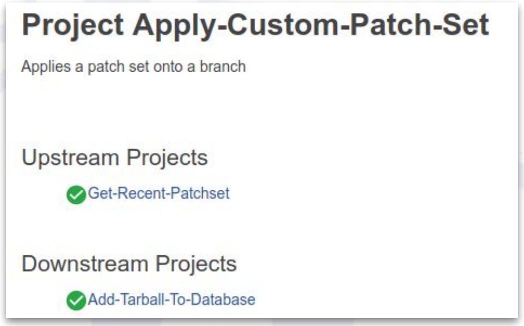
→ If build succeeds: Create and upload a tarball to the DB

→ Run through all testing

Polls DPDK Git repository (including all subtrees)

→ Create and upload a tarball to the DB

→ Run through all testing



Project Apply-Custom-Patch-Set

Applies a patch set onto a branch

Upstream Projects

✓ Get-Recent-Patchset

Downstream Projects

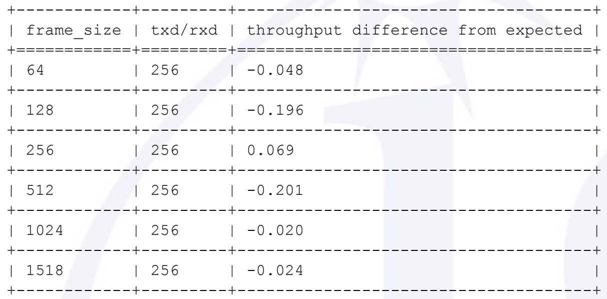
✓ Add-Tarball-To-Database

How Testing is Performed (Performance)

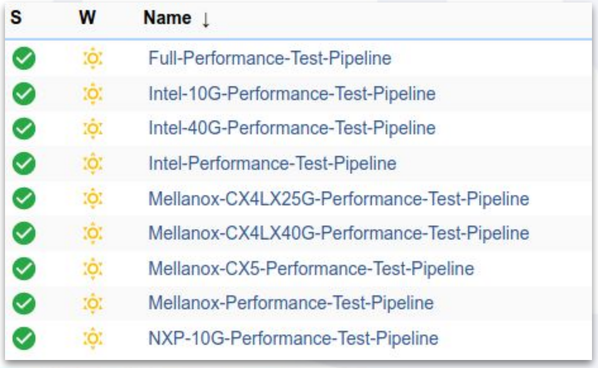
For each applicable environment:

→ Get tarball from DB

→ Run DTS nic_single_core_perf



```
+------------+---------+------------------------------------+
| frame_size | txd/rxd | throughput difference from expected |
+============+=========+====================================+
| 64         | 256     | -0.048                             |
+------------+---------+------------------------------------+
| 128        | 256     | -0.196                             |
+------------+---------+------------------------------------+
| 256        | 256     | 0.069                              |
+------------+---------+------------------------------------+
| 512        | 256     | -0.201                             |
+------------+---------+------------------------------------+
| 1024       | 256     | -0.020                             |
+------------+---------+------------------------------------+
| 1518       | 256     | -0.024                             |
+------------+---------+------------------------------------+
```



S	W	Name ↓
✓	☼	Full-Performance-Test-Pipeline
✓	☼	Intel-10G-Performance-Test-Pipeline
✓	☼	Intel-40G-Performance-Test-Pipeline
✓	☼	Intel-Performance-Test-Pipeline
✓	☼	Mellanox-CX4LX25G-Performance-Test-Pipeline
✓	☼	Mellanox-CX4LX40G-Performance-Test-Pipeline
✓	☼	Mellanox-CX5-Performance-Test-Pipeline
✓	☼	Mellanox-Performance-Test-Pipeline
✓	☼	NXP-10G-Performance-Test-Pipeline

How Testing is Performed (Smoke)

For each applicable environment:

→ Get tarball from DB

→ Run applicable tests

→ Compile testing (make)

→ Compile testing (meson)

→ Unit testing

→ DPDK+OVS compile testing

→ DPDK+SPDK compile testing

```
+------------------+-------------------+------------------+----------------+--------------------+--------------+
|   Environment    | dpdk_compile_spdk | dpdk_compile_ovs | dpdk_unit_test | dpdk_meson_compile | dpdk_compile |
+==================+===================+==================+================+====================+==============+
| FreeBSD 11.2     | PASS              | SKIPPED          | SKIPPED        | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
| Ubuntu 18.04     | PASS              | PASS             | PASS           | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
| openSUSE Leap 15 | SKIPPED           | PASS             | SKIPPED        | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
| Arch Linux       | SKIPPED           | PASS             | SKIPPED        | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
| Fedora 31        | SKIPPED           | PASS             | SKIPPED        | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
| CentOS 8         | SKIPPED           | PASS             | SKIPPED        | PASS               | PASS         |
+------------------+-------------------+------------------+----------------+--------------------+--------------+
```

Testing runs mostly in containers where the container OS is the same as the VM OS so the same kernel is used. Unit testing cannot reliably be ran in a container, so it is on a separate snapshotted VM.

These tests exist in the dpdklab-ci repository (IOL). An older patch to include it in the dpdk-ci repository was submitted to the ci mailing list.

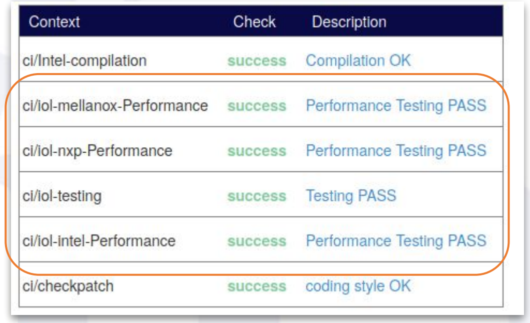
Where Results are Sent and Stored

Test finished

→ Add results to DB (can be viewed on the Dashboard)

All related tests for a tarball are finished

→ Grab results from DB and send report



Context	Check	Description
ci/Intel-compilation	success	Compilation OK
ci/iol-mellanox-Performance	success	Performance Testing PASS
ci/iol-nxp-Performance	success	Performance Testing PASS
ci/iol-testing	success	Testing PASS
ci/iol-intel-Performance	success	Performance Testing PASS
ci/checkpatch	success	coding style OK

Related tests are grouped such that when they are done, a report is sent out to the patchwork test report mailing list. Reports are sent as one group per vendor (iol-$vendor-Performance) and a single group for smoke testing (iol-testing).

By default, if tests fail, results are also sent to the environment maintainers.

Subtree testing is also sent to the same mailing list and environment maintainers.

https://lab.dpdk.org

Security

Since this is a public CI which also stores sensitive results, extra precautions have been lead by the UNH-IOL and put into place.

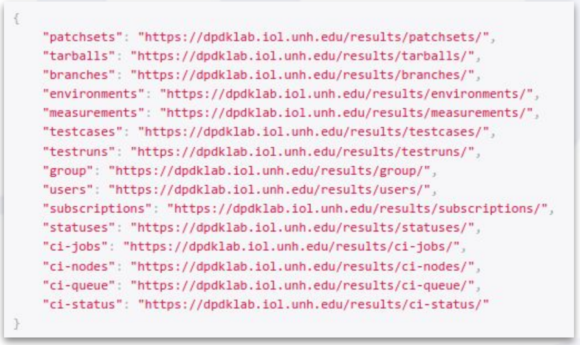
- CI nodes are only accessible from within the private network. A VPN is required to access the network.
- Bare metal machines use OverlayFS for their root file system. This allows rebooting the machines to remove any persistence created by CI users. These systems are rebooted once a week.
- Smoke testing runs on unprivileged containers when applicable. If the test cannot run on a container, a snapshotted VM is used instead.
- Test scripts are shared via a read-only NFS share.
- The REST API is behind the private network to avoid possible outside enumeration. Vendors can utilize the API by connecting through the VPN.

DB and REST API

State of the CI and results get sent to the DB. This all goes through the REST API for validation.

Participants cannot view other vendors results.

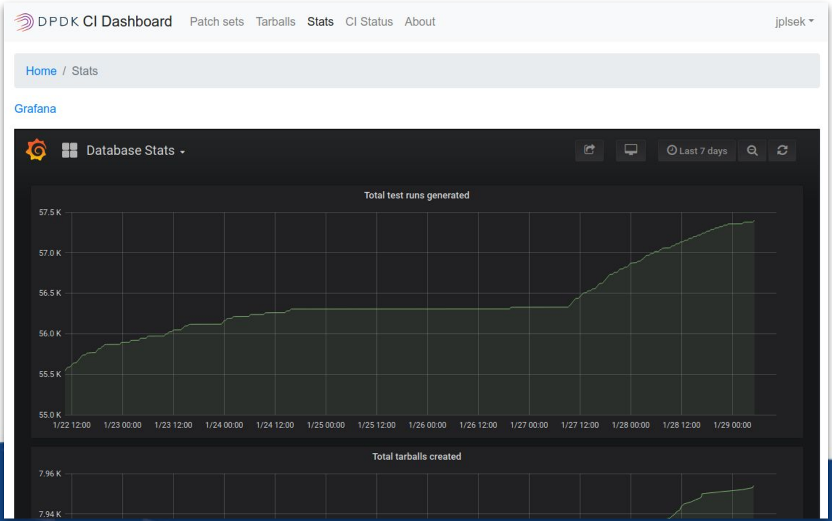
- Patchset: Patch series meta information.
- Tarballs: Tarball meta information. Contains both patch series and subtree tarballs.
- Branches: Subtree information.
- Environments: Vendor and VM system information.
- Testruns: Test runs. Contains test results, which test case, and which tarball.
- Group and Users: Vendor and user meta information.
- Subscriptions: How emails are sent (includes mailing lists).
- CI-*: Proxy API to Jenkins to show the CI status from within the dashboard.

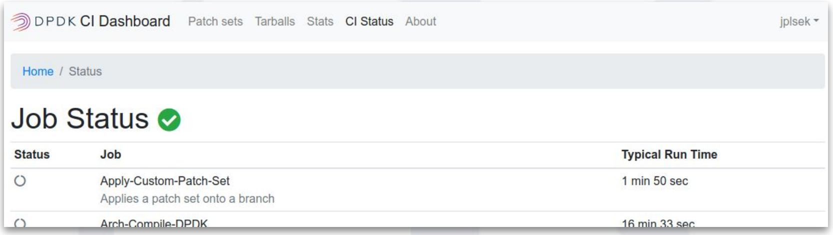


```
{
    "patchsets": "https://dpdklab.iol.unh.edu/results/patchsets/",
    "tarballs": "https://dpdklab.iol.unh.edu/results/tarballs/",
    "branches": "https://dpdklab.iol.unh.edu/results/branches/",
    "environments": "https://dpdklab.iol.unh.edu/results/environments/",
    "measurements": "https://dpdklab.iol.unh.edu/results/measurements/",
    "testcases": "https://dpdklab.iol.unh.edu/results/testcases/",
    "testruns": "https://dpdklab.iol.unh.edu/results/testruns/",
    "group": "https://dpdklab.iol.unh.edu/results/group/",
    "users": "https://dpdklab.iol.unh.edu/results/users/",
    "subscriptions": "https://dpdklab.iol.unh.edu/results/subscriptions/",
    "statuses": "https://dpdklab.iol.unh.edu/results/statuses/",
    "ci-jobs": "https://dpdklab.iol.unh.edu/results/ci-jobs/",
    "ci-nodes": "https://dpdklab.iol.unh.edu/results/ci-nodes/",
    "ci-queue": "https://dpdklab.iol.unh.edu/results/ci-queue/",
    "ci-status": "https://dpdklab.iol.unh.edu/results/ci-status/"
}
```

Dashboard

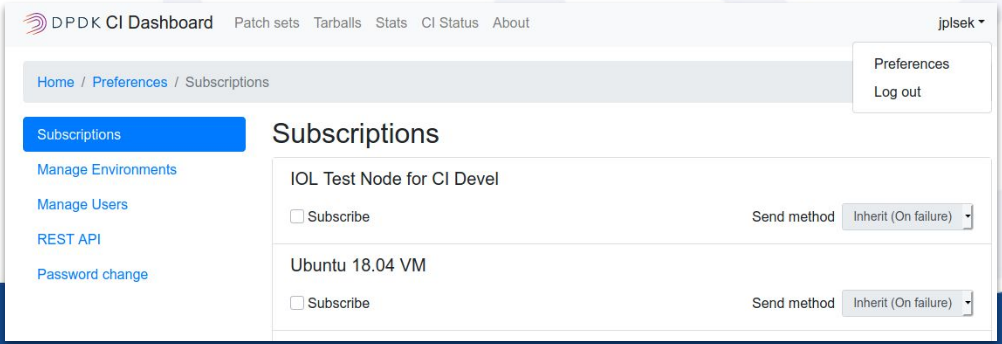
- Participants can manage their subscriptions, environments, API keys
- View Jenkins CI Status
- View stats with Grafana





Job Status ✓

Status	Job	Typical Run Time
○	Apply-Custom-Patch-Set Applies a patch set onto a branch	1 min 50 sec
○	Arch-Compile-DPDK	16 min 33 sec



Subscriptions

IOL Test Node for CI Devel

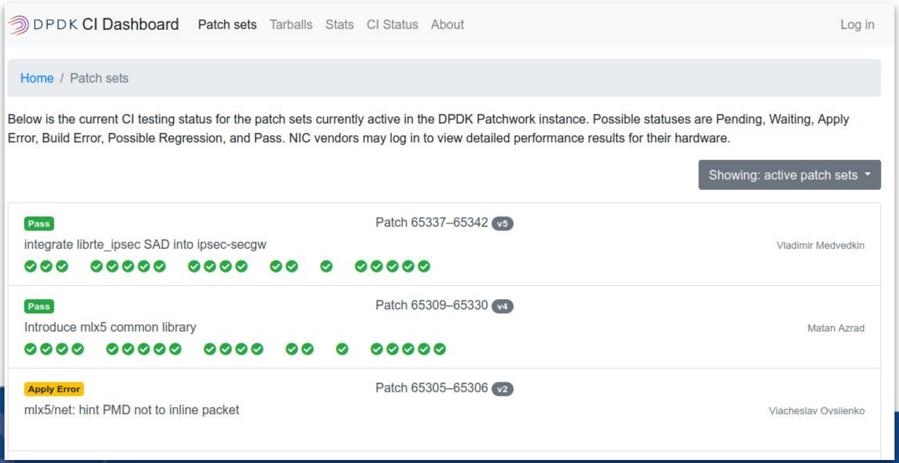
☐ Subscribe Send method Inherit (On failure)

Ubuntu 18.04 VM

☐ Subscribe Send method Inherit (On failure)

Dashboard

- Uses REST API to populate results
- Uses Patchwork REST API to populate patch information (cached)
- View overall results outside of Patchwork or emails

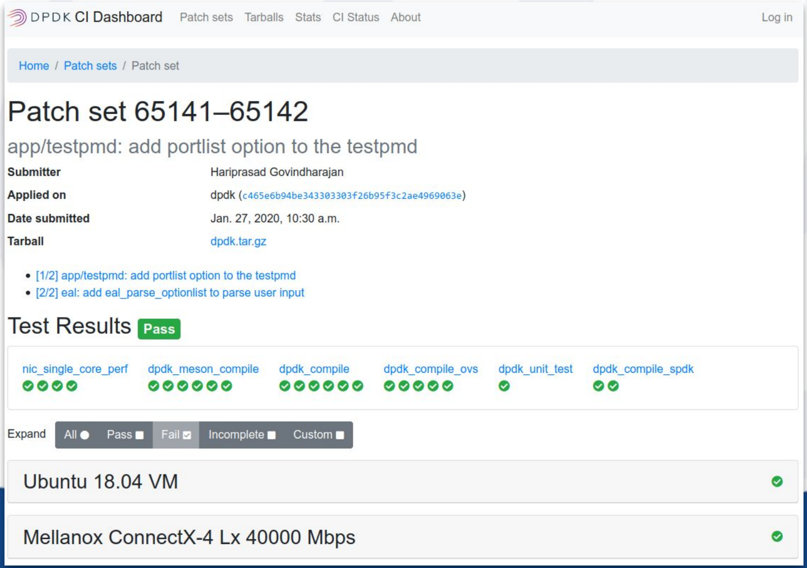


Below is the current CI testing status for the patch sets currently active in the DPDK Patchwork instance. Possible statuses are Pending, Waiting, Apply Error, Build Error, Possible Regression, and Pass. NIC vendors may log in to view detailed performance results for their hardware.

Pass — Patch 65337–65342 v5
integrate librte_ipsec SAD into ipsec-secgw — Vladimir Medvedkin
✓✓✓ ✓✓✓✓✓ ✓✓✓✓ ✓✓ ✓ ✓✓✓✓✓

Pass — Patch 65309–65330 v4
Introduce mlx5 common library — Matan Azrad
✓✓✓✓ ✓✓✓✓✓ ✓✓✓✓ ✓✓ ✓ ✓✓✓✓✓

Apply Error — Patch 65305–65306 v2
mlx5/net: hint PMD not to inline packet — Viacheslav Ovsiienko



Patch set 65141–65142

app/testpmd: add portlist option to the testpmd

Submitter	Hariprasad Govindharajan
Applied on	dpdk (c465e6b94be343303303f26b95f3c2ae4969063e)
Date submitted	Jan. 27, 2020, 10:30 a.m.
Tarball	dpdk.tar.gz

- [1/2] app/testpmd: add portlist option to the testpmd
- [2/2] eal: add eal_parse_optionlist to parse user input

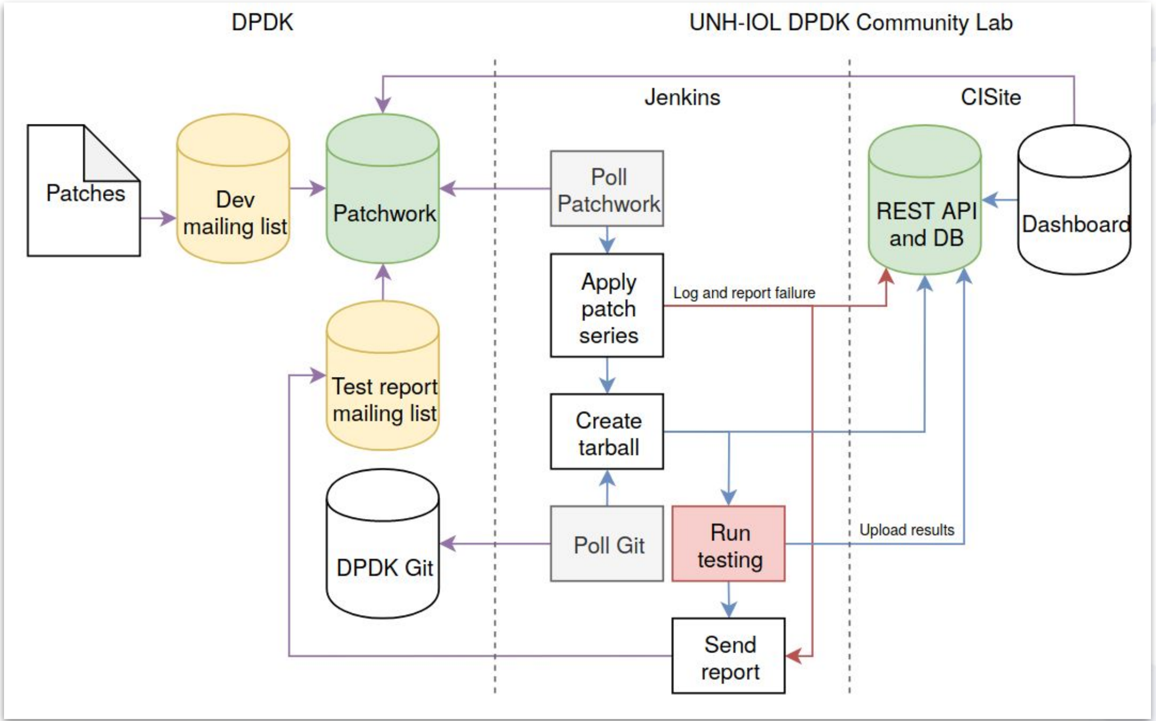
Test Results Pass

nic_single_core_perf	dpdk_meson_compile	dpdk_compile	dpdk_compile_ovs	dpdk_unit_test	dpdk_compile_spdk
✓✓✓✓	✓✓✓✓✓✓	✓✓✓✓✓✓	✓✓✓✓✓	✓	✓✓

Ubuntu 18.04 VM ✓

Mellanox ConnectX-4 Lx 40000 Mbps ✓

Process Overview



Roadmap / Work Queue

- Broadcom ovs_perf once dts performance testing is online
  - Waiting on Broadcom to finalize hardware turning
- DPDK+SPDK unit testing
  - SPDK team is updating their unit testing, once complete, these tests will be added
- Arm compile testing and additional performance testing
  - Waiting on hardware from ARM
- Crypto and VirtIO testing
  - Will use new hardware from Intel
  - Working with dev team to update tests to run on other hardware / architectures