DPDK

DPDK Community Lab

Hosted at the
University of New Hampshire's
InterOperability Laboratory
(UNH-IOL)

**Policies and Procedures**

Version 0.2 (DRAFT)
2018-11-13

Table of Contents

# Revisions

| Rev ID | Date | Description |
|---|---|---|
| 0.1 | 2018-07-03 | Initial Draft version |
| 0.2 | 2018-11-13 | Update name to “DPDK Community Lab”<br>Fix some grammatical issues |

# 1. Introduction

1.1. The Linux Foundation's DPDK Project has established a DPDK Community Lab at the University of New Hampshire InterOperability Laboratory (UNH-IOL). This document records the DPDK Project's policies, procedures and specifications for use of the DPDK Community Lab. This document may be updated from time to time by the approval processes defined by the Linux Foundation's DPDK Project.

# 2. Objective and Scope of the Project

2.1. There are four main goals of the DPDK Community Lab as identified at the time it was established:

1. Identify any regression in DPDK performance. The DPDK Community Lab will host equipment provided by multiple Members to UNH-IOL and run basic performance tests on new patches/patch sets on an automated basis. The aim of these tests is to determine if there has been any unexpected drop in DPDK performance as a result of recent changes. Members of the Linux Foundation DPDK Project ("DPDK Members") may provide, access, and maintain their equipment per the terms of the UNH-IOL Usage Agreement.

2. Identify any regression in the performance of DPDK-enabled applications. DPDK Members will be able to submit software applications to be run in the community lab. The details of how this can be managed may be complex, so this capability may be added at a later stage.

3. Demonstrate any new feature performance of DPDK. In each release, there may be some new performance optimizations or some new solutions. DPDK Members will be able to utilize the platforms in the community lab to show the new performance gains through DPDK-enabled applications.

4. The DPDK Community Lab may also be used as a training or demo lab for DPDK events.

# 3. DPDK Community Lab Usage Guidelines

The following high-level guidelines help to shape and define the following policies and specifications.

## 3.1. Use of DPDK Community Lab for Marketing Purposes

3.1.1. Data generated by the DPDK Community Lab shall not be used for competitive marketing purposes between DPDK Member equipment providers.

3.1.2. Data from the test lab may be used to demonstrate and market the performance improvement of new DPDK features in a DPDK Member agnostic manner.

# 4. DPDK Community Lab Policies

The following policies are established by the Linux Foundation's DPDK Project to define use and access to the DPDK Community Lab.

## 4.1. DPDK Member-Provided Platforms Access Policy

4.1.1. DPDK Members providing platforms to the test lab will have access to their platforms remotely or on-site upon request. Only the providing Member may have access to their provided platform(s). Resources provided by a DPDK Member will not be available to other DPDK Members without the express permission of the providing DPDK Member.

## 4.2. DPDK Member-Provided Platforms Update Policy

4.2.1. DPDK Members may provide hardware, firmware and/or software updates to their provided platform(s) at any time; however, Members must notify the UNH-IOL contacts in advance of any changes. UNH-IOL will work with the Member to schedule downtime for hardware updates and remote access for Member-driven firmware/software updates and/or support of UNH-IOL applied firmware/software updates. This requirement is intended to help prevent upgrades from causing false negatives in the automated testing systems.

## 4.3. DPDK Member-Provided Platforms Testing Policy

4.3.1. DPDK Members will provide all test scripts, software, and hardware required to test their platforms in the test lab.

4.3.2. DPDK Members will be responsible for the performance tuning of their software and hardware platforms. It is expected that the test scripts will produce acceptable results for DPDK releases and the DPDK master branch in the absence of patches that cause a performance regression.

4.3.3. Test scripts are required to provide results in the agreed test lab format.

4.3.4. Total test time should be limited to 30 minutes per patch-set, and shall not exceed 60 minutes per patch-set. If multiple DUTs are on a single system such that tests cannot be run simultaneously, note that this time limit applies to all tests running sequentially.

## 4.4. Provided Platform Result Disclosure Policy

4.4.1. DPDK Members must explicitly consent to disclose results to other participants and/or publically through any available method (eg: anonymous Dashboard display). Such consent, once given, is given for all future results gathered until such time as such consent is revoked.

4.4.1.1. DPDK Members MUST declare a Primary Contact who is responsible for giving such consent in writing to UNH-IOL (e-mail is sufficient).

4.4.1.2. The Primary Contact MAY define a Backup Contact if the Primary Contact is unavailable; however, this must be communicated to the UNH-IOL in advance.

4.4.2. DPDK Members may elect to keep gathered results to their sole benefit. As example, this may be appropriate when the Member is tuning their system.

4.4.3. The Primary Contact from each member company must provide UNH-IOL in writing with a list of individual e-mail addresses and/or a company mailing list to which to send e-mail failure reports. Members may also request e-mails on success when performing setup or maintenance work on a device, but the normal state shall be to only send e-mail on failure.

4.4.3.1. Each of these contacts will be given an account on the dashboard which they may use to view results (see section 5.6) and adjust their e-mail preferences.

## 4.5. DPDK Member Participation Disclosure Policy

4.5.1. DPDK Members participating in the DPDK Community Lab agree to be disclosed as participants in the DPDK Community Lab.

4.5.2. Specific devices from DPDK Members, and the specific results of those devices shall not be disclosed.

# 5. DPDK Community Lab Specification

The following specifications are established by the Linux Foundation's DPDK Project to define specific requirements of participation in the community lab.

## 5.1. DPDK Branch(s) to test

5.1.1. At this time, patch-sets that apply cleanly to master are the only patch-sets tested.

## 5.2. Test Script Requirements Specification

5.2.1. Vendors provide one test script that performs all desired tests for their Platform.

5.2.2. The Vendor test script will be run for each patch set by the UNH-IOL Test Harness, which is currently a Jenkins Pipeline job.

5.2.3. Vendor provided test scripts will compare a patch-sets performance versus a 'Baseline' behavior. For each test performed against a Platform, this 'Baseline' is an **expected_value** from the same test that will be performed against the DPDK Master. The Threshold is the absolute or percent tolerance for results below the expected value; results less than the Baseline minus the Threshold will be considered to be test failures. The test script will internally maintain the list of expected values in some form that is out of scope of this document.

5.2.4. The Member script shall support an --update-expected argument which will cause the script to update all expected values based on the results of the current test run. This option will be supplied by the UNH-IOL test harness during a periodic run using the latest DPDK master with no patchset applied on top. Periodic runs will occur at least weekly if not nightly.

5.2.5. Before running the DPDK Member script, the test harness will create two files in the current working directory of the script.

5.2.5.1. The contents of master with the applied patch-sets is supplied as the tarball under test which will be placed into the current working directory with the file name **dpdk.tar.gz.**

5.2.5.2. In addition, a second file named **tarball.json** will contain metadata about the tarball. In particular, it will at least contain the following top-level keys:

```
{
  "branch": "master",
```

```
    “commit_id”: “abcdefabcdefabcdefabcdef”,
    “patchset”: “https://dpdklab.iol.unh.edu/results/patchsets/XYZ/”,
    /* … */
  }
```

5.2.5.2.1. **branch** identifies the upstream branch that the tarball is based on, e.g., master, dpdk-next-net

5.2.5.2.2. **commit_id** is the upstream commit of the head of the branch at the time the tarball was made, and

5.2.5.2.3. **patchset** is the patchset that has been applied on top of the branch. If the value of the patchset key is null, then this tarball is taken directly from the upstream git branch with no further patches applied.

5.2.6. The test script is expected to return a consistent set of test cases with the same input parameters for each Test Run. If these test cases or input parameters are changed, the Member must notify the UNH-IOL about this change so the results can be stored as part of a new test environment. This will allow meaningful comparisons of results from the same test environment.

## 5.3. Test Script Result Reporting Specification

5.3.1. DPDK Members scripts shall report the results from their scripts in the following JSON format. An example is shown below:

```
{
  "results": [
    {
      "parameters": {
        "frame_size": {
          “value”: 64,
          “unit”: “bytes”
        },
        "txd/rxd": {
          “value”: 1024,
          “unit”: “descriptors”
        }
      },
      "throughput": {
        "result": "PASS",
        "delta": -0.452,
        "unit": "Mpps"
      }
    },
    /* ... */
  ]
}
```

5.3.2. The JSON file will contain a top-level key **results** whose value is a list containing each individual performance result. Each list element shall be a dictionary of two keys. One key must be named “parameters”, which is a dictionary of the input parameters for the test case. The second key shall be named after the output measurement being recorded.

5.3.3. Each input parameter (e.g., frame_size) shall be represented as a dictionary containing two keys: **value** shall contain the numeric value of the measurement and **unit** shall be a string describing the units.

5.3.4. Each output measurement (e.g., throughput) shall be represented as a dictionary containing three or four keys. The **result** key shall contain the string “PASS” or “FAIL” depending on the whether the measurement was within the threshold defined by the Member in their script. The **unit** key shall contain the units for the test measurement.

5.3.4.1. A Member must supply a key **delta** whose value is the difference between the actual and expected measurements (such that negative values indicate values lower than expected). In this case, no information is supplied or recorded regarding the absolute measurement.

5.3.4.2. A Member may optionally supply a key **expected_value** whose value is the absolute baseline measurement. In this case, the absolute measurement can be calculated and may be presented to the Member, such as the dashboard discussed in section 5.4. Note that these absolute values will never be exposed to anyone other than the Member company.

5.3.5. The input parameters for each test case are expected to be consistent for every Test Run. When a Member supplies their system, they are expected to give the input parameters for their tests which will be entered into the database and matched with the results given in the results JSON output. If these test cases are changed, the Member must notify the UNH-IOL about this change.

5.3.5.1. For the example given in 5.3.1 above, the parameters "frame_size" and "txd/rxd" and the key "throughput" are used to look up the proper entries in the measurements given by the specified environment.

## 5.4. Test Harness Requirements Specification

5.4.1. The DPDK Community Lab will maintain a Test Harness that executes each Member’s test script for each provided Platform for each patch-set.

5.4.2. The Test Harness will maintain a record of the test environment that the test script was run in. The test environment is considered to be different if any of the following occur:

5.4.2.1. If the Member changes their test script's input parameters, or tests covered.

5.4.2.2. If the Member's Platform changes, including firmware changes.

5.4.3. The Test Harness will periodically update the **expected_value** for each vendors test script's Baseline comparison value.

5.4.4. The Test Harness will maintain a database of confidential results that can recall all reported test script results. Access to all database information is provided only through the following Dashboard specifications.

## 5.5. Public Dashboard Specification

5.5.1. The public will have access to the following information:

5.5.1.1. The Patch-set tested

5.5.1.2. The overall aggregate result of all testing across all DPDK Members devices and tests performed.

5.5.1.2.1. If all tests Pass, as indicated by the DPDK-Members scripts, then a Pass will be displayed.

5.5.1.2.2. If any single test Fails, then the public will see a "Possible Regression" for that patch result.

5.5.1.2.3. If the patch-set fails to apply properly

5.5.1.3. The Patch-set title

5.5.1.4. The Patch-set submitter

5.5.2. Additional vendor-specific information to be added to the public dashboard must be approved by recorded vote of participating Members. Note that this does not apply to bug fixes and new functionality that does not expose additional vendor-specific information; these may be applied by UNH-IOL at any time.

## 5.6. Private DPDK-Member only Dashboard Specification

5.6.1. DPDK-Member only Dashboard will display the following information

5.6.1.1. The result of the patch-set tested per provided Platform

5.6.1.2. The delta-values of the script output, per test performed.

5.6.1.3. Vendors will see the following for eachof their Platforms, Per Test:

5.6.1.3.1. If the test is a Pass, as indicated by the DPDK-Members scripts, then a Pass will be displayed.

5.6.1.3.2. If the test is a Fail, then the vendor for that platform will see a Fail for that test result..

5.6.1.4. If the patch-set fails to apply properly

5.6.1.5. The patch-set title

5.6.1.6. The patch-set submitter

5.6.2. The DPDK-Member only Dashboard will only be displayed to the logged-in users associated with the DPDK-Member.

5.6.3. Additional vendor-specific information to be added to the private DPDK-Member only dashboard must be approved by recorded vote of participating Members. Note that this does not apply to bug fixes and new functionality that does not expose vendor-specific information; these may be applied by UNH-IOL at any time.

# 6. Appendix: Definitions and Acronyms

The following definitions and acronyms are established by the Linux Foundation's DPDK Project to aid in understanding this document:

| | |
|---|---|
| Baseline | An **expected_value** from the same test that will be performed against the DPDK Master. |
| DPDK | Data Plane Development Kit |
| "DPDK Member" / "Member" | A company recognized by the Linux Foundation's DPDK Project as a Member |
| DUT | Device under test |
| expected_value | The value expected when a test is performed against the DPDK Master rather than a patch-set. |
| Patch-set | A related set of patches submitted for inclusion in DPDK by the project maintainers |
| Platform | Any DPDK capable hardware provided by the DPDK Member for testing in the DPDK Community Lab |
| "Possible Regression" | A result for a patch-set that triggers a DPDK Member-defined testing threshold as a failure |
| Test Harness | Mechanism maintained by the Performance Lab to execute test scripts against a Platform for a given patch-set. |
| Test Script | Member provided script for one or more Platforms. |
| Test Run | A single execution of a Member's test script against a patch-set for a Platform. |
| Threshold | The absolute or percent tolerance for results below the expected_value; results less than the Baseline minus the Threshold will be considered to be test failures |
| UNH-IOL | University of New Hampshire InterOperability Lab |