# DPDK Performance Test Lab Proposal

## Contents



## Introduction

The goal of this document is to describe the purpose, scope and implementation options for an open DPDK performance test lab.

Based on the expected budget for the DPDK project when it moves to The Linux Foundation, it's not realistic to have a complete, open CI lab similar in size and scope to FD.io's CSIT (https://wiki.fd.io/view/CSIT). Because of this, a distributed CI system integrated with Patchwork has been put in place for basic build testing of patches (build test, checkpatch etc.). See the archives of the ci@dpdk.org mailing list (http://dpdk.org/ml/archives/ci/), the DPDK Patchwork page (http://dpdk.org/dev/patchwork/project/dpdk/list/) and the archive of the test-reports@dpdk.org mailing list (http://dpdk.org/ml/archives/test-report/) for details of this distributed CI system.

During discussions on moving the DPDK project to The Linux Foundation, a strong desire was expressed to have an open lab for performance testing. It was felt that performance numbers generated in an open lab would be more transparent, more trustworthy and more easily accessible than those generated in private vendor labs. Restricting the scope of an open DPDK lab to just performance testing initially should allow it to fit within the expected budget of the DPDK project when it moves to The Linux Foundation.

## Purpose

There are four main goals of the performance test lab:

1. Identify any regression in DPDK performance. The DPDK performance test lab will host equipment from multiple vendors and run basic performance tests on a daily 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.

2. Identify any regression in the performance of DPDK-enabled applications. Gold and Silver Members of the project will be able to submit software applications to be run in the performance test 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, we may have some new performance optimizations or some new solutions. Gold and Silver Members of the project will be able to utilize the platforms in the performance test lab to show the new performance gains through DPDK-enabled applications.

4. The open performance test lab could also be used as a training or demo lab for DPDK events. We could allocate some hardware during a training session or event to show DPDK performance live.

All testing performed in the lab will be open and public. Controls will obviously be required on who can make changes to the lab, but all test setup, configuration and results will be publically accessible.

It’s important to note that the DPDK performance test lab is not intended as a sales/marketing tool for vendors to use to promote the benefits of their hardware platforms or software solutions. It’s a reference lab for identifying performance regressions in DPDK software.

## Hosting Options

The preferred method is to establish an open DPDK lab hosted by The Linux Foundation in one of their data centers. This will enable the lab to operate in an open and transparent way. This will require the following:

1. The DPDK project will need to have sufficient funds to pay hosting costs to The Linux Foundation, and staffing costs (partial Sys Admin, partial Release Engineer) to manage the equipment.
2. Vendors interested in participating in the lab will need to be prepared to donate hardware for the tests to be run on.

A second option is to adopt the model used by the OPNFV Pharos project where individual companies host open community labs. See https://wiki.opnfv.org/display/pharos/Community+Labs for details. The advantage of this model would be that the DPDK project would not need to pay hosting costs. The disadvantages include:

1. A reduction in openness/transparency, because the results are from individual vendor labs rather than from one hosted by a neutral entity (the Linux Foundation).
2. The need for each vendor to set up public access to their private lab infrastructure. This will require some maintenance costs which each vendor would need to fund.
3. The fact that the lab would be distributed would make keeping a consistent configuration more difficult. We should be running the same DPDK versions with the same patches and same configuration on all of the equipment in the DPDK to ensure consistency. If the equipment is all located in a central lab that's much easier to achieve than if it's distributed and controlled by multiple different vendors.
4. Submitting DPDK-enabled software applications to be run in the lab would be more complex with a distributed lab configuration, because anybody proposing such an application would need to deal with each vendor individually.

Because of these disadvantages, this is not the preferred option, but can be revisited if constraints on project budget make an LF-hosted lab unrealistic.

A third option is to follow the distributed model adopted for CI where performance testing is run completely in private and the results are published. This does not meet the main criterion of having an open lab, so it's not considered any further.

## Lab Size

The amount of equipment which can be hosted will depend on two main factors: the size of the project budget available to cover hosting and staffing costs for the lab, and the level of interest from vendors in contributing equipment to be hosted in the lab. We don't know either of these items yet, so the actual initial lab size will need to be determined later.

However, a reasonable estimate would seem to be to start with a minimum of ~20U of space for servers. This would allow 4U per vendor for up to 5 vendors. This figure

will need to be adjusted when we have a better idea of the level of interest and specific hardware requirements.

# Performance Regression Test

## Purpose

The aim of performance regression testing is to run basic performance tests on a regular basis to identify any regression in performance. The same tests should be run on multiple architectures with NICs from multiple vendors to give the broadest possible test coverage.

## Test Scope

Tests will be run on the DPDK master on a daily basis. This will allow identification of any performance regression due to patches applied in the last 24 hours.

In addition, each new version of a stable release and an LTS release (when DPDK supports LTS releases) will be tested. New versions of stable and LTS releases do not happen very often so they do not need to be tested on a daily basis.

## Test Cases

Additional tests can be added over time but the initial set of proposed tests are as below:

1. Maximum Throughput with Zero Packet Loss

The purpose of this test is to determine the maximum throughput of a NIC/PMD with zero packet loss. This test will help to identify any regression in the performance of the particular PMDs being tested, the core DPDK libraries, or the l3fwd sample application.

Testing will be as per RFC2544, section 26 (Throughput). Specifically:

Objective: To determine the DUT throughput as defined in RFC 1242.

Procedure: Send a specific number of frames at a specific rate through the DUT and then count the frames that are transmitted by the DUT. If the count of offered frames is equal to the count of received frames, the fewer frames are received than were transmitted, the rate of the offered stream is reduced and the test is rerun. The throughput is the fastest rate at which the count of test frames transmitted by the DUT is equal to the number of test frames sent to it by the test equipment.

Tests will:

- Use the l3fwd sample application
- Be run for 60 seconds duration
- Have an acceptable packet loss rate of 0
- Use 256 flows per port

2. Maximum Single Core Performance

The purpose of this test is to determine the maximum throughput that a single core can process. This test differs from the previous one in that if packet loss occurs, we will record and report the packet loss rate, but will not reduce the throughput until zero packet loss is achieved. This test will help to identify any regression in the performance of the fast path of particular PMDs.

Tests will:

- Use the testpmd application
- Be run for 60 seconds duration
- Use 256 flows per port

3. Vhost/Virtio (Phy-VM-Phy)

The purpose of this test is to measure performance in virtualized environments. Testpmd is used to set up a simple vhost-virtio PVP test as shown in the diagram below. As in the previous test, if packet loss occurs we will record and report the packet loss rate, but will not reduce the throughput until zero packet loss is achieved. This test will help to identify any regression in the performance of vhost-user, virtio-user, the core DPDK libraries, or the testpmd application.

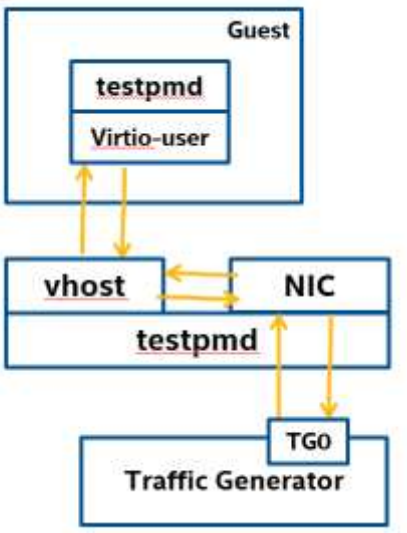


Figure 1: Vhost/virtio PVP test setup

Tests will:

- Use the testpmd application
- Be run for 60 seconds duration.

## Software Applications

The tests described in the Performance Regression Test section above use basic DPDK applications like l3fwd and testpmd. It should be possible for members of the DPDK project to submit more complex software applications to be run as part of the regression test suite.

Further details are TBD.

## Test Results

Test results will be publically accessible and automatically posted to a website so they're accessible to everybody. Discussion will be required with the Linux Foundation team on the best method for publishing the results. It may be possible to use infrastructure that is already in place for other LF-hosted projects, or alternatively it may be necessary to have dedicated active/backup database servers specifically for DPDK.

Some judgment will be required when interpreting the results. Some proposed changes to DPDK will have unavoidable performance impacts which will be known in advance (e.g. the expansion of the mbuf to 2 cache lines in release 1.8). There may also be small variations in performance results between test runs.

In cases where there are unexpected changes in performance, discussion should occur on the dev@dpdk.org mailing list on the reasons for the change and ways to mitigate the impact.

## Traffic Generator

Any software packet generator can be used. It would be beneficial to use the same packet generator on all platforms in the lab in order to ensure consistency.

## Hardware Requirements

TBD.

## Terminology

All terminology used in this document is as defined in RFC1242 (Benchmarking Terminology for Network Interconnection Devices).

## Open Issues

1. Do we need a facility for tests to be run on demand? This might be useful when trying to narrow down a performance regression to the particular patch that introduced it. It might also be useful to allow engineers to test the performance impacts of their patches before they're submitted. Allowing this does add complexity to the scheduling process though.

2. To ensure consistency, we need to make sure that test cases are implemented in a consistent way across all platforms in the lab. We need to agree who reviews and approves new test cases to ensure this.
3. We need to verify whether a SW traffic generator can meet RFC2544 requirements for the "Maximum Throughput with Zero Packet Loss" test case. If not, we may need consider changing the test case to just maximum throughput, removing the zero packet loss requirement.
4. Do any companies have concerns over the need for legal review/approval of performance data?
5. Will every contributor to the lab use the DPDK test framework, or will other test frameworks need to be supported?
6. We need to determine a user-friendly way to format and display the results. These should be posted on a publicly accessible website, but we need to determine the infrastructure required to support this.
7. Infrastructure requirements for scheduling of tests (e.g. using Jenkins) need to be determined.