White Paper

OpenFlow Performance Testing

TABLE OF CONTENTS

Abstract

The process of testing a switch to verify it meets Open Networking Foundation (ONF) OpenFlow specifications is well understood and standardized. Take a switch to a properly equipped lab, which runs a standard set of conformance tests, and—assuming the switch passes—the ONF provides a metaphorical seal of approval for OpenFlow conformance.

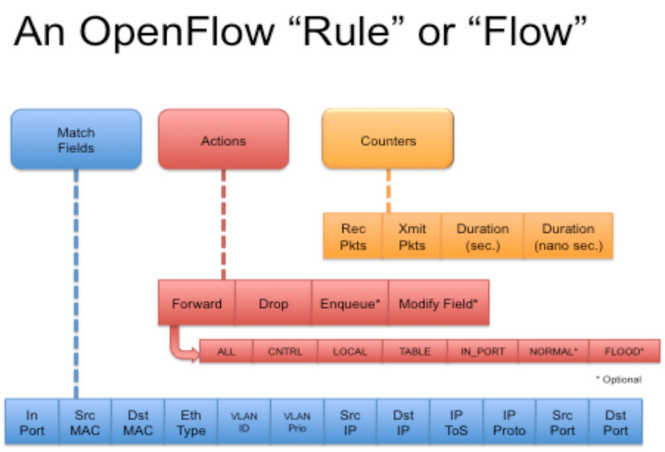
OpenFlow performance testing, however, is still in its infancy. While a few open-source tools exist for testing OpenFlow performance, and these features are starting to appear in commercial testing products, there's no standard for comparing the performance of OpenFlow products. Yet the networking vendors building SDN products and the companies deploying them have a real "need to know" not only that OpenFlow is meeting specs, but also that it is performing as expected and desired. This paper explains the importance of OpenFlow performance testing and describes test methodologies companies can use to do their own performance testing for OpenFlow.

Why OpenFlow Performance Testing is Needed

While OpenFlow is a standard—and the ONF has strict requirements for a switch to be considered conformant with the specification—conformance testing says nothing about real-world performance. There are many differences in performance with both OpenFlow controllers and switches, and choosing the wrong product for your deployment can have disastrous consequences. The only sure way to avoid this scenario is through thorough performance testing.

The Parameters of This Paper

OpenFlow solutions include one or more controllers and one or more switches. While what ultimately matters to end customers is the performance of the complete system, system builders and integrators want to evaluate the performance of components separately to choose the best components.



This paper focuses on Ethernet switch testing only (we'll cover controller and complete system testing in future papers). The examples described in this paper will focus on hardware switches, which are more complex and interesting for performance testing than software switches. Whereas software switches run on standard servers, hardware switches use ASICs that accelerate throughput. Still, the same methodology described here can be applied to software switches as well.

Finally, although there are multiple versions of OpenFlow available, this paper will focus on OpenFlow 1.3.0, because it incorporates multiple table pipeline processing and group tables; it is a commonly used version; and it provides a core concept for testing that can be updated and applied to OpenFlow 1.4.0 and beyond.

Goals and Challenges for OpenFlow Performance Testing

OpenFlow performance testing differs from typical switch testing in that it is not entirely focused on packet forwarding performance. OpenFlow performance testing deals both with factors such as how fast a switch can process 'flow-mod' messages, and orthogonal packet processing through the OpenFlow pipeline. The term flow-mod is short-hand for OpenFlow messages that a controller sends to a switch to add, modify or delete rules from the switches forwarding table. Once an OpenFlow 'rule' is installed in a switch's hardware forwarding table, packets are forwarded typically at line rate, which can be verified with well-known switch performance testing methodologies. The OpenFlow pipeline refers to multiple flow tables in the packet forwarding plane introduced in OpenFlow 1.1.0. Packets can now be processed against multiple matches, and execute multiple actions such as packet modification and forwarding to a destination port.

OpenFlow performance requirements for a switch can differ greatly depending on the solution deployment. Take, for example, the key OpenFlow performance metric of how fast a controller can add new flows into a switch's forwarding table. A solution that uses OpenFlow to implement an Ethernet Exchange Point might require only a few new OpenFlow flows per second, whereas an OpenFlow solution that implements dynamic IP routing may require hundreds of new flows per second.

Therefore, it's important to measure the number of flow-mods per second that can be sent from the controller and installed in the forwarding table of the switch. The bottleneck in this pipeline usually occurs inside the switch, between the switch's CPU and the ASIC that stores the forwarding table. Performance test results among different switches can vary dramatically, from single-digit flow-mods per second to many hundreds of flow-mods per second—even within the same switch, before and after a firmware upgrade.

Considerations for OpenFlow Performance Testing

We have identified six tests for gauging OpenFlow performance of an Ethernet switch. They are:

- Table capacity testing
- Flow-mod performance
- Packet in/out performance
- Table-miss performance
- Flow statistics testing
- Pipeline Processing performance

For each test, the equipment used for testing should emulate the behavior of an OpenFlow controller, and it should also connect to the data plane ports on the switch. In this way, the test equipment can send packets through the switch to verify that the forwarding rules inserted by the emulated controller are installed and implemented properly.

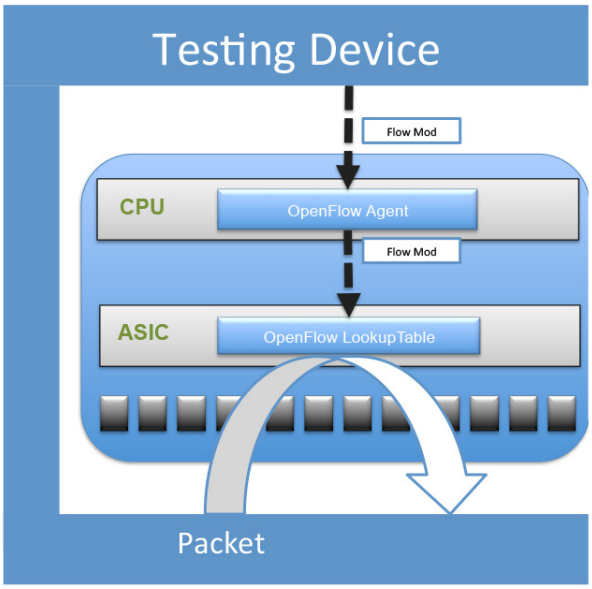


Table Capacity Testing

OpenFlow 1.3.0 was designed for one or more tables of rules (flows) that could be implemented using TCAM, a type of memory used by almost all switches that searches first on memory content opposed to where the content is stored. Different switches have different TCAM sizes, and the amount of TCAM used varies based on vendor optimizations and which flow rules are inserted.

To avoid confusion when discussing OpenFlow table capacity, it's helpful to refer to the 'full 12-tuple match,' which refers to an OpenFlow rule that matches on all 12 header fields. Some switch vendors also support optimizations that obscure the table capacity, such as placing OpenFlow rules that match only Layer-2 header fields into Layer-2 memory instead of TCAMs.

Broadly, table capacity testing entails validating that a flow got installed; determining when a table is full; figuring out how many flows are in the full table; and repeating this process for each table under test.

More specifically, when developing a methodology for testing the OpenFlow table capacity per table of an Ethernet switch, you need to consider the following:

- **Accounting for different table capabilities**—Be sure to run capacity tests with different match fields such as Layer 2-only and Layer 3-only matches in addition to full 12-tuple matches to check for any optimizations.
- **Validating multiple table scale**—Confirm the number of tables supported by switch (ex. 255), send flow mods for each table, and again run capacity tests on different match fields.
- **Validating that a flow is actually installed**—There are three validation options that provide increasing levels of accuracy.
  - **Option 1:** Send the flow-mod message and simply wait for the TCP acknowledgment to confirm the switch received the message.
  - **Option 2:** Send the flow-mod followed by an OpenFlow barrier request message. If the switch sends a barrier reply, it means the flow-mod message has been fully processed by the switch.
  - **Option 3:** Send the flow-mod and add data plane verification. This approach confirms the exact time a flow is installed and is also useful for verifying if flows are being forwarded in software or hardware. Software forwarding can be used by vendors to increase table size and capabilities, but can dramatically impact packet throughput. Use a test methodology such as the IETF RFC-2544 to verify the forwarding performance for packets matching each flow.

- **Determining if a table is full**—Install flows per table until the switch sends an error code (ofp_error_msg with OFPET_FLOW_MOD_FAILED, ALL_TABLES_FULL, or OVERLAP) indicating why it is full. Be sure to set the timeout value on the flows to be long enough that flows will not timeout and be removed by the switch during the test.
- **Figuring out how to count the number of flows successfully installed**—Check the counter on the test controller, and send a flow stats request to the switch to verify how many flows are installed in each table. Validate that the switch and the test module have the same number.
- **Deciding how many flows to insert at a time before stopping to verify that a table is full**—Smaller step sizes increase accuracy but also can take longer. A best practice is to start using large step sizes to find the general capacity range for the table, then use small step sizes to ascertain a more accurate count.

OpenFlow group tables require additional special considerations. Group tables are separate from flow tables and stored separately in memory. Where flow tables are likely to scale the match criteria, a "group identifier" in a group table; group tables can also scale the associated action list or "buckets" associated with a single group identifier, in addition to a single action in the bucket.

The group type "indirect" will match many incoming packets with a single action; a default route entry for example. For this group type, group table scale testing is done with the same considerations previously listed. For the group type "all" however, which has a multiple action bucket, option 3 under the flow validation methods is the preferred way to confirm all actions are being executed in the given action bucket. This group type will replicate a packet for each action in the bucket set, so table scale is directly tied to the number of actions taken, and not just the "group identifier" match which is the equivalent of a flow in a flow table.

Flow-Mod Performance

Flow-mod testing determines the latency between when the controller sends a flow-mod message to add, delete, or modify a rule in a switch's OpenFlow table and when the switch starts using that flow rule to forward packets. In other words, this tells you how fast a switch's OpenFlow lookup table can be updated. If it takes 100 milliseconds to install a single flow rule, the flow-mod performance will be 10 flow-mods per second.

Flow-mod performance on Ethernet switches can range from fewer than 10 flow-mods per second to hundreds or even thousands per second.

Some considerations for developing a methodology for flow-mod testing include:

- **How do you measure the time between the emulated controller sending a FLOW_MOD message and the switch installing it in the forwarding table?** As mentioned previously, the OpenFlow barrier request/reply messages are designed to verify a switch has processed an OpenFlow message. However, there is research[1] that indicates this may not be sufficient to provide fine-grained measurement of the time a flow rule is installed in the ASIC. A better method is as follows:
  - Delete all flows in all tables.
  - Create a lowest-priority default flow to DROP all packets, so nothing gets forwarded.
  - Start sending packets that match the flows being tested.
  - Send a group of FLOW_MODs to add flows; synchronize the timestamp on the sending of FLOW_MOD messages and on the first packets successfully received from the output port.
  - Calculate the time between sending the first FLOW_MOD and receiving the first packet for the last FLOW_MOD; divide by the number of flows added to calculate the average time to install a FLOW_MOD.
  - This test can be repeated for n tables so long as the instruction of each table except for the last table id includes the "Goto-Table" instruction. This will ensure that packet forwarding only occurs after the last table is populated with flows.
- **How many FLOW_MODs should you send before stopping to verify that they are installed?** Some tests add flows as fast as possible until the TABLE_FULL error is returned, or you could install 10, 20, 100, or some other number of flows at a time.
- **Does the time to install a flow increase with the total number of flows installed?** For instance, does it take longer to install the 2,000th flow as it does to install the 20th flow? This could adversely impact your network if your application uses large tables and there is research[1] indicating that, for large TCAMs, it takes longer to add flows as the TCAM memory fills up. In this case, test a range of flow table sizes (e.g., 10, 50, 100, 1,000, 2,000, up to a full table).
- **Does installing a flow exhibit different performance than modifying or deleting a flow?**[1] Indicates that modifying a flow can take longer than adding a new flow, so you should test adds, modifies and deletes.
- **Does a flow with wildcards take longer to install than a full 12-tuple exact match?** If so, you should consider testing with both exact match and wildcard fields, using different table sizes, and separately for add, change, and delete.

Packet In/Out Performance

One of the available OpenFlow actions for packets that match a rule is to encapsulate the packet and forward it to the controller (called a Packet-In message). Likewise, an OpenFlow controller can encapsulate a packet and send it to a switch to be forwarded through the network (called a Packet-Out message). Because this feature is very useful for many applications—including topology discovery, MAC learning, and captive portal—it's important to know exactly how fast encapsulated data packets can be sent between the switch and the controller.

The performance bottleneck for OpenFlow Packet-In and Packet-Out messages tends to be inside the switch, between the ASIC and the local switch CPU, or in the processing power of the local switch CPU. While switch performance varies greatly, typically you don't see more than 100 Packet-Ins/second from a switch.

Considerations when developing a methodology for Packet In/Out testing include:

- **Does the type of rule included in the Packet-Out message affect the latency of sending the packet?** Be sure to test with different flow-mods inside the packet-out message.
- **Determine how fast you send packets**—If the switch can handle only 10 Packet Outs/second, and you start by sending 1,000 packets/second, you might get a catastrophic failure. Send one packet, verify the latency (i.e., guess the breaking point), and gradually increase the rate over time until you reach the actual breaking point.
- **Find out if the size of the packet affects the latency**—It likely does. Try stepping through RFC-2544 frame sizes (64, 128, 256, 512, 1024, 1280, 1518, etc.).

Table-Miss Flow Entry Performance

The "table-miss" flow entry is a new feature in the OpenFlow 1.3.0 standard. It provides an alternative means to process unmatched flows, which in prior versions were automatically dropped. The table-miss flow entry provides a wildcard match of priority zero and allows the following actions: discard, goto_table, and output to controller. The output to controller is similar to packet-in processing. There are two basic performance tests to run against the table-miss flow entry:

- **Compare the drop performance of a table-miss discard for all flows vs. the current default action of drop for all flows**—When all tables are empty, all flows drop by default. This test will determine the processing overhead of using the table-miss wildcard entry with a drop action. Monitoring the switch indiscard count against time is one way to measure the performance of the table-miss entry.

- **The table-miss entry should also be tested using the output to controller action to gauge the maximum sustainable packet-in rate that can be forwarded to the controller**—There are two basic tests to run for this action:
  - Maximum throughput with all traffic hitting the table-miss entry (egress congestion performance). This test will measure the capacity of an OpenFlow switch to direct many flows ingressing many ports out a single port. The ingress rate in pps where drops begin should be measured by calculating the difference between all traffic in all datapath ports and the controller port. This will measure a switch's capacity to be dropped into an existing network with live traffic and have only the table-miss entry to process packets with.
  - Throughput performance with both matched and unmatched traffic. This test will measure the switches capability to deal with table-miss flows in addition to flows hitting match entries and determine whether and at what point unmatched traffic begins to impact the performance of flows with matches. Traffic drops on all OpenFlow ports and the egress controller port should be monitored for drops.

Flow Statistics Testing

The Statistics field in OpenFlow tables keeps track of the number of packets and bytes that have matched on each flow, as well as the time since the last packet matched the flow. The time calculation aids in the removal of inactive flows.

Considerations when developing a test methodology for flow statistics include:

- Keep in mind that flows include packet/byte counters, one for each entry in the table.
- Find out if polling the flow counters at a high rate affects any other operations on the switch.

OpenFlow Timer Testing

Each flow has hard and idle timeouts. The purpose of OpenFlow timer testing is to determine if these timeouts are accurate. For instance:

- Does the switch remove the flows when it is supposed to, or is there a delay? In other words, is it doing what I tell it to do?
- Is the precision affected by the total number of flows?
- Is the precision affected by the number of flows timing out simultaneously?

Pipeline Processing Performance

Pipeline processing performance testing gauges the ability of an OpenFlow switch to process packets orthogonally. That is to match on multiple criteria across multiple tables before taking an action. This is an important enhancement to the OpenFlow standard since version 1.1 that allows OpenFlow switches to perform more like today's switches that typically have multiple tables containing multiple types of records; for example MAC vs IPv4 addresses. Orthogonal lookups provide an important benchmark on how quickly a switch processes packets and the amount of latency added from an endless array of potential match and action set combinations.

Pipeline processing should incorporate the previously discussed methods of evaluating Flow-Mod performance in addition to pipeline processing. Specifically, pipeline performance is best measured starting with traffic running, the switch initially programmed to drop all packets, and adding flows until the final table is populated and traffic is received at intended destination.

The key considerations for pipeline performance testing are:

- Realistic packet matching and modification scenarios. For example match on dst_mac, vlan_id, dst_ip and modify with dec_ttl.
- Compare results only from the same pipeline processing test on each switch tested.
- Predetermine number of tables to use per test.
- Perform both simple pipeline and complex pipeline tests to provide an overall assessment of a switch's capability. For example, a simple test may have two matches and one packet modification across two tables, where a complex test will perform five matches and two packet modifications over ten tables.

Getting Started on OpenFlow Performance Testing

Knowing that OpenFlow products are performing as desired, as well as the ability to compare performance among products, is important for both the builders and the users of SDN products.

Tallac Networks and Spirent Communications have worked out detailed methodologies for OpenFlow performance testing that they are happy to share with interested parties.

Tallac Networks provides innovative technologies, consulting services, and products to simplify the implementation of Software-Defined Networking (SDN) solutions. More information can be found at www.tallac.com.

Spirent Communications develops innovative test solutions for engineers working within the communications industry, allowing them to evaluate the performance of the latest technologies, infrastructure, and applications to be deployed worldwide. Spirent also provides tools for service technicians and field test engineers to improve network quality and make troubleshooting of live networks efficient and effective. For more information, visit www.spirent.com.

[1] *OFLOPS: An Open Framework for OpenFlow Switch Evaluation, Charalampos Rotsos, Nadi Sarrar, Steve Uhlig, Rob Sherwood and Andrew W. Moore*

SPIRENT

1325 Borregas Avenue
Sunnyvale, CA 94089 USA

AMERICAS 1-800-SPIRENT | +1-818-676-2683 | sales@spirent.com

EUROPE AND THE MIDDLE EAST +44 (0) 1293 767979 | emeainfo@spirent.com

ASIA AND THE PACIFIC +86-10-8518-2539 | salesasia@spirent.com

Rev A. 06/14