OPEN NETWORKING
FOUNDATION

# Testing-Interop Working Group

*Interoperability Event Technical Issues*
*Report June 2013*
*Version 0.4*

*ONF TR-501*


Contact:

Michael Haugh, ONF Testing-Interop WG Chair, mhaugh@ixiacom.com

Ron Milford, ONF Testing-Interop WG Vice-Chair, rmilford@incntre.iu.edu

ONF Document Type: SDN Library
ONF Document Name: ONF PlugFest Technical Report - June 2013

## Disclaimer

## Contents



# 1 Introduction

The ONF held it's 3rd OpenFlow interoperability event June 4–8 2013 at the Indiana University InCNTRE SDN Lab in Indianapolis, Indiana. Please refer to the event whitepaper for descriptions of the event, tests and participants. This document's purpose is to present many of the issues encountered during the event. This is not a comprehensive list as not all issues were reported or investigated to the extent required to report them. We've attempted to present a detailed description of each issue wherever possible, resolutions or temporary workarounds used during the event to overcome those issues and recommendations for further action to resolve those issues. Due to the temporary nature of the testbed networks and time constraints, we were not able to fully debug and completely isolate the cause of all issues, but those issues are presented to increase awareness and further investigation in future events.

Vendor names were left out of the issue descriptions to help protect the confidentiality of the participants.
To further identify the relevance of issues, we've added the following information to each issue:

**Impact Level [low|med|high]**
**Impact Areas** – Areas of implementation or operations that will be impacted by this issue.
**Audience** – Who needs to pay attention to this issue including implementers and ONF working groups.

# 2 OpenFlow 1.0 Issues

A single testbed running OF 1.0 testcases was implemented. In parallel, there were 1.0 test frameworks running ad-hoc testing. Significant progress was made in respect to previous plugfests considering several new participants. There were no 1.0 specific issues reported. There were still several issues encountered that affected all versions and are detailed in the cross-version section of this document.

# 3 OpenFlow 1.2 Issues

Most vendors have chosen to bypass OpenFlow 1.2 and target OpenFlow 1.3 development. There were a very limited number of 1.2 capable switches and controllers available and most that supported 1.2 were able to run in 1.3 mode as well, so little testing was done with 1.2 only. Several tests were executed with combined 1.2 & 1.3 testbeds. There were no 1.2 specific issues reported. There were several issues that affect all versions and are detailed in the cross-version section of this document.

# 4 OpenFlow 1.3 Issues

For the first time a 1.3 testbed was added to the plugfest. The majority of participants participated in 1.3 interoperability testing. The main issues revolved around basic control channel establishment, multiple table configuration and multi-part messaging. The table configuration issues in particular should be reviewed, discussed and addressed by other working groups within the ONF. These issues consumed a significant amount of time to diagnose, resolve or work around. This limited the ability to progress on to the main test plans in many cases and pointed out the need for increased prior conformance-like testing or at least initial basic 1:1 testing at future events.

## 4.1 Control Channel Establishment

**Impact Level: High**
**Impact Areas: Implementation**
**Spec Relevance: 1.3**
**Audience: Implementers**
**Description:** In OF 1.3, the controller needs to discover a significant amount of additional information from the switch. Many controllers would attempt to discover this information from the switch after the first hello, but prior to establishment of the control channel. In some cases, the requests would come too quickly. Some switches would reply too slowly to large data informational/statistics requests and the

channel would not be established, or would be turned down. In other cases, all expected features would not be listed as supported, or an error would be received in response to a request, so the control channel would never be established. Multipart Request messages in particular, seemed to generate a significant number of errors from the switches.

**Spec Wording:**

**OpenFlow Spec v1.3.2**

**7.3.5 Multipart Messages** - *Multipart messages are used to encode requests or replies that potentially carry a large amount of data and would not always fit in a single OpenFlow message, which is limited to 64KB. The request or reply is encoded as a sequence of multipart messages with a specific multipart type, and re-assembled by the receiver. Multipart messages are primarily used to request statistics or state information from the switch.*

**Comments:** Openflow version 1.3.2 has many optional features, and implements special messages allowing the controller to gather detailed information about a device. The controller nevertheless needs to take into account the load generated by requesting this information. One could argue that the initial information exchange is part of the control channel setup. In some cases this information may be required for operational purposes. As a work around during the event, controllers manually configured device properties without trying to actively query the device. This issue could be hidden when the controller drops and reconnects the channel. The channel could possibly come up in a subsequent attempt.

**Recommendations:** Controllers should be more graceful in handling the information gathering process by spacing requests, increasing timeouts or retrying failed requests. Extensibility should evaluate if the spec should define additional steps in the control channel setup, including initial device information exchange and under what conditions the controller is allowed to drop the connection. Should also evaluate addition of maximum message rate, implementation of barrier messages for informational requests or mechanism for switch to respond with error message indicating resources are exhausted.

## 4.2 Multiple Table Instantiation

**Impact Level: High**

**Impact Areas:** Initial device configuration and reconfiguration of operational devices.

**Spec Relevance: 1.3**

**Audience: Extensibility, Config Management, Implementors**

**Description:** Currently, the 1.3 specification allows for the controller to instruct the switch on the configuration of its Flow Table Structure. While this may be relatively straightforward with a software based switch, it is extremely difficult to accomplish runtime configuration of table structures in hardware based switches.

**Spec Wording:**

**OpenFlow Spec v1.3.2**

**7.3.5.5.1 Table Features request and reply**

*If the request body contains an array of one or more ofp_table_features structs, the switch will attempt to change its flow tables to match the requested flow table*

*configuration. This operation configures the entire pipeline, and the set of flow tables in the pipeline must match the set in the request, or an error must be returned.*

**Comments:** A reboot may be required for some hardware configurations to modify the flow-table structure. Since this is a long term state and should not be changed often, it might be better configured using OF-Config. The ability to configure the table structure during run-time will greatly affect how conformance testing is performed.

**Recommendations:** The Extensibility Working Group should re-evaluate runtime configuration of table structures to see if it falls within their scope or if this is a feature that is reasonably required or if it should be made optional. The Config Management Working Group should consider methods for initial table configuration using OF-Config.

## 4.3 Multiple Table Support

**Impact Level: High**

**Impact Areas:** Features or use cases requiring multiple table support in hardware based devices.

**Spec Relevance: 1.3**

**Audience: Forwarding Abstraction, Extensibility, Config Management, Implementors**

**Description:** To ensure no loops are developed in jumping between tables. OpenFlow indicates a requirement of only jumping from lower table number to higher table number, which simplifies the logic and coding, but still allows for all possible combinations of jumps as long as it obeys this requirement. Again, it may be relatively straight forward to configure any possible combination the controller can dream up in a software based switch, but there are architecture limitations that vary significantly with hardware based switches.

The specification allows for the switch to report its current flow table pipeline setup, but there is no mechanism to report the complete range of possible configurations which could grow exponentially as the number of tables being used increases.

**Spec Wording:**

**OpenFlow Spec v1.3.2**

**7.3.5.5.1 Table Features request and reply**

*A flow entry can only direct a packet to a flow table number which is greater than its own flow table number, in other words pipeline processing can only go forward and not backward.*

**Comments:** This area has a lot of overlap with hardware constraints and the mapping of controller side application logic to the most appropriate switch configuration. Controllers need to be aware of constraints they may encounter. Until NDMs/TTPs are defined It might be useful to allow for manual configuration of parameters or information that can not be transferred via OpenFlow messages.

**Recommendations:** The Extensibility working group should reconsider direct configuration of table structures in hardware devices using the OF wire protocol. The FAWG has proposed that the set function of the table feature message should be disabled when using NDMs/TTPs. FAWG should consider addressing these issues in the TTP model currently being designed by proposing a limited number of well-known combinations / topologies / patterns of table hierarchy and features. The Config Management working group should consider TTP configuration methods using OF-Config.

# 5 OpenFlow Cross-Version Issues

## 5.1 Port Reporting

**Impact Level: High/Medium**

**Impact Areas:** This highly impacts monitoring, planning and reporting functions.

**Spec Version Relevance: All**

**Audience: Extensibility, Implementers**

**Description:** Disparity between switches in how they report OpenFlow ports. Some implementations report all OpenFlow physical ports regardless of status. Some only report OpenFlow Ports that are up. Some reported OpenFlow ports as up, but dropped all packets.

**Spec Wording:**

**OpenFlow Spec v1.0**

**5.3.1 Handshake** *The ports field is an array of ofp_phy_port structures that describe all the physical ports in the system that support OpenFlow.*

**OpenFlow Spec v1.3.2**

**7.3.5.7 Port Description** *The port description request OFPMP_PORT_DESCRIPTION enables the controller to get a description of all the ports in the system that support OpenFlow.*

**Comments:** The controller must deal with multiple behaviors of switches and has no way of knowing what behavior to expect. The 1.0 spec indicates that all physical ports that support OpenFlow should be reported. The 1.3 spec seems to remove the physical port statement which may indicate the intention was to encompasses both physical and logical ports.

**Recommendations:** The 1.3 spec should clarify requirements for port reporting. OpenFlow switches should report all OpenFlow enabled physical and existing OpenFlow enabled logical ports regardless of status allowing the controller to report available interfaces to applications and operators. This also allows controllers to track and report ports that may be down due to maintenance or outages.

## 5.2 VLAN based hybrids

**Impact Level: Low/Medium**

**Impact Areas:** VLAN based hybrid networks and network application that require the use of VLANs.

**Spec Relevance: All**

**Audience: Config Management, Extensibility, Implementers**

**Description:** VLAN based hybrid switches must know required vlans ahead of time and have them configured on all ports supporting openflow. This limits the use of VLANs by the controller. In addition there is some question about processing packets prior to entry of the OpenFlow domain on a VLAN based hybrid. Is a hybrid switch allowed to add or modify the VLAN tag of a packet in order to put the packet on the OF VLAN thereby entering the OF domain.

**Spec Wording:**

**OpenFlow Spec v1.3.2**

> **5.1 OpenFlow Header** *For example, a switch may use the VLAN tag or input port of the packet to decide whether to process the packet using one pipeline or the other, or it may direct all packets to the OpenFlow pipeline. This classification mechanism is outside the scope of this specification.*

**Comments:** VLAN configuration is outside the scope of the OF-Wire specification, but might be within the scope of OF-Config.

There are several schools of thought on how this should be dealt with.

1. The VLAN header of the packet MUST NOT be changed in any way outside of the OpenFlow process. It must be forwarded as it was received at the switch ingress port or upon output from a Flow Rule or Flow Table after the action set has been applied.

   In this case the port would be tagged with the OpenFlow enabled VLAN(s). The normal switch 802.1q process would send any packets entering the port with the correct VLAN tag(s) to the OpenFlow VLAN(s) to be handled by the OpenFlow process. Any packets arriving without a tag or with another non-OpenFlow VLAN ID would be sent to the normal L2 switch process.

   If this were strictly enforced, untagged packets could never enter the OF domain.

2. The switch would be allowed to add the OpenFlow enabled VLAN tag at the ingress/egress port prior to the packet entering the OF processing pipeline. The controller would not be aware of the OpenFlow enabled VLAN, VLAN restrictions on the switch or that a modification has been made to the incoming packet. The OpenFlow Enabled VLAN ID could be excluded in any packet_in to the controller.

   This would allow a host, non-OF capable device or OF device from another OF domain may send packets to the switch that would not be tagged. The port would be untagged in the OF VLAN. Untagged packets would have the correct OpenFlow VLAN ID added by the normal switch 802.1q process before being handed off to the OpenFlow process. The reverse process must occur on egress.

   The controller in this instance is not required to have any special knowledge of the device configurations, but would not be able to utilize VLANs in the network or match on any VLAN ID other than the OpenFlow enabled VLAN ID. If the switch supports QinQ, then other VLANs could be used and matched against, but the match would need to be made against the inner VLAN ID.

3. The switch would be allowed to add a VLAN tag at the ingress/egress port prior to the packet entering the OF processing pipeline. The controller would not be aware that a modification has been made to the incoming packet, but would be aware of all VLAN restrictions on the device.

   This would allow one or more VLANs in a single OF domain and allowed on an untagged port as in the previous example or a tagged port. If an untagged packet arrived on the tagged port it would be

sent to the normal L2 switch process. If untagged packets arriving on the tagged port were required to be part of the OF domain , then the default VLAN would also have to be an OF VLAN in which the default VLAN ID would be added by the normal switch 802.1q process before being handed off to the OpenFlow process which is the same behavior as the previous example.

**Recommendations:** There are too many possible scenarios to address in this document. The question of packet processing prior to entering the OF domain should go back to other ONF working Groups for further discussion and clarification. This type of question was originally under the purview of the Hybrid WG. Since that working group has disbanded, it is unclear which WG should deal with this issue.

The Config Management working groups should consider if configuration of VLANs within an OF Logical switch is within their particular scope.

Extensibility and Config Management should consider if either the OpenFlow Wire Protocol or OF-Config should allow for communication of VLAN implementation constraints to the controller if they exist.

## 5.3 VLAN Tag in packet_in

**Impact Level: High**
**Impact Areas:** Any network or network application that requires the use of VLANs
**Spec Relevance: All**
**Audience: Extensibility, Switch Implementers**
**Description:** Some switches added an internal VLAN tag to the packet_in header. We saw in previous events VLAN tags in packet_in were missing, or changed.
**Spec Wording:**

**OpenFlow Spec v1.3.2**

**7.4.1 Packet-In Message** *The data field contains the packet itself, or a fraction of the packet if the packet is buffered. The packet header reflect any changes applied to the packet in previous processing.*

**Comments:** It is important to maintain the integrity of the packet including the VLAN header.
**Recommendations:** The spec should clarify internal treatment of a packet header. The Header of the packet inside the packet_in MUST NOT be changed in any way. It must be forwarded as it was received at the switch ingress port or upon output from a Flow Rule or Flow Table after the action set has been applied. The spec should be modified to limit previous processing of the packet header to OpenFlow processing only. Special exception might be made for VLAN based hybrid switches (refer to VLAN based hybrids)

## 5.4 Topology Discovery

**Impact Level: High**
**Impact Areas: All**
**Spec Relevance: All**
**Audience: Architecture Working Group, Implementors**

**Description:** Controllers are still exhibiting issues with topology discovery logic such as:

- Deleting flows in switches
- Installing default rules to drop all or forward LLDP to the controller
- Sending LLDP (or other discovery packet) Packet_out on all up interfaces.

Hybrid switches often still do not turn off LLDP processing on OpenFlow ports by default. Spurious LLDP packets are being generated and sent to the controller, confusing the controller if the controller is using LLDP as a discovery mechanism. In other cases LLDP discovery packets are being sent to the normal L2 pipeline instead of the OpenFlow pipeline resulting in lost packets and failed discovery.

**Spec Wording:** *none*

**Comments:** Topology discovery is out of the scope of the specification, but it is extremely important to interoperability.

**Recommendations:** Architecture and Testing & Interop should create a Best Practices White Paper for topology discovery behavior.

## 5.5 Fail-Secure

**Impact Level: High**

**Impact Areas:** Initial switch startup and controller connection, recovery from loss of control channel connection.

**Spec Relevance: All**

**Audience: Extensibility, Controller Implementers**

**Description:** When a switch is operating in "fail secure" mode and the control channel session is lost, flows remain in the switch flow table as expected. When the control channel session is re-established, the controller does not query switch for flow rules. Some controllers flush the table upon reconnection without regard for previous flows left due to "fail secure" mode.

**Spec Wording:** There is no description of the "fail secure mode" or "fail standalone mode" recovery process in the 1.0 or 1.0.1 specifications.

**OpenFlow Spec v1.3.2**

**6.3.2 Connection Interruption** *In "fail secure mode", the only change to switch behavior is that packets and messages destined to the controllers are dropped. Flow entries should continue to expire according to their timeouts in "fail secure mode". In "fail standalone mode", the switch processes all packets using the OFPP_NORMAL reserved port; in other words, the switch acts as a legacy Ethernet switch or router. The "fail standalone mode" is usually only available on Hybrid switches (see 5.1).*

*Upon connecting to a controller again, the existing flow entries remain. The controller then has the option of deleting all flow entries, if desired.*

*The first time a switch starts up, it will operate in either "fail secure mode" or "fail standalone mode" mode, until it successfully connects to a controller. Configuration of*

*the default set of flow entries to be used at startup is outside the scope of the OpenFlow protocol.*

**Comments:** It appears that the intention of the spec was to allow for this behavior of deleting flows upon reconnections, but the order of the wording is very confusing and seems to be applied to "fail standalone" mode. While there may be use cases for flushing the table upon reconnection, this behavior defeats the purpose of "fail secure mode" and will cause interruptions in traffic.

**Recommendations:** Upon reconnection, controller implementations should be aware that a lost control channel means the device could be down, or that the control channel may have been interrupted, but traffic is still flowing as expected. Controllers should analyze the remaining flow table on the switch when reconnecting and not flush the flow-table by default. This would help minimize traffic interruption. Extensibility should clarify existing language in 1.3.2 spec to indicate flushing the table is not the only option. Extensibility to evaluate and recommend course of action to determine if recovery process should be defined in specification, white paper or just left to controller implementers to decide.

# 6 OF-Config Issues

None Reported

# 7 Test FrameWork Issues

None Reported

# 8 Additional Comments

## 8.1 Tools

The tools to troubleshoot 1.2 and above were still lacking during the event. Wireshark's ability to dissect 1.2 & 1.3 packets was still limited. In addition, there is presently no way to dissect multiple versions.

## 8.2 Issue Reporting

The June PlugFest was extremely useful to all who attended. Unfortunately, the reporting of issues was sporadic and inconsistent in detail levels. With 4 parallel testbeds, and with the large size of some of the testbeds, there were not enough observers to dedicate to maintaining the testbeds, troubleshooting and documenting the issues in each testbed. The Controller vendors running each test case were asked to document and share issues to include in the technical document, but this was not always done consistently. To help improve reporting, future events will include standard initial testing in each test bed and coinciding results forms.

# Appendix A: References

OpenFlow Switch Specification 1.0.0 - OpenFlow Switch Specification 1.0.0
OpenFlow Switch Errata v1.0.1 - Errata v1.0.1
OpenFlow Specification 1.2 - OpenFlow Switch Specification 1.2

OpenFlow Switch Specification 1.3.2 - OpenFlow Switch Specification 1.3.2
OpenFlow Configuration and Management Protocol - OpenFlow Configuration and Management Protocol 1.1 (OF-Config 1.1)
ONF-Testing-Interop-June-2013-Whitepaper -

## Appendix B: Revision History

| Version | Date | Notes |
|---|---|---|
| 0.1 | 8/1/2013 | Initial Draft – Indiana University, Ron Milford |
| 0.2 | 9/5/2013 | Impact and Audience additions, Changes based on T&I Call Review – Indiana University, Ron Milford |
| 0.3 | 9/27/2013 | Final Draft – Indiana University, Ron Milford |
| 0.4 | 10/22/2013 | Incorporate comments from Jean Tourrilhes & Curt Beckmann |