Internet Engineering Task Force (IETF)
Request for Comments: 7654
Category: Informational
ISSN: 2070-1721

S. Banks
VSS Monitoring
F. Calabria
Cisco Systems
G. Czirjak
R. Machat
Juniper Networks

October 2015

Benchmarking Methodology for In-Service Software Upgrade (ISSU)

Abstract

Modern forwarding devices attempt to minimize any control- and data-plane disruptions while performing planned software changes by implementing a technique commonly known as In-Service Software Upgrade (ISSU). This document specifies a set of common methodologies and procedures designed to characterize the overall behavior of a Device Under Test (DUT), subject to an ISSU event.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

As required by most Service Provider (SP) network operators, ISSU functionality has been implemented by modern forwarding devices to upgrade or downgrade from one software version to another with a goal of eliminating the downtime of the router and/or the outage of service. However, it is noted that while most operators desire complete elimination of downtime, minimization of downtime and service degradation is often the expectation.

The ISSU operation may apply in terms of an atomic version change of the entire system software or it may be applied in a more modular sense, such as for a patch or maintenance upgrade. The procedure described herein may be used to verify either approach, as may be supported by the vendor hardware and software.

In support of this document, the desired behavior for an ISSU operation can be summarized as follows:

- The software is successfully migrated from one version to a successive version or vice versa.

- There are no control-plane interruptions throughout the process. That is, the upgrade/downgrade could be accomplished while the device remains "in service". It is noted, however, that most service providers will still undertake such actions in a maintenance window (even in redundant environments) to minimize any risk.

- Interruptions to the forwarding plane are minimal to none.

- The total time to accomplish the upgrade is minimized, again to reduce potential network outage exposure (e.g., an external failure event might impact the network as it operates with reduced redundancy).

This document provides a set of procedures to characterize a given forwarding device’s ISSU behavior quantitatively, from the perspective of meeting the above expectations.

Different hardware configurations may be expected to be benchmarked, but a typical configuration for a forwarding device that supports ISSU consists of at least one pair of Routing Processors (RPs) that operate in a redundant fashion, and single or multiple forwarding engines (line cards) that may or may not be redundant, as well as fabric cards or other components as applicable. This does not preclude the possibility that a device in question can perform ISSU functions through the operation of independent process components, which may be upgraded without impact to the overall operation of the device. As an example, perhaps the software module involved in SNMP functions can be upgraded without impacting other operations.

The concept of a multi-chassis deployment may also be characterized by the current set of proposed methodologies, but the implementation-specific details (i.e., process placement and others) are beyond the scope of the current document.

Since most modern forwarding devices, where ISSU would be applicable, do consist of redundant RPs and hardware-separated control-plane and data-plane functionality, this document will focus on methodologies that would be directly applicable to those platforms. It is anticipated that the concepts and approaches described herein may be readily extended to accommodate other device architectures as well.

2. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lowercase uses of these words are not to be interpreted as carrying the significance of RFC 2119.

3. Generic ISSU Process, Phased Approach

ISSU may be viewed as the behavior of a device when exposed to a planned change in its software functionality. This may mean changes to the core operating system, separate processes or daemons, or even firmware logic in programmable hardware devices (e.g., Complex Programmable Logic Device (CPLD) or Field-Programmable Gate Array (FPGA)). The goal of an ISSU implementation is to permit such actions with minimal or no disruption to the primary operation of the device in question.

ISSU may be user initiated through direct interaction with the device or activated through some automated process on a management system or even on the device itself. For the purposes of this document, we will focus on the model where the ISSU action is initiated by direct user intervention.

The ISSU process can be viewed as a series of different phases or activities, as defined below. For each of these phases, the test operator must record the outcome as well as any relevant observations (defined further in the present document). Note that, a given vendor implementation may or may not permit the abortion of the in-progress

ISSU at particular stages. There may also be certain restrictions as to ISSU availability given certain functional configurations (for example, ISSU in the presence of Bidirectional Failure Detection (BFD) [RFC5880] may not be supported). It is incumbent upon the test operator to ensure that the DUT is appropriately configured to provide the appropriate test environment. As with any properly orchestrated test effort, the test plan document should reflect these and other relevant details and should be written with close attention to the expected production operating environment. The combined analysis of the results of each phase will characterize the overall ISSU process with the main goal of being able to identify and quantify any disruption in service (from the data- and control-plane perspective) allowing operators to plan their maintenance activities with greater precision.

3.1. Software Download

In this first phase, the requested software package may be downloaded to the router and is typically stored onto a device. The downloading of software may be performed automatically by the device as part of the upgrade process, or it may be initiated separately. Such separation allows an administrator to download the new code inside or outside of a maintenance window; it is anticipated that downloading new code and saving it to disk on the router will not impact operations. In the case where the software can be downloaded outside of the actual upgrade process, the administrator should do so; downloading software can skew timing results based on factors that are often not comparative in nature. Internal compatibility verification may be performed by the software running on the DUT, to verify the checksum of the files downloaded as well as any other pertinent checks. Depending upon vendor implementation, these mechanisms may include 1) verifying that the downloaded module(s) meet a set of identified prerequisites such as (but not limited to) hardware or firmware compatibility or minimum software requirements or even 2) ensuring that device is "authorized" to run the target software.

Where such mechanisms are made available by the product, they should be verified, by the tester, with the goal of avoiding operational issues in production. Verification should include both positive verification (ensuring that an ISSU action should be permitted) as well as negative tests (creation of scenarios where the verification mechanisms would report exceptions).

3.2. Software Staging

In this second phase, the requested software package is loaded in the pertinent components of a given forwarding device (typically the RP in standby state). Internal compatibility verification may be performed by the software running on the DUT, as part of the upgrade process itself, to verify the checksum of the files downloaded as well as any other pertinent checks. Depending upon vendor implementation, these mechanisms may include verification that the downloaded module(s) meet a set of identified prerequisites such as hardware or firmware compatibility or minimum software requirements. Where such mechanisms are made available by the product, they should be verified, by the tester (again with the goal of avoiding operational issues in production). In this case, the execution of these checks is within the scope of the upgrade time and should be included in the testing results. Once the new software is downloaded to the pertinent components of the DUT, the upgrade begins, and the DUT begins to prepare itself for upgrade. Depending on the vendor implementation, it is expected that redundant hardware pieces within the DUT are upgraded, including the backup or secondary RP.

3.3. Upgrade Run

In this phase, a switchover of RPs may take place, where one RP is now upgraded with the new version of software. More importantly, the "Upgrade Run" phase is where the internal changes made to information and state (stored on the router, on disk, and in memory) are either migrated to the "new" version of code, or transformed/rebuilt to meet the standards of the new version of code, and pushed onto the appropriate pieces of hardware. It is within this phase that any outage(s) on the control or forwarding plane may be expected to be observed. This is the critical phase of the ISSU, where the control plane should not be impacted and any interruptions to the forwarding plane should be minimal to none.

If any control- or data-plane interruptions are observed within this stage, they should be recorded as part of the results document.

For some implementations, the two stages, as described in Section 3.2 and above, may be concatenated into one monolithic operation. In that case, the calculation of the respective ISSU time intervals may need to be adapted accordingly.

3.4. Upgrade Acceptance

In this phase, the new version of software must be running in all the physical nodes of the logical forwarding device (RPs and line cards as applicable). At this point, configuration control is returned to the operator, and normal device operation, i.e., outside of ISSU-oriented operation, is resumed.

4. Test Methodology

As stated by [RFC6815], the Test Topology Setup must be part of an Isolated Test Environment (ITE).

The reporting of results must take into account the repeatability considerations from Section 4 of [RFC2544]. It is RECOMMENDED to perform multiple trials and report average results. The results are reported in a simple statement including the measured frame loss and ISSU impact times.

4.1. Test Topology

The hardware configuration of the DUT (Device Under Test) should be identical to the one expected to be or currently deployed in production in order for the benchmark to have relevance. This would include the number of RPs, hardware version, memory, and initial software release, any common chassis components, such as fabric hardware in the case of a fabric-switching platform, and the specific line cards (version, memory, interfaces type, rate, etc.).

For the control and data plane, differing configuration approaches may be utilized. The recommended approach relies on "mimicking" the existing production data- and control-plane information, in order to emulate all the necessary Layer 1 through Layer 3 communications and, if appropriate, the upper-layer characteristics of the network, as well as end-to-end traffic/communication pairs. In other words, design a representative load model of the production environment and deploy a collapsed topology utilizing test tools and/or external devices, where the DUT will be tested. Note that, the negative impact of ISSU operations is likely to impact scaled, dynamic topologies to a greater extent than simpler, static environments. As such, this methodology (based upon production configuration) is advised for most test scenarios.

The second, more simplistic approach is to deploy an ITE in which endpoints are "directly" connected to the DUT. In this manner, control-plane information is kept to a minimum (only connected interfaces), and only a basic data-plane of sources and destinations is applied. If this methodology is selected, care must be taken to understand that the systemic behavior of the ITE may not be identical to that experienced by a device in a production network role. That is, control-plane validation may be minimal to none with this methodology. Consequently, if this approach is chosen, comparison with at least one production configuration is recommended in order to understand the direct relevance and limitations of the test exercise.

4.2. Load Model

In consideration of the defined test topology, a load model must be developed to exercise the DUT while the ISSU event is introduced. This applied load should be defined in such a manner as to provide a granular, repeatable verification of the ISSU impact on transit traffic. Sufficient traffic load (rate) should be applied to permit timing extrapolations at a minimum granularity of 100 milliseconds, e.g., 100 Mbps for a 10 Gbps interface. The use of steady traffic streams rather than bursty loads is preferred to simplify analysis.

The traffic should be patterned to provide a broad range of source and destination pairs, which resolve to a variety of FIB (Forwarding Information Base) prefix lengths. If the production network environment includes multicast traffic or VPNs (L2, L3, or IPsec), it is critical to include these in the model.

For mixed protocol environments (e.g., IPv4 and IPv6), frames should be distributed between the different protocols. The distribution should approximate the network conditions of deployment. In all cases, the details of the mixed protocol distribution must be included in the reporting.

The feature, protocol timing, and other relevant configurations should be matched to the expected production environment. Deviations from the production templates may be deemed necessary by the test operator (for example, certain features may not support ISSU or the test bed may not be able to accommodate such). However, the impact of any such divergence should be clearly understood, and the differences must be recorded in the results documentation. It is recommended that a Network Management System (NMS) be deployed, preferably similar to that utilized in production. This will allow for monitoring of the DUT while it is being tested, both in terms of supporting the impact analysis on system resources as well as detecting interference with non-transit (management) traffic as a result of the ISSU operation. It is suggested that the actual test exercise be managed utilizing direct console access to the DUT, if at all possible, to avoid the possibility that a network interruption impairs execution of the test exercise.

All in all, the load model should attempt to simulate the production network environment to the greatest extent possible in order to maximize the applicability of the results generated.

5. ISSU Test Methodology

As previously described, for the purposes of this test document, the ISSU process is divided into three main phases. The following methodology assumes that a suitable test topology has been constructed per Section 4. A description of the methodology to be applied for each of the above phases follows.

5.1. Pre-ISSU Recommended Verifications

The steps of this phase are as follows.

1. Verify that enough hardware and software resources are available to complete the Load operation (e.g., enough disk space).

2. Verify that the redundancy states between RPs and other nodes are as expected (e.g., redundancy on, RPs synchronized).

3. Verify that the device, if running protocols capable of NSR (Non-Stop Routing), is in a "ready" state; that is, that the sync between RPs is complete and the system is ready for failover, if necessary.

4. Gather a configuration snapshot of the device and all of its applicable components.

5. Verify that the node is operating in a "steady" state (that is, no critical or maintenance function is being currently performed).

6. Note any other operational characteristics that the tester may deem applicable to the specific implementation deployed.

5.2. Software Staging

The steps of this phase are as follows.

1. Establish all relevant protocol adjacencies and stabilize routing within the test topology. In particular, ensure that the scaled levels of the dynamic protocols are dimensioned as specified by the test topology plan.

2. Clear, relevant logs and interface counters to simplify analysis. If possible, set logging timestamps to a highly granular mode. If the topology includes management systems, ensure that the appropriate polling levels have been applied, sessions have been established, and the responses are per expectation.

3. Apply the traffic loads as specified in the load model previously developed for this exercise.

4. Document an operational baseline for the test bed with relevant data supporting the above steps (include all relevant load characteristics of interest in the topology, e.g., routing load, traffic volumes, memory and CPU utilization).

5. Note the start time (T0) and begin the code change process utilizing the appropriate mechanisms as expected to be used in production (e.g., active download with TFTP, FTP, SCP, etc., or direct install from local or external storage facility). In order to ensure that ISSU process timings are not skewed by the lack of a network-wide synchronization source, the use of a network NTP source is encouraged.

6. Take note of any logging information and command-line interface (CLI) prompts as needed. (This detail will be vendor specific.) Respond to any DUT prompts in a timely manner.

7. Monitor the DUT for the reload of the secondary RP to the new software level. Once the secondary has stabilized on the new code, note the completion time. The duration of these steps will be recorded as "T1".

8. Review system logs for any anomalies, check that relevant dynamic protocols have remained stable, and note traffic loss if any. Verify that deployed management systems have not identified any unexpected behavior.

5.3. Upgrade Run

The following assumes that the software load step and upgrade step are discretely controllable. If not, maintain the aforementioned timer and monitor for completion of the ISSU as described below.

1. Note the start time and initiate the actual upgrade procedure.

2. Monitor the operation of the secondary route processor while it initializes with the new software and assumes mastership of the DUT. At this point, pay particular attention to any indications of control-plane disruption, traffic impact, or other anomalous

behavior. Once the DUT has converged upon the new code and returned to normal operation, note the completion time and log the duration of this step as "T2".

3. Review the syslog data in the DUT and neighboring devices for any behavior that would be disruptive in a production environment (line card reloads, control-plane flaps, etc.). Examine the traffic generators for any indication of traffic loss over this interval. If the Test Set reported any traffic loss, note the number of frames lost as "TPL_frames", where TPL stands for "Total Packet Loss". If the Test Set also provides outage duration, note this as "TPL_time". (Alternatively, TPL_time may be calculated as (TPL / Offered Load) * 1000. The units for Offered Load are packets per second; the units for TPL_time are milliseconds.)

4. Verify the DUT status observations as per any NMS managing the DUT and its neighboring devices. Document the observed CPU and memory statistics both during and after the ISSU upgrade event, and ensure that memory and CPU have returned to an expected (previously baselined) level.

5.4. Post-ISSU Verification

The following describes a set of post-ISSU verification tasks that are not directly part of the ISSU process, but are recommended for execution in order to validate a successful upgrade.

1. Configuration delta analysis

   Examine the post-ISSU configurations to determine if any changes have occurred either through process error or due to differences in the implementation of the upgraded code.

2. Exhaustive control-plane analysis

   Review the details of the Routing Information Base (RIB) and FIB to assess whether any unexpected changes have been introduced in the forwarding paths.

3. Verify that both RPs are up and that the redundancy mechanism for the control plane is enabled and fully synchronized.

4. Verify that no control-plane (protocol) events or flaps were detected.

5. Verify that no L1 and or L2 interface flaps were observed.

6. Document the hitless operation or presence of an outage based upon the counter values provided by the Test Set.

5.5. ISSU under Negative Stimuli

As an OPTIONAL Test Case, the operator may want to perform an ISSU test while the DUT is under stress by introducing route churn to any or all of the involved phases of the ISSU process.

One approach relies on the operator to gather statistical information from the production environment and determine a specific number of routes to flap every 'fixed' or 'variable' interval. Alternatively, the operator may wish to simply preselect a fixed number of prefixes to flap. As an example, an operator may decide to flap 1% of all the BGP routes every minute and restore them 1 minute afterwards. The tester may wish to apply this negative stimulus throughout the entire ISSU process or, most importantly, during the run phase. It is important to ensure that these routes, which are introduced solely for stress proposes, must not overlap the ones (per the load model) specifically leveraged to calculate the TPL_time (recorded outage). Furthermore, there should not be 'operator-induced' control-plane protocol adjacency flaps for the duration of the test process as it may adversely affect the characterization of the entire test exercise. For example, triggering IGP adjacency events may force recomputation of underlying routing tables with attendant impact to the perceived ISSU timings. While not recommended, if such trigger events are desired by the test operator, care should be taken to avoid the introduction of unexpected anomalies within the test harness.

6. ISSU Abort and Rollback

Where a vendor provides such support, the ISSU process could be aborted for any reason by the operator. However, the end results and behavior may depend on the specific phase where the process was aborted. While this is implementation dependent, as a general recommendation, if the process is aborted during the "Software Download" or "Software Staging" phases, no impact to service or device functionality should be observed. In contrast, if the process is aborted during the "Upgrade Run" or "Upgrade Accept" phases, the system may reload and revert back to the previous software release, and, as such, this operation may be service affecting. Where vendor support is available, the abort/rollback functionality should be verified, and the impact, if any, quantified generally following the procedures provided above.

7. Final Report: Data Presentation and Analysis

All ISSU impact results are summarized in a simple statement describing the "ISSU Disruption Impact" including the measured frame loss and impact time, where impact time is defined as the time frame determined per the TPL_time reported outage. These are considered to be the primary data points of interest.

However, the entire ISSU operational impact should also be considered in support of planning for maintenance, and, as such, additional reporting points are included.

Software download / secondary update	T1
Upgrade/Run	T2
ISSU Traffic Disruption (Frame Loss)	TPL_frames
ISSU Traffic Impact Time (milliseconds)	TPL_time
ISSU Housekeeping Interval	T3
(Time for both RPs up on new code and fully synced - Redundancy restored)	
Total ISSU Maintenance Window	T4 (sum of T1+T2+T3)

The results reporting must provide the following information:

- DUT hardware and software detail

- Test Topology definition and diagram (especially as related to the ISSU operation)

- Load Model description including protocol mixes and any divergence from the production environment

- Time Results as per above

- Anomalies Observed during ISSU

- Anomalies Observed in post-ISSU analysis

It is RECOMMENDED that the following parameters be reported as outlined below:

Parameter	Units or Examples
Traffic Load	Frames per second and bits per second
Disruption (average)	Frames
Impact Time (average)	Milliseconds
Number of trials	Integer count
Protocols	IPv4, IPv6, MPLS, etc.
Frame Size	Octets
Port Media	Ethernet, Gigabit Ethernet (GbE), Packet over SONET (POS), etc.
Port Speed	10 Gbps, 1 Gbps, 100 Mbps, etc.
Interface Encaps	Ethernet, Ethernet VLAN, PPP, High-Level Data Link Control (HDLC), etc.
Number of Prefixes	Integer count
flapped (ON Interval)	(Optional) # of prefixes / Time (min.)
flapped (OFF Interval)	(Optional) # of prefixes / Time (min.)

Document any configuration deltas that are observed after the ISSU upgrade has taken effect. Note differences that are driven by changes in the patch or release level, as well as items that are aberrant changes due to software faults. In either of these cases, any unexpected behavioral changes should be analyzed and a determination made as to the impact of the change (be it functional variances or operational impacts to existing scripts or management mechanisms).

7.1. Data Collection Considerations

When a DUT is undergoing an ISSU operation, it’s worth noting that the DUT’s data collection and reporting of data, such as counters, interface statistics, log messages, etc., may not be accurate. As such, one should not rely on the DUT’s data collection methods, but rather, should use the test tools and equipment to collect data used for reporting in Section 7. Care and consideration should be paid in testing or adding new test cases, such that the desired data can be collected from the test tools themselves, or other external equipment, outside of the DUT itself.

8. Security Considerations

All BMWG memos are limited to testing in a laboratory Isolated Test Environment (ITE), thus avoiding accidental interruption to production networks due to test activities.

All benchmarking activities are limited to technology characterization using controlled stimuli in a laboratory environment with dedicated address space and the other constraints [RFC2544].

The benchmarking network topology will be an independent test setup and MUST NOT be connected to devices that may forward the test traffic into a production network or misroute traffic to the test management network.

Further, benchmarking is performed on a "black-box" basis, relying solely on measurements observable external to the Device Under Test / System Under Test (DUT/SUT).

Special capabilities should not exist in the DUT/SUT specifically for benchmarking purposes. Any implications for network security arising from the DUT/SUT should be identical in the lab and in production networks.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2544] Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, DOI 10.17487/RFC2544, March 1999, <http://www.rfc-editor.org/info/rfc2544>.

9.2. Informative References

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <http://www.rfc-editor.org/info/rfc5880>.

[RFC6815] Bradner, S., Dubray, K., McQuaid, J., and A. Morton, "Applicability Statement for RFC 2544: Use on Production Networks Considered Harmful", RFC 6815, DOI 10.17487/RFC6815, November 2012, <http://www.rfc-editor.org/info/rfc6815>.

Acknowledgments

The authors wish to thank Vibin Thomas for his valued review and feedback.

Authors' Addresses

Sarah Banks
VSS Monitoring
Email: sbanks@encrypted.net

Fernando Calabria
Cisco Systems
Email: fcalabri@cisco.com

Gery Czirjak
Juniper Networks
Email: gczirjak@juniper.net

Ramdas Machat
Juniper Networks
Email: rmachat@juniper.net