

Internet Engineering Task Force (IETF)  
Request for Comments: 8456  
Category: Informational  
ISSN: 2070-1721

V. Bhuvaneshwaran  
A. Basil  
Veryx Technologies  
M. Tassinari  
Hewlett Packard Enterprise  
V. Manral  
NanoSec  
S. Banks  
VSS Monitoring  
October 2018

## Benchmarking Methodology for Software-Defined Networking (SDN) Controller Performance

### Abstract

This document defines methodologies for benchmarking the control-plane performance of Software-Defined Networking (SDN) Controllers. The SDN Controller is a core component in the SDN architecture that controls the behavior of the network. SDN Controllers have been implemented with many varying designs in order to achieve their intended network functionality. Hence, the authors of this document have taken the approach of considering an SDN Controller to be a black box, defining the methodology in a manner that is agnostic to protocols and network services supported by controllers. This document provides a method for measuring the performance of all controller implementations.

### Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <https://www.rfc-editor.org/info/rfc8456>.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction .....	4
1.1. Conventions Used in This Document .....	4
2. Scope .....	4
3. Test Setup .....	4
3.1. Test Setup - Controller Operating in Standalone Mode .....	5
3.2. Test Setup - Controller Operating in Cluster Mode .....	6
4. Test Considerations .....	7
4.1. Network Topology .....	7
4.2. Test Traffic .....	7
4.3. Test Emulator Requirements .....	7
4.4. Connection Setup .....	8
4.5. Measurement Point Specification and Recommendation .....	9
4.6. Connectivity Recommendation .....	9
4.7. Test Repeatability .....	9
4.8. Test Reporting .....	9
5. Benchmarking Tests .....	11
5.1. Performance .....	11
5.1.1. Network Topology Discovery Time .....	11
5.1.2. Asynchronous Message Processing Time .....	13
5.1.3. Asynchronous Message Processing Rate .....	14
5.1.4. Reactive Path Provisioning Time .....	17
5.1.5. Proactive Path Provisioning Time .....	19
5.1.6. Reactive Path Provisioning Rate .....	21
5.1.7. Proactive Path Provisioning Rate .....	23
5.1.8. Network Topology Change Detection Time .....	25
5.2. Scalability .....	26
5.2.1. Control Sessions Capacity .....	26
5.2.2. Network Discovery Size .....	27
5.2.3. Forwarding Table Capacity .....	29

5.3. Security .....	31
5.3.1. Exception Handling .....	31
5.3.2. Handling Denial-of-Service Attacks .....	32
5.4. Reliability .....	34
5.4.1. Controller Failover Time .....	34
5.4.2. Network Re-provisioning Time .....	36
6. IANA Considerations .....	37
7. Security Considerations .....	38
8. References .....	38
8.1. Normative References .....	38
8.2. Informative References .....	38
Appendix A. Benchmarking Methodology Using OpenFlow Controllers ...	39
A.1. Protocol Overview .....	39
A.2. Messages Overview .....	39
A.3. Connection Overview .....	39
A.4. Performance Benchmarking Tests .....	40
A.4.1. Network Topology Discovery Time .....	40
A.4.2. Asynchronous Message Processing Time .....	42
A.4.3. Asynchronous Message Processing Rate .....	43
A.4.4. Reactive Path Provisioning Time .....	44
A.4.5. Proactive Path Provisioning Time .....	46
A.4.6. Reactive Path Provisioning Rate .....	47
A.4.7. Proactive Path Provisioning Rate .....	49
A.4.8. Network Topology Change Detection Time .....	50
A.5. Scalability .....	51
A.5.1. Control Sessions Capacity .....	51
A.5.2. Network Discovery Size .....	52
A.5.3. Forwarding Table Capacity .....	54
A.6. Security .....	55
A.6.1. Exception Handling .....	55
A.6.2. Handling Denial-of-Service Attacks .....	57
A.7. Reliability .....	59
A.7.1. Controller Failover Time .....	59
A.7.2. Network Re-provisioning Time .....	61
Acknowledgments .....	63
Authors' Addresses .....	64

## 1. Introduction

This document provides generic methodologies for benchmarking Software-Defined Networking (SDN) Controller performance. To achieve the desired functionality, an SDN Controller may support many northbound and southbound protocols, implement a wide range of applications, and work either alone or as part of a group. This document considers an SDN Controller to be a black box, regardless of design and implementation. The tests defined in this document can be used to benchmark an SDN Controller for performance, scalability, reliability, and security, independently of northbound and southbound protocols. Terminology related to benchmarking SDN Controllers is described in the companion terminology document [RFC8455]. These tests can be performed on an SDN Controller running as a virtual machine (VM) instance or on a bare metal server. This document is intended for those who want to measure an SDN Controller's performance as well as compare the performance of various SDN Controllers.

### 1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. Scope

This document defines a methodology for measuring the networking metrics of SDN Controllers. For the purpose of this memo, the SDN Controller is a function that manages and controls Network Devices. Any SDN Controller without a control capability is out of scope for this memo. The tests defined in this document enable the benchmarking of SDN Controllers in two ways: standalone mode (a standalone controller) and cluster mode (a cluster of homogeneous controllers). These tests are recommended for execution in lab environments rather than in live network deployments. Performance benchmarking of a federation of controllers (i.e., a set of SDN Controllers) managing different domains, is beyond the scope of this document.

## 3. Test Setup

As noted above, the tests defined in this document enable the measurement of an SDN Controller's performance in standalone mode and cluster mode. This section defines common reference topologies that are referred to in individual tests described later in this document.

## 3.1. Test Setup - Controller Operating in Standalone Mode

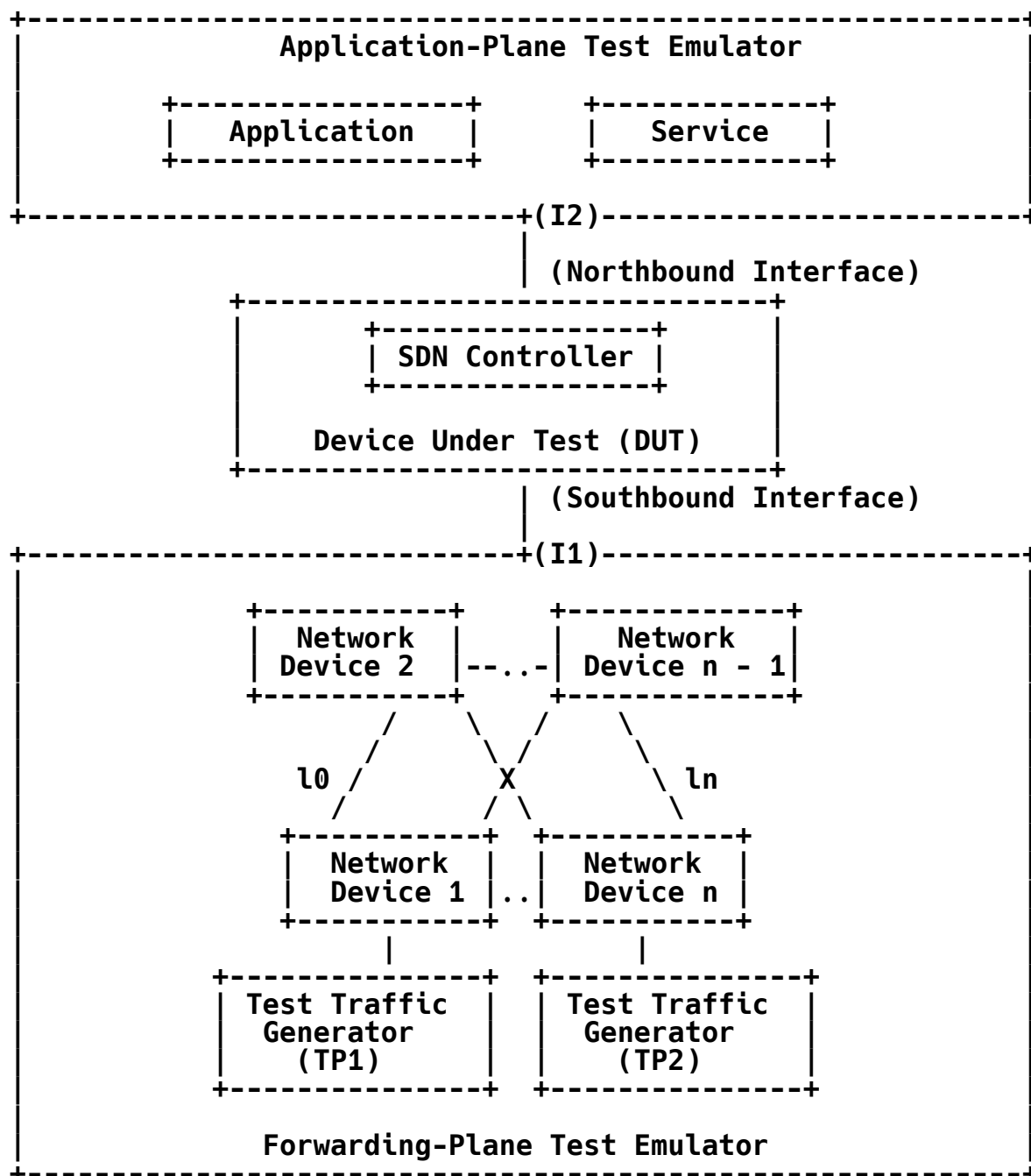


Figure 1

### 3.2. Test Setup - Controller Operating in Cluster Mode

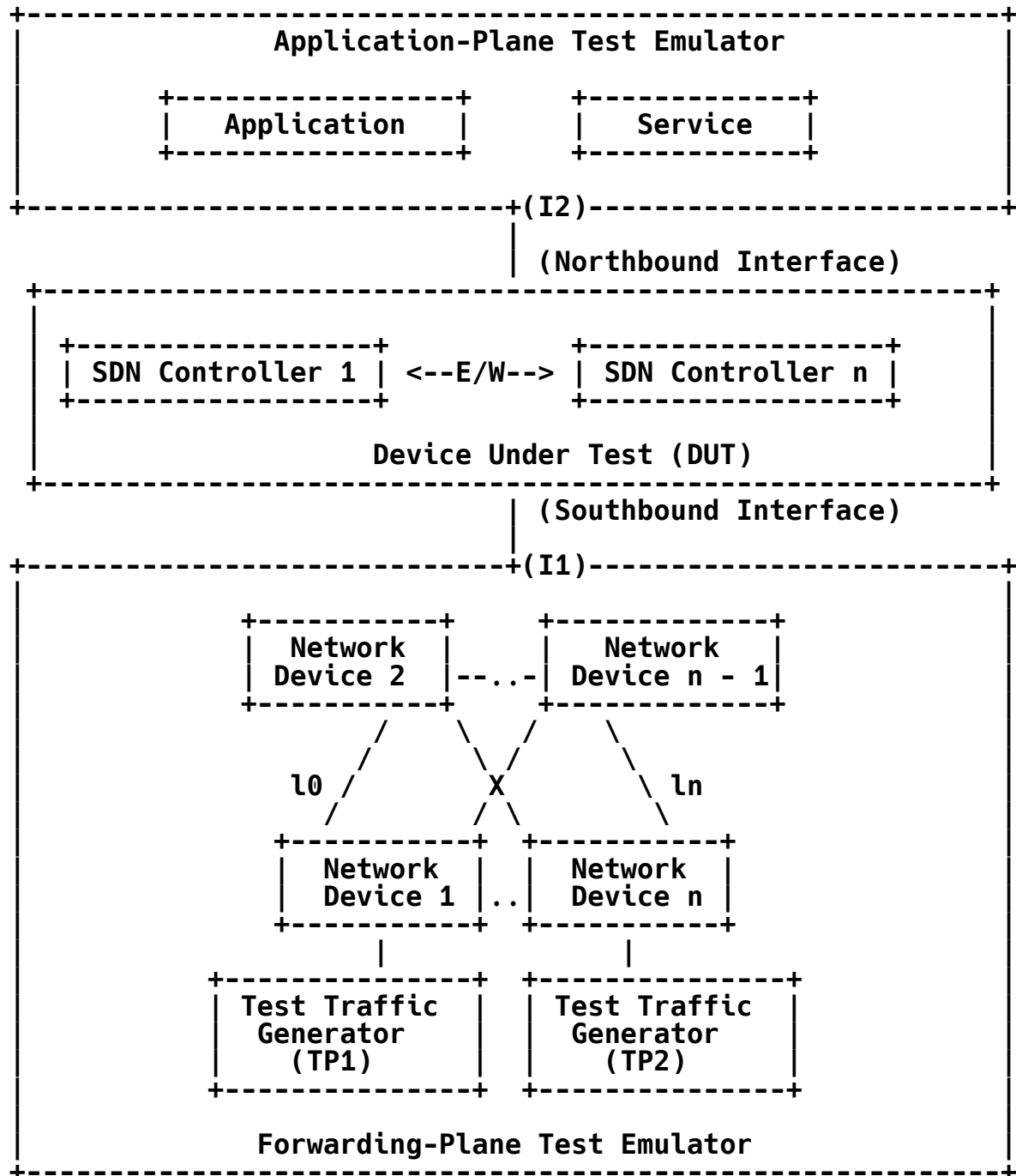


Figure 2

## 4. Test Considerations

### 4.1. Network Topology

The test cases **SHOULD** use Leaf-Spine topology with at least two Network Devices in the topology for benchmarking. Test traffic generators TP1 and TP2 **SHOULD** be connected to the leaf Network Device 1 and the leaf Network Device n. To achieve a complete performance characterization of the SDN Controller, it is recommended that the controller be benchmarked for many network topologies and a varying number of Network Devices. Further, care should be taken to make sure that a loop-prevention mechanism is enabled in either the SDN Controller or the network when the topology contains redundant network paths.

### 4.2. Test Traffic

Test traffic is used to notify the controller about the asynchronous arrival of new flows. The test cases **SHOULD** use frame sizes of 128, 512, and 1508 bytes for benchmarking. Tests using jumbo frames are optional.

### 4.3. Test Emulator Requirements

The test emulator **SHOULD** timestamp the transmitted and received control messages to/from the controller on the established network connections. The test cases use these values to compute the controller processing time.

#### 4.4. Connection Setup

There may be controller implementations that support unencrypted and encrypted network connections with Network Devices. Further, the controller may be backward compatible with Network Devices running older versions of southbound protocols. It may be useful to measure the controller's performance with one or more applicable connection setup methods defined below. For cases with encrypted communications between the controller and the switch, key management and key exchange **MUST** take place before any performance or benchmark measurements.

1. Unencrypted connection with Network Devices, running the same protocol version.
2. Unencrypted connection with Network Devices, running different protocol versions.

Examples:

- a. Controller running current protocol version and switch running older protocol version.
  - b. Controller running older protocol version and switch running current protocol version.
3. Encrypted connection with Network Devices, running the same protocol version.
  4. Encrypted connection with Network Devices, running different protocol versions.

Examples:

- a. Controller running current protocol version and switch running older protocol version.
- b. Controller running older protocol version and switch running current protocol version.



#### 4.5. Measurement Point Specification and Recommendation

The accuracy of the measurements depends on several factors, including the point of observation where the indications are captured. For example, the notification can be observed at the controller or test emulator. The test operator **SHOULD** make the observations/measurements at the interfaces of the test emulator, unless explicitly specified otherwise in the individual test. In any case, the locations of measurement points **MUST** be reported.

#### 4.6. Connectivity Recommendation

The SDN Controller in the test setup **SHOULD** be connected directly with the forwarding-plane and management-plane test emulators to avoid any delays or failure introduced by the intermediate devices during benchmarking tests. When the controller is implemented as a virtual machine, details of the physical and logical connectivity **MUST** be reported.

#### 4.7. Test Repeatability

To increase confidence in the measured results, it is recommended that each test **SHOULD** be repeated a minimum of 10 times.

#### 4.8. Test Reporting

Each test has a reporting format that contains some global and identical reporting components, and some individual components that are specific to individual tests. The following parameters for test configuration and controller settings **MUST** be reflected in the test report.

Test Configuration Parameters:

1. Controller name and version
2. Northbound protocols and versions
3. Southbound protocols and versions
4. Controller redundancy mode (standalone or cluster mode)
5. Connection setup (unencrypted or encrypted)
6. Network Device type (physical, virtual, or emulated)
7. Number of nodes

8. Number of links
9. Data-plane test traffic type
10. Controller system configuration (e.g., physical or virtual machine, CPU, memory, caches, operating system, interface speed, storage)
11. Reference test setup (e.g., the setup shown in Section 3.1)

**Parameters for Controller Settings:**

1. Topology rediscovery timeout
2. Controller redundancy mode (e.g., active-standby)
3. Controller state persistence enabled/disabled

To ensure the repeatability of the test, the following capabilities of the test emulator **SHOULD** be reported:

1. Maximum number of Network Devices that the forwarding plane emulates
2. Control message processing time (e.g., topology discovery messages)

One way to determine the above two values is to simulate the required control sessions and messages from the control plane.

## 5. Benchmarking Tests

### 5.1. Performance

#### 5.1.1. Network Topology Discovery Time

##### Objective:

Measure the time taken by the controller(s) to determine the complete network topology, defined as the interval starting with the first discovery message from the controller(s) at its southbound interface and ending with all features of the static topology determined.

##### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

##### Prerequisites:

1. The controller MUST support network discovery.
2. The tester should be able to retrieve the discovered topology information through either the controller's management interface or northbound interface to determine if the discovery was successful and complete.
3. Ensure that the controller's topology rediscovery timeout has been set to the maximum value, to avoid initiation of the rediscovery process in the middle of the test.

##### Procedure:

1. Ensure that the controller is operational and that its network applications, northbound interface, and southbound interface are up and running.
2. Establish the network connections between the controller and the Network Devices.
3. Record the time for the first discovery message ( $T_{m1}$ ) received from the controller at the forwarding-plane test emulator interface ( $I1$ ).

4. Query the controller every  $t$  seconds (the RECOMMENDED value for  $t$  is 3) to obtain the discovered network topology information through the northbound interface or the management interface, and compare it with the deployed network topology information.
5. Stop the trial when the discovered topology information matches the deployed network topology or when the discovered topology information returns the same details for three consecutive queries.
6. Record the time for the last discovery message ( $T_{mn}$ ) sent to the controller from the forwarding-plane test emulator interface ( $I1$ ) when the trial completes successfully (e.g., when the topology matches).

#### Measurements:

Topology Discovery Time ( $DT1$ ) =  $T_{mn} - T_{m1}$

Average Topology Discovery Time ( $TD_m$ ) = 
$$\frac{DT1 + DT2 + DT3 \dots DTn}{\text{Total Trials}}$$

Topology Discovery Time Variance ( $TD_v$ ) = 
$$\frac{\text{SUM}[\text{SQUAREOF}(DT_i - TD_m)]}{\text{Total Trials} - 1}$$

#### Reporting Format:

The Topology Discovery Time results MUST be reported in tabular format, with a row for each successful iteration. The last row of the table indicates the Topology Discovery Time variance, and the previous row indicates the Average Topology Discovery Time.

If this test is repeated with a varying number of nodes over the same topology, the results SHOULD be reported in the form of a graph. The X coordinate SHOULD be the number of nodes ( $N$ ), and the Y coordinate SHOULD be the Average Topology Discovery Time.

### 5.1.2. Asynchronous Message Processing Time

#### Objective:

Measure the time taken by the controller(s) to process an asynchronous message, defined as the interval starting with an asynchronous message from a Network Device after the discovery of all the devices by the controller(s) and ending with a response message from the controller(s) at its southbound interface.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisite:

The controller MUST have successfully completed the network topology discovery for the connected Network Devices.

#### Procedure:

1. Generate asynchronous messages from every connected Network Device to the SDN Controller, one at a time in series from the forwarding-plane test emulator for the Trial Duration.
2. Record every request transmit time (T1) and the corresponding response received time (R1) at the forwarding-plane test emulator interface (I1) for every successful message exchange.

#### Measurements:

$$\text{Asynchronous Message Processing Time (APT1)} = \frac{\text{SUM}\{R_i\} - \text{SUM}\{T_i\}}{N_{rx}}$$

Where  $N_{rx}$  is the total number of successful messages exchanged.

$$\text{Average Asynchronous Message Processing Time} = \frac{\text{APT1} + \text{APT2} + \text{APT3} \dots \text{APT}_n}{\text{Total Trials}}$$

$$\text{Asynchronous Message Processing Time Variance (TAMv)} = \frac{\text{SUM}[\text{SQUAREOF}(\text{APT}_i - \text{TAM}_m)]}{\text{Total Trials} - 1}$$

Where TAM<sub>m</sub> is the Average Asynchronous Message Processing Time.

#### Reporting Format:

The Asynchronous Message Processing Time results **MUST** be reported in tabular format, with a row for each iteration. The last row of the table indicates the Asynchronous Message Processing Time variance, and the previous row indicates the Average Asynchronous Message Processing Time.

The report **SHOULD** capture the following information, in addition to the configuration parameters captured per Section 4.8:

- Successful messages exchanged (N<sub>rx</sub>)
- Percentage of unsuccessful messages exchanged, computed using the formula  $((1 - \text{N}_{rx}/\text{N}_{tx}) * 100)$ , where N<sub>tx</sub> is the total number of messages transmitted to the controller

If this test is repeated with a varying number of nodes with the same topology, the results **SHOULD** be reported in the form of a graph. The X coordinate **SHOULD** be the number of nodes (N), and the Y coordinate **SHOULD** be the Average Asynchronous Message Processing Time.

#### 5.1.3. Asynchronous Message Processing Rate

##### Objective:

Measure the number of responses to asynchronous messages (a new flow arrival notification message, link down, etc.) for which the controller(s) performed processing and replied with a valid and productive (non-trivial) response message.

Using a single procedure, this test will measure the following two benchmarks on the Asynchronous Message Processing Rate (see Section 2.3.1.3 of [RFC8455]):

1. Maximum Asynchronous Message Processing Rate
2. Loss-Free Asynchronous Message Processing Rate

Here, two benchmarks are determined through a series of trials where the number of messages sent to the controller(s) and the responses received from the controller(s) are counted over the Trial Duration. The message response rate and the Message Loss Ratio are calculated for each trial.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisites:

1. The controller(s) MUST have successfully completed the network topology discovery for the connected Network Devices.
2. Choose and record the Trial Duration ( $T_d$ ), the sending rate STEP size, the tolerance on equality for two consecutive trials ( $P\%$ ), and the maximum possible message-sending rate ( $N_{tx1}/T_d$ ).

#### Procedure:

1. Generate asynchronous messages continuously at the maximum possible rate on the established connections from all the emulated/simulated Network Devices for the given Trial Duration ( $T_d$ ).
2. Record the total number of responses received ( $N_{rx1}$ ) from the controller as well as the number of messages sent ( $N_{tx1}$ ) to the controller within the Trial Duration ( $T_d$ ).
3. Calculate the Asynchronous Message Processing Rate (APR1) and the Message Loss Ratio (Lr1). Ensure that the controller(s) has returned to normal operation.
4. Repeat the trial by reducing the asynchronous message-sending rate used in the last trial by the STEP size.
5. Continue repeating the trials and reducing the sending rate until both the maximum value of  $N_{rxn}$  (number of responses received from the controller) and the  $N_{rxn}$  corresponding to a Loss Ratio of zero have been found.

6. The trials corresponding to the benchmark levels MUST be repeated using the same asynchronous message rates until the responses received from the controller are equal (+/-P%) for two consecutive trials.
7. Record the number of responses received (Nrxn) from the controller as well as the number of messages sent (Ntxn) to the controller in the last trial.

#### Measurements:

$$\text{Asynchronous Message Processing Rate (APRn)} = \frac{\text{Nrxn}}{\text{Td}}$$

Maximum Asynchronous Message Processing Rate = MAX(APRn) for all n

$$\text{Asynchronous Message Loss Ratio (Lrn)} = 1 - \frac{\text{Nrxn}}{\text{Ntxn}}$$

Loss-Free Asynchronous Message Processing Rate = MAX(APRn)  
given Lrn = 0

#### Reporting Format:

The Asynchronous Message Processing Rate results MUST be reported in tabular format, with a row for each trial.

The table should report the following information, in addition to the configuration parameters captured per Section 4.8, with columns:

- Offered rate (Ntxn/Td)
- Asynchronous Message Processing Rate (APRn)
- Loss Ratio (Lr)
- Benchmark at this iteration (blank for none, Maximum Asynchronous Message Processing Rate, Loss-Free Asynchronous Message Processing Rate)

The results MAY be presented in the form of a graph. The X axis SHOULD be the offered rate, and dual Y axes would represent the Asynchronous Message Processing Rate and the Loss Ratio, respectively.



If this test is repeated with a varying number of nodes over the same topology, the results **SHOULD** be reported in the form of a graph. The X axis **SHOULD** be the number of nodes (N), and the Y axis **SHOULD** be the Asynchronous Message Processing Rate. Both the Maximum Asynchronous Message Processing Rate and the Loss-Free Asynchronous Message Processing Rate should be plotted for each N.

#### 5.1.4. Reactive Path Provisioning Time

##### Objective:

Measure the time taken by the controller to set up a path reactively between source and destination nodes, defined as the interval starting with the first flow provisioning request message received by the controller(s) at its southbound interface and ending with the last flow provisioning response message sent from the controller(s) at its southbound interface.

##### Reference Test Setup:

This test **SHOULD** use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document. The number of Network Devices in the path is a parameter of the test that may be varied from two to the maximum discovery size in repetitions of this test.

##### Prerequisites:

1. The controller **MUST** contain the network topology information for the deployed network topology.
2. The controller should know the location of the destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for "flow miss" in the Network Device is configured to "send to controller".
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while paving the entire path.

**Procedure:**

1. Send a single traffic stream from test traffic generator TP1 to test traffic generator TP2.
2. Record the time of the first flow provisioning request message sent to the controller (Tsf1) from the Network Device at the forwarding-plane test emulator interface (I1).
3. Wait for the arrival of the first traffic frame at the endpoint (i.e., test traffic generator TP2) or the expiry of the Trial Duration (Td).
4. Record the time of the last flow provisioning response message received from the controller (Tdf1) to the Network Device at the forwarding-plane test emulator interface (I1).

**Measurements:**

Reactive Path Provisioning Time (RPT1) = Tdf1 - Tsf1

Average Reactive Path Provisioning Time =  

$$\frac{\text{RPT1} + \text{RPT2} + \text{RPT3} \dots \text{RPTn}}{\text{Total Trials}}$$

Reactive Path Provisioning Time Variance (TRPv) =  

$$\frac{\text{SUM}[\text{SQUAREOF}(\text{RPTi} - \text{TRPm})]}{\text{Total Trials} - 1}$$

Where TRPm is the Average Reactive Path Provisioning Time.

**Reporting Format:**

The Reactive Path Provisioning Time results MUST be reported in tabular format, with a row for each iteration. The last row of the table indicates the Reactive Path Provisioning Time variance, and the previous row indicates the Average Reactive Path Provisioning Time.

The report should capture the following information, in addition to the configuration parameters captured per Section 4.8:

- Number of Network Devices in the path

### 5.1.5. Proactive Path Provisioning Time

#### Objective:

Measure the time taken by the controller to set up a path proactively between source and destination nodes, defined as the interval starting with the first proactive flow provisioned in the controller(s) at its northbound interface and ending with the last flow provisioning response message sent from the controller(s) at its southbound interface.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisites:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should know the location of the destination endpoint for which the path has to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for "flow miss" in the Network Device is "drop".

#### Procedure:

1. Send a single traffic stream from test traffic generator TP1 to test traffic generator TP2.
2. Install the flow entries so that the traffic travels from test traffic generator TP1 until it reaches test traffic generator TP2 through the controller's northbound interface or management interface.
3. Wait for the arrival of the first traffic frame at test traffic generator TP2 or the expiry of the Trial Duration (Td).
4. Record the time when the proactive flow is provisioned in the controller (Tsf1) at the management-plane test emulator interface (I2).
5. Record the time of the last flow provisioning message received from the controller (Tdf1) at the forwarding-plane test emulator interface (I1).

**Measurements:**

**Proactive Flow Provisioning Time (PPT1) = Tdf1 - Tsf1**

**Average Proactive Path Provisioning Time =**  

$$\frac{\text{PPT1} + \text{PPT2} + \text{PPT3} \dots \text{PPTn}}{\text{Total Trials}}$$

**Proactive Path Provisioning Time Variance (TPPv) =**  

$$\frac{\text{SUM}[\text{SQUAREOF}(\text{PPTi} - \text{TPPm})]}{\text{Total Trials} - 1}$$

Where TPPm is the Average Proactive Path Provisioning Time.

**Reporting Format:**

The Proactive Path Provisioning Time results MUST be reported in tabular format, with a row for each iteration. The last row of the table indicates the Proactive Path Provisioning Time variance, and the previous row indicates the Average Proactive Path Provisioning Time.

The report should capture the following information, in addition to the configuration parameters captured per Section 4.8:

- Number of Network Devices in the path

#### 5.1.6. Reactive Path Provisioning Rate

**Objective:**

Measure the maximum number of independent paths a controller can concurrently establish per second between source and destination nodes reactively, defined as the number of paths provisioned per second by the controller(s) at its southbound interface for the flow provisioning requests received for path provisioning at its southbound interface between the start of the test and the expiry of the given Trial Duration.

**Reference Test Setup:**

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

**Prerequisites:**

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should know the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for "flow miss" in the Network Device is configured to "send to controller".
4. Ensure that each Network Device in a path requires the controller to make the forwarding decision while provisioning the entire path.

**Procedure:**

1. Send traffic with unique source and destination addresses from test traffic generator TP1.
2. Record the total number of unique traffic frames (Ndf) received at test traffic generator TP2 within the Trial Duration (Td).

**Measurements:**

$$\text{Reactive Path Provisioning Rate (RPR1)} = \frac{Ndf}{Td}$$

$$\text{Average Reactive Path Provisioning Rate} = \frac{RPR1 + RPR2 + RPR3 \dots RPRn}{\text{Total Trials}}$$

$$\text{Reactive Path Provisioning Rate Variance (RPPv)} = \frac{\text{SUM}[\text{SQUAREOF}(RPRi - RPPm)]}{\text{Total Trials} - 1}$$

Where RPPm is the Average Reactive Path Provisioning Rate.

**Reporting Format:**

The Reactive Path Provisioning Rate results MUST be reported in tabular format, with a row for each iteration. The last row of the table indicates the Reactive Path Provisioning Rate variance, and the previous row indicates the Average Reactive Path Provisioning Rate.

The report should capture the following information, in addition to the configuration parameters captured per Section 4.8:

- Number of Network Devices in the path
- Offered rate

### 5.1.7. Proactive Path Provisioning Rate

#### Objective:

Measure the maximum number of independent paths a controller can concurrently establish per second between source and destination nodes proactively, defined as the number of paths provisioned per second by the controller(s) at its southbound interface for the paths requested in its northbound interface between the start of the test and the expiry of the given Trial Duration. The measurement is based on data-plane observations of successful path activation.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisites:

1. The controller MUST contain the network topology information for the deployed network topology.
2. The controller should know the location of destination addresses for which the paths have to be provisioned. This can be achieved through dynamic learning or static provisioning.
3. Ensure that the default action for "flow miss" in the Network Device is "drop".

#### Procedure:

1. Send traffic continuously with unique source and destination addresses from test traffic generator TP1.
2. Install corresponding flow entries so that the traffic travels from simulated sources at test traffic generator TP1 until it reaches the simulated destinations at test traffic generator TP2 through the controller's northbound interface or management interface.
3. Record the total number of unique traffic frames (Ndf) received at test traffic generator TP2 within the Trial Duration (Td).

**Measurements:**

$$\text{Proactive Path Provisioning Rate (PPR1)} = \frac{N_{df}}{T_d}$$

$$\text{Average Proactive Path Provisioning Rate} = \frac{\text{PPR1} + \text{PPR2} + \text{PPR3} \dots \text{PPRn}}{\text{Total Trials}}$$

$$\text{Proactive Path Provisioning Rate Variance (PPV)} = \frac{\text{SUM}[\text{SQUAREOF}(\text{PPRi} - \text{PPVm})]}{\text{Total Trials} - 1}$$

Where PPM is the Average Proactive Path Provisioning Rate.

**Reporting Format:**

The Proactive Path Provisioning Rate results MUST be reported in tabular format, with a row for each iteration. The last row of the table indicates the Proactive Path Provisioning Rate variance, and the previous row indicates the Average Proactive Path Provisioning Rate.

The report should capture the following information, in addition to the configuration parameters captured per Section 4.8:

- Number of Network Devices in the path
- Offered rate



#### 5.1.8. Network Topology Change Detection Time

**Objective:**

Measure the amount of time taken by the controller to detect any changes in the network topology, defined as the interval starting with the notification message received by the controller(s) at its southbound interface and ending with the first topology rediscovery message sent from the controller(s) at its southbound interface.

**Reference Test Setup:**

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

**Prerequisites:**

1. The controller MUST have successfully discovered the network topology information for the deployed network topology.
2. The periodic network discovery operation should be configured to twice the Trial Duration ( $T_d$ ) value.

**Procedure:**

1. Trigger a topology change event by bringing down an active Network Device in the topology.
2. Record the time when the first topology change notification is sent to the controller ( $T_{cn}$ ) at the forwarding-plane test emulator interface ( $I1$ ).
3. Stop the trial when the controller sends the first topology rediscovery message to the Network Device or the expiry of the Trial Duration ( $T_d$ ).
4. Record the time when the first topology rediscovery message is received from the controller ( $T_{cd}$ ) at the forwarding-plane test emulator interface ( $I1$ ).

**Measurements:**

Network Topology Change Detection Time (TDT1) =  $T_{cd} - T_{cn}$

Average Network Topology Change Detection Time =  

$$\frac{TDT1 + TDT2 + TDT3 \dots TDTn}{\text{Total Trials}}$$

Network Topology Change Detection Time Variance (NTDv) =  

$$\frac{\text{SUM}[\text{SQUAREOF}(TDTi - NTDm)]}{\text{Total Trials} - 1}$$

Where NTDm is the Average Network Topology Change Detection Time.

**Reporting Format:**

The Network Topology Change Detection Time results MUST be reported in tabular format, with a row for each iteration. The last row of the table indicates the Network Topology Change Detection Time variance, and the previous row indicates the Average Network Topology Change Detection Time.

**5.2. Scalability****5.2.1. Control Sessions Capacity****Objective:**

Measure the maximum number of control sessions the controller can maintain, defined as the number of sessions that the controller can accept from Network Devices, starting with the first control session and ending with the last control session that the controller(s) accepts at its southbound interface.

**Reference Test Setup:**

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

**Prerequisites:**

None

**Procedure:**

1. Establish control connections with the controller from every Network Device emulated in the forwarding-plane test emulator.
2. Stop the trial when the controller starts dropping the control connections.
3. Record the number of successful connections established (CCn) with the controller at the forwarding-plane test emulator.

**Measurement:**

Control Sessions Capacity = CCn

**Reporting Format:**

The Control Sessions Capacity results **MUST** be reported in addition to the configuration parameters captured per Section 4.8.

**5.2.2. Network Discovery Size****Objective:**

Measure the network size (number of nodes, links, and hosts) that a controller can discover, defined as the size of a network that the controller(s) can discover, starting with a network topology provided by the user for discovery and ending with the number of nodes, links, and hosts that the controller(s) were able to successfully discover.

**Reference Test Setup:**

This test **SHOULD** use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

**Prerequisites:**

1. The controller **MUST** support automatic network discovery.
2. The tester should be able to retrieve the discovered topology information through either the controller's management interface or northbound interface.

**Procedure:**

1. Establish the network connections between the controller and the network nodes.
2. Query the controller every  $t$  seconds (the RECOMMENDED value for  $t$  is 30) to obtain the discovered network topology information through the northbound interface or the management interface.
3. Stop the trial when the discovered network topology information remains the same as that of the last two query responses.
4. Compare the obtained network topology information with the deployed network topology information.
5. If the comparison is successful, increase the number of nodes by 1 and repeat the trial.  
If the comparison is unsuccessful, decrease the number of nodes by 1 and repeat the trial.
6. Continue the trial until the comparison (step 5) is successful.
7. Record the number of nodes for the last trial run ( $N_s$ ) where the topology comparison was successful.

**Measurement:**

Network Discovery Size =  $N_s$

**Reporting Format:**

The Network Discovery Size results MUST be reported in addition to the configuration parameters captured per Section 4.8.

### 5.2.3. Forwarding Table Capacity

#### Objective:

Measure the maximum number of flow entries a controller can manage in its Forwarding Table.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisites:

1. The controller's Forwarding Table should be empty.
2. "Flow idle time" MUST be set to a higher or infinite value.
3. The controller MUST have successfully completed network topology discovery.
4. The tester should be able to retrieve the Forwarding Table information through either the controller's management interface or northbound interface.

#### Procedures:

##### o Reactive Flow Provisioning Mode:

1. Send bidirectional traffic continuously with unique source and destination addresses from test traffic generators TP1 and TP2 at the Asynchronous Message Processing Rate of the controller.
2. Query the controller at a regular interval (e.g., every 5 seconds) for the number of learned flow entries from its northbound interface.
3. Stop the trial when the retrieved value is constant for three consecutive iterations, and record the value received from the last query (Nrp).

o Proactive Flow Provisioning Mode:

1. Install unique flows continuously through the controller's northbound interface or management interface until a failure response is received from the controller.
2. Record the total number of successful responses (Nrp).

Note:

Some controller designs for Proactive Flow Provisioning mode may require the switch to send flow setup requests in order to generate flow setup responses. In such cases, it is recommended to generate bidirectional traffic for the provisioned flows.

Measurements:

Proactive Flow Provisioning Mode:

Max Flow Entries = Total number of flows provisioned (Nrp)

Reactive Flow Provisioning Mode:

Max Flow Entries = Total number of learned flow entries (Nrp)

Forwarding Table Capacity = Max Flow Entries

Reporting Format:

The Forwarding Table Capacity results MUST be tabulated with the following information, in addition to the configuration parameters captured per Section 4.8:

- Provisioning Type (Proactive/Reactive)

### 5.3. Security

#### 5.3.1. Exception Handling

##### Objective:

Determine the effects of handling error packets and notifications on performance tests. The impact **MUST** be measured for the following performance tests:

1. Path Provisioning Rate
2. Path Provisioning Time
3. Network Topology Change Detection Time

##### Reference Test Setup:

This test **SHOULD** use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

##### Prerequisites:

1. This test **MUST** be performed after obtaining the baseline measurement results for the performance tests listed above.
2. Ensure that the invalid messages are not dropped by the intermediate devices connecting the controller and Network Devices.

##### Procedure:

1. Perform the above-listed performance tests, and send 1% of the messages from the Asynchronous Message Processing Rate test (Section 5.1.3) as invalid messages from the connected Network Devices emulated at the forwarding-plane test emulator.
2. Perform the above-listed performance tests, and send 2% of the messages from the Asynchronous Message Processing Rate test (Section 5.1.3) as invalid messages from the connected Network Devices emulated at the forwarding-plane test emulator.

##### Note:

Invalid messages can be frames with incorrect protocol fields or any form of failure notifications sent towards the controller.

**Measurements:**

Measurements **MUST** be done as per the equation defined in the "Measurements" section of the corresponding test listed under "Objective".

**Reporting Format:**

The Exception Handling results **MUST** be reported in tabular format, with a column for each of the below parameters and row for each of the above-listed performance tests:

- Without Exceptions
- With 1% Exceptions
- With 2% Exceptions

**5.3.2. Handling Denial-of-Service Attacks****Objective:**

Determine the effects of handling DoS attacks on performance and scalability tests. The impact **MUST** be measured for the following tests:

1. Path Provisioning Rate
2. Path Provisioning Time
3. Network Topology Change Detection Time
4. Network Discovery Size

**Reference Test Setup:**

This test **SHOULD** use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

**Prerequisite:**

This test **MUST** be performed after obtaining the baseline measurement results for the performance tests listed above.



**Procedure:**

Perform the above-listed tests, and launch a DoS attack towards the controller while the trial is running.

**Note:** DoS attacks can be launched on one of the following interfaces:

1. Northbound (e.g., query for flow entries continuously on the northbound interface)
2. Management (e.g., Ping requests to the controller's management interface)
3. Southbound (e.g., TCP SYN messages on the southbound interface)

**Measurements:**

Measurements **MUST** be done as per the equation defined in the "Measurements" section of the corresponding test listed under "Objective".

**Reporting Format:**

The results regarding the handling of DoS attacks **MUST** be reported in tabular format, with a column for each of the below parameters and a row for each of the above-listed tests.

- Without any attacks
- With attacks

The report should also specify the nature of the attack and the interface in question.

## 5.4. Reliability

### 5.4.1. Controller Failover Time

#### Objective:

Measure the time taken to switch from an active controller to the backup controller when the controllers work in redundancy mode and the active controller fails, defined as the interval starting when the active controller is brought down and ending with the first rediscovery message received from the new controller at its southbound interface.

#### Reference Test Setup:

This test SHOULD use the test setup illustrated in Section 3.2 of this document.

#### Prerequisites:

1. Master controller election MUST be completed.
2. Nodes are connected to the controller cluster per the implemented redundancy mode (e.g., active-standby).
3. The controller cluster should have successfully completed the network topology discovery.
4. The Network Device MUST send all new flows to the controller when it receives them from the test traffic generator.
5. The controller should have learned the location of the destination (D1) at test traffic generator TP2.

**Procedure:**

1. Send unidirectional traffic continuously with incremental sequence numbers and source addresses from test traffic generator TP1 at the rate at which the controller can process the traffic without any drops.
2. Ensure that there are no packet drops observed at test traffic generator TP2.
3. Bring down the active controller.
4. Stop the trial when the first frame after the failover operation is received on test traffic generator TP2.
5. Record the time at which the last valid frame was received (T1) at test traffic generator TP2 before the sequence error and the time at which the first valid frame was received (T2) after the sequence error at test traffic generator TP2.

**Measurements:**

Controller Failover Time = (T2 - T1)

Packet Loss = Number of missing packet sequences

**Reporting Format:**

The Controller Failover Time results MUST be tabulated with the following information:

- Number of cluster nodes
- Redundancy mode
- Controller Failover Time
- Packet Loss
- Cluster keep-alive interval

### 5.4.2. Network Re-provisioning Time

#### Objective:

Measure the time taken by the controller to reroute traffic when there is a failure in existing traffic paths, defined as the interval starting with the first failure notification message received by the controller and ending with the last flow re-provisioning message sent by the controller at its southbound interface.

#### Reference Test Setup:

This test SHOULD use one of the test setups illustrated in Section 3.1 or Section 3.2 of this document.

#### Prerequisites:

1. A network with a specified number of nodes and redundant paths MUST be deployed.
2. The controller MUST know the location of test traffic generators TP1 and TP2.
3. Ensure that the controller does not pre-provision the alternate path in the emulated Network Devices at the forwarding-plane test emulator.

#### Procedure:

1. Send bidirectional traffic continuously with a unique sequence number from test traffic generators TP1 and TP2.
2. Bring down a link or switch in the traffic path.
3. Stop the trial after receiving the first frame after network reconvergence.
4. Record the time of the last received frame prior to the frame loss at test traffic generator TP2 (TP2-Tlfr) and the time of the first frame received after the frame loss at test traffic generator TP2 (TP2-Tffr). There must be a gap in sequence numbers of these frames.
5. Record the time of the last received frame prior to the frame loss at test traffic generator TP1 (TP1-Tlfr) and the time of the first frame received after the frame loss at test traffic generator TP1 (TP1-Tffr).

**Measurements:**

Forward Direction Path Re-provisioning Time (FDRT)  
= (TP2-Tffr - TP2-Tlfr)

Reverse Direction Path Re-provisioning Time (RDRT)  
= (TP1-Tffr - TP1-Tlfr)

Network Re-provisioning Time = (FDRT + RDRT)/2

Forward Direction Packet Loss = Number of missing sequence frames  
at test traffic generator TP1

Reverse Direction Packet Loss = Number of missing sequence frames  
at test traffic generator TP2

**Reporting Format:**

The Network Re-provisioning Time results MUST be tabulated with the following information:

- Number of nodes in the primary path
- Number of nodes in the alternate path
- Network Re-provisioning Time
- Forward Direction Packet Loss
- Reverse Direction Packet Loss

**6. IANA Considerations**

This document has no IANA actions.

## 7. Security Considerations

The benchmarking tests described in this document are limited to the performance characterization of controllers in a lab environment with isolated networks.

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 controller.

Special capabilities **SHOULD NOT** exist in the controller specifically for benchmarking purposes. Any implications for network security arising from the controller **SHOULD** be identical in the lab and in production networks.

## 8. References

### 8.1. 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, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8455] Bhuvaneshwaran, V., Basil, A., Tassinari, M., Manral, V., and S. Banks, "Terminology for Benchmarking Software-Defined Networking (SDN) Controller Performance", RFC 8455, DOI 10.17487/RFC8455, October 2018, <<https://www.rfc-editor.org/info/rfc8455>>.

### 8.2. Informative References

- [OpenFlow-Spec] ONF, "OpenFlow Switch Specification" Version 1.4.0 (Wire Protocol 0x05), October 2013, <<https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.4.0.pdf>>.

## Appendix A. Benchmarking Methodology Using OpenFlow Controllers

This section gives an overview of the OpenFlow protocol [OpenFlow-Spec] and provides a test methodology for benchmarking SDN Controllers supporting the OpenFlow southbound protocol. The OpenFlow protocol is used as an example to illustrate the methodologies defined in this document.

### A.1. Protocol Overview

OpenFlow [OpenFlow-Spec] is an open standard protocol defined by the Open Networking Foundation (ONF) and used for programming the forwarding plane of network switches or routers via a centralized controller.

### A.2. Messages Overview

The OpenFlow protocol supports three message types -- namely, controller-to-switch, asynchronous, and symmetric.

Controller-to-switch messages are initiated by the controller and used to directly manage or inspect the state of the switch. These messages allow controllers to query/configure the switch ("features" messages, configuration messages), collect information from a switch (Read-State messages), send packets on a specified port of a switch (OFPT\_PACKET\_OUT messages), and modify the switch forwarding plane and state (Modify-State messages, Role-Request messages, etc.).

Asynchronous messages are generated by the switch without a controller soliciting them. These messages allow switches to update controllers to denote an arrival of a new flow (OFPT\_PACKET\_IN messages), switch state changes ("flow-removed" messages, port-status messages), and errors (Error messages).

Symmetric messages are generated in either direction without solicitation. These messages allow switches and controllers to set up a connection (Hello messages), verify liveness (Echo messages), and offer additional functionalities (Experimenter messages).

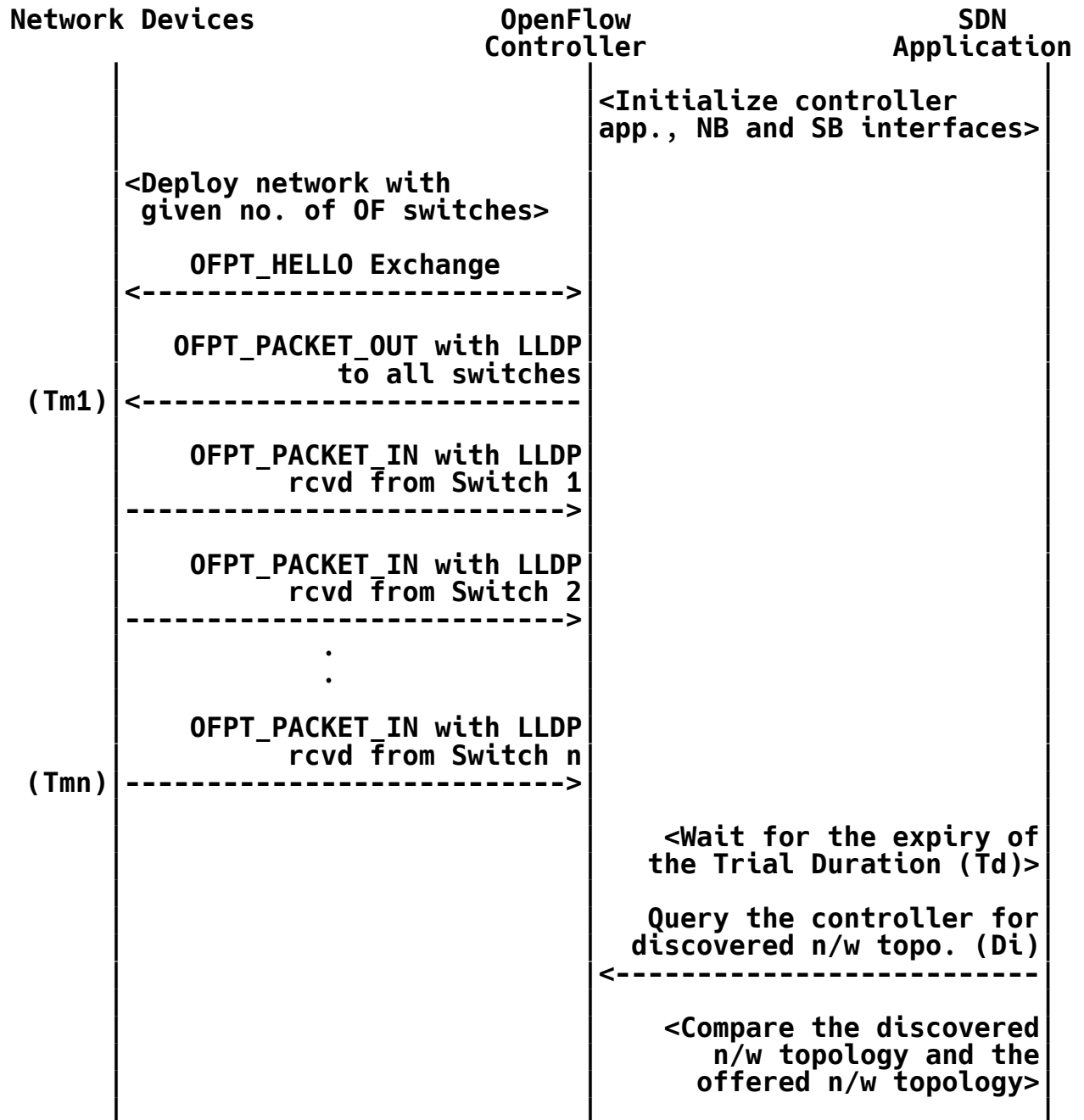
### A.3. Connection Overview

The OpenFlow channel is used to exchange OpenFlow messages between an OpenFlow switch and an OpenFlow controller. The OpenFlow channel connection can be set up using plain TCP or TLS. By default, a switch establishes a single connection with the SDN Controller. A switch may establish multiple parallel connections to a single controller (auxiliary connection) or multiple controllers to handle controller failures and load balancing.

#### A.4. Performance Benchmarking Tests

##### A.4.1. Network Topology Discovery Time

Procedure:



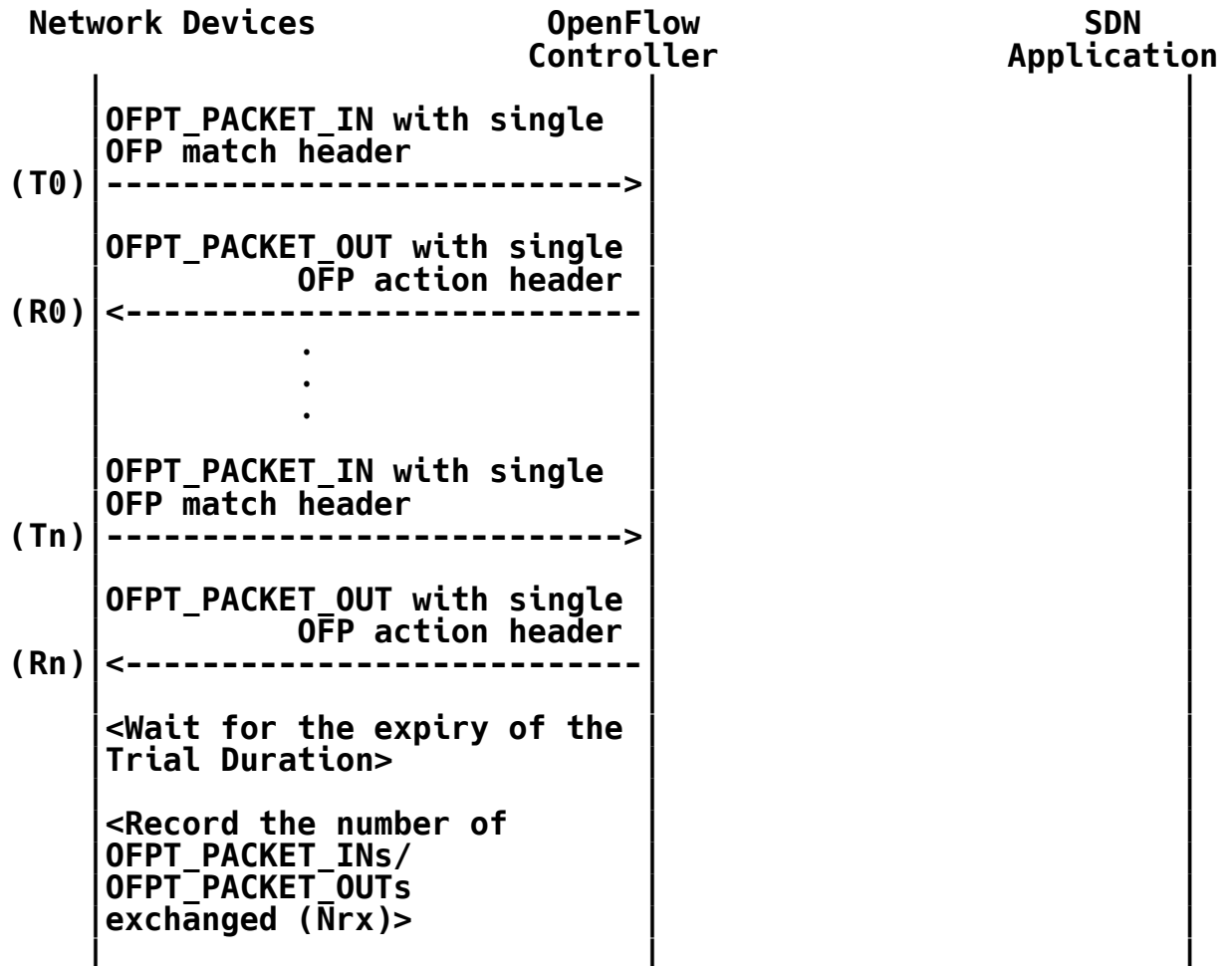


**Legend:****NB: Northbound****SB: Southbound****OF: OpenFlow****OFPT: OpenFlow Protocol****LLDP: Link-Layer Discovery Protocol****T<sub>m1</sub>: Time of reception of first LLDP message from controller****T<sub>mn</sub>: Time of last LLDP message sent to controller****Discussion:**

The Network Topology Discovery Time can be obtained by calculating the time difference between the first OFPT\_PACKET\_OUT with an LLDP message received from the controller (T<sub>m1</sub>) and the last OFPT\_PACKET\_IN with an LLDP message sent to the controller (T<sub>mn</sub>) when the comparison is successful.

#### A.4.2. Asynchronous Message Processing Time

##### Procedure:



##### Legend:

T0,T1, ..Tn: transmit timestamps of OFPT\_PACKET\_IN messages

R0,R1, ..Rn: receive timestamps of OFPT\_PACKET\_OUT messages

Nrx: Number of successful OFPT\_PACKET\_IN/OFPT\_PACKET\_OUT message exchanges

##### Discussion:

The Asynchronous Message Processing Time will be obtained by calculating the sum of  $((R0 - T0), (R1 - T1) .. (Rn - Tn)) / Nrx$ .

A.4.3. Asynchronous Message Processing Rate

Procedure:

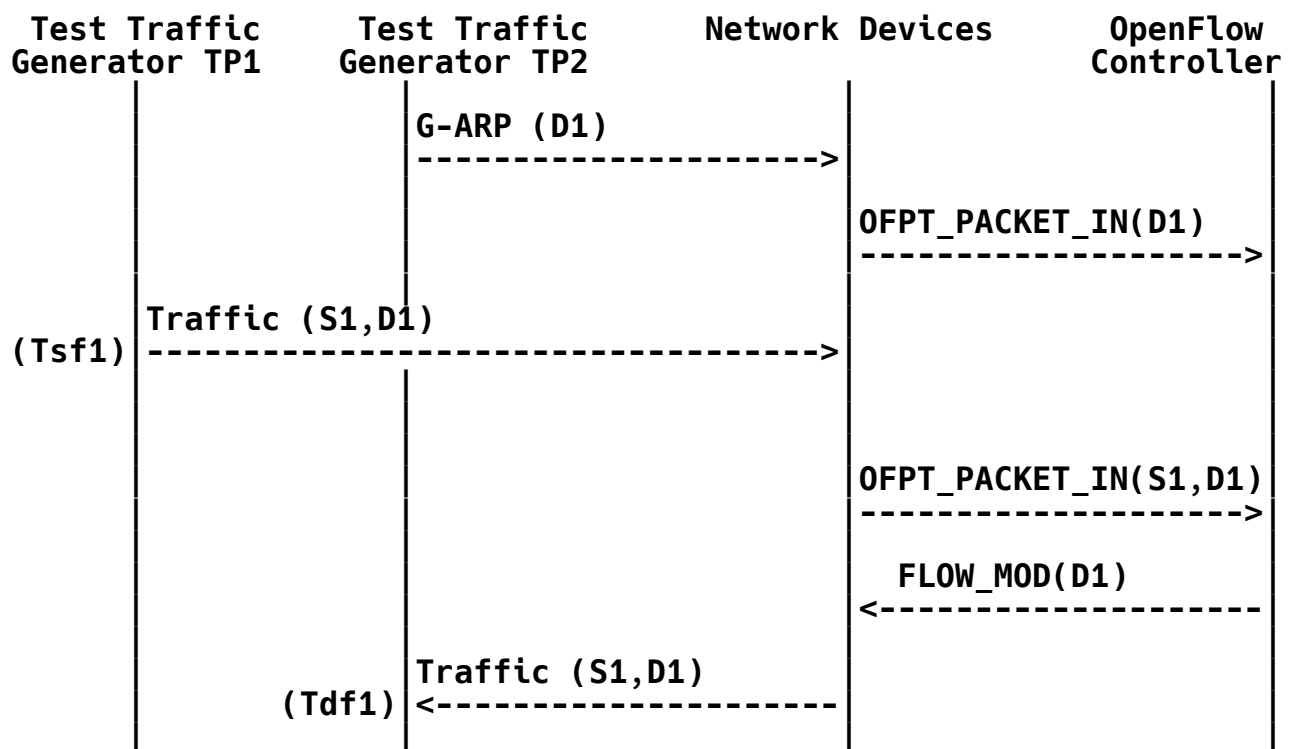
Network Devices	OpenFlow Controller	SDN Application
	OFPT_PACKET_IN with single OFP match header ----->	
	OFPT_PACKET_OUT with single OFP action header <-----	
	. . . . .	
	OFPT_PACKET_IN with single OFP match header ----->	
	OFPT_PACKET_OUT with single OFP action header <-----	
	<Repeat the steps until the expiry of the Trial Duration>	
(Ntx1)	<Record the number of OFP match headers sent>	
(Nrx1)	<Record the number of OFP action headers rcvd>	

Note: The Ntx1 on initial trials should be greater than Nrx1. Repeat the trials until the Nrxn for two consecutive trials equals (+/-P%).

**Discussion:**

Using a single procedure, this test will measure two benchmarks:

1. The Maximum Asynchronous Message Processing Rate will be obtained by calculating the maximum OFPT\_PACKET\_OUTs (Nrxn) received from the controller(s) across n trials.
2. The Loss-Free Asynchronous Message Processing Rate will be obtained by calculating the maximum OFPT\_PACKET\_OUTs received from the controller(s) when the Loss Ratio equals zero. The Loss Ratio is obtained by calculating  $1 - N_{rxn}/N_{txn}$ .

**A.4.4. Reactive Path Provisioning Time****Procedure:**

**Legend:**

G-ARP: Gratuitous ARP message

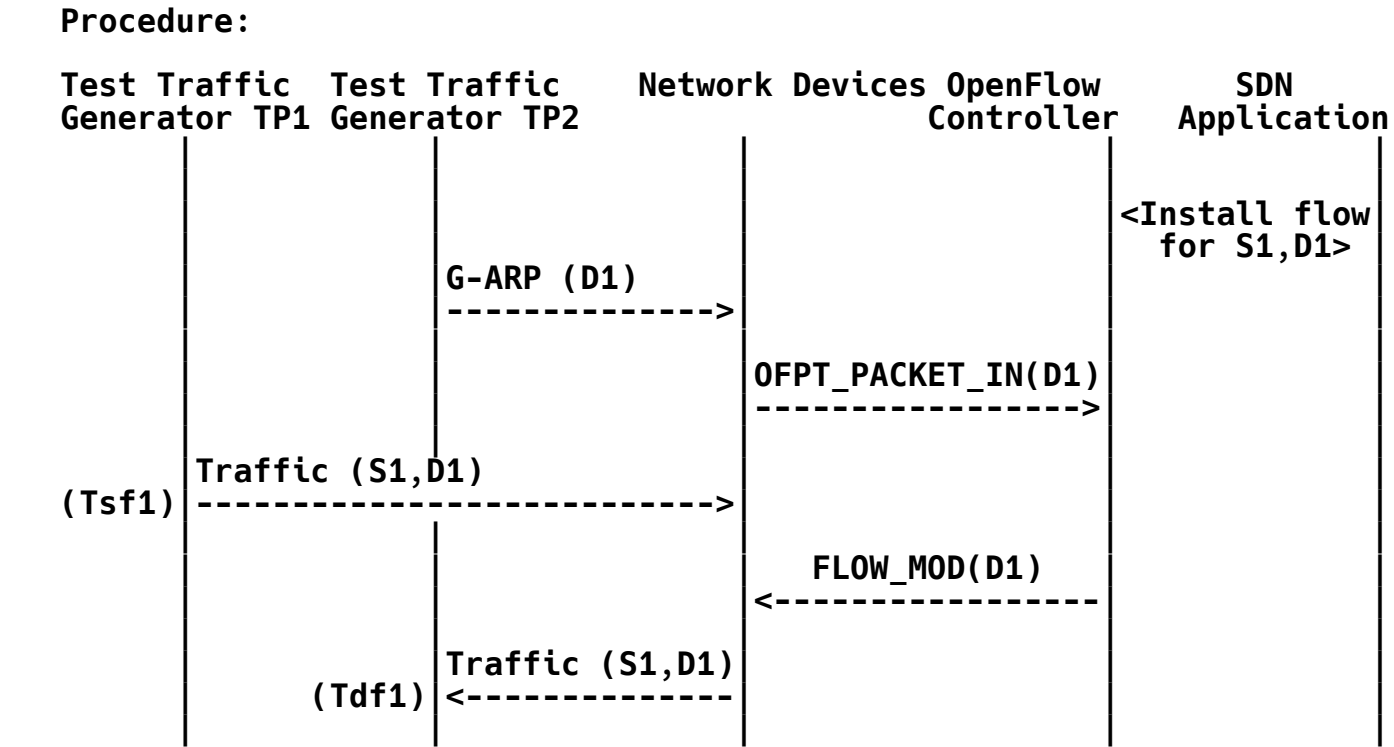
Tsf1: Time of first frame sent from TP1

Tdf1: Time of first frame received from TP2

**Discussion:**

The Reactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive times of the traffic ( $T_{sf1} - T_{df1}$ ).

A.4.5. Proactive Path Provisioning Time



Legend:

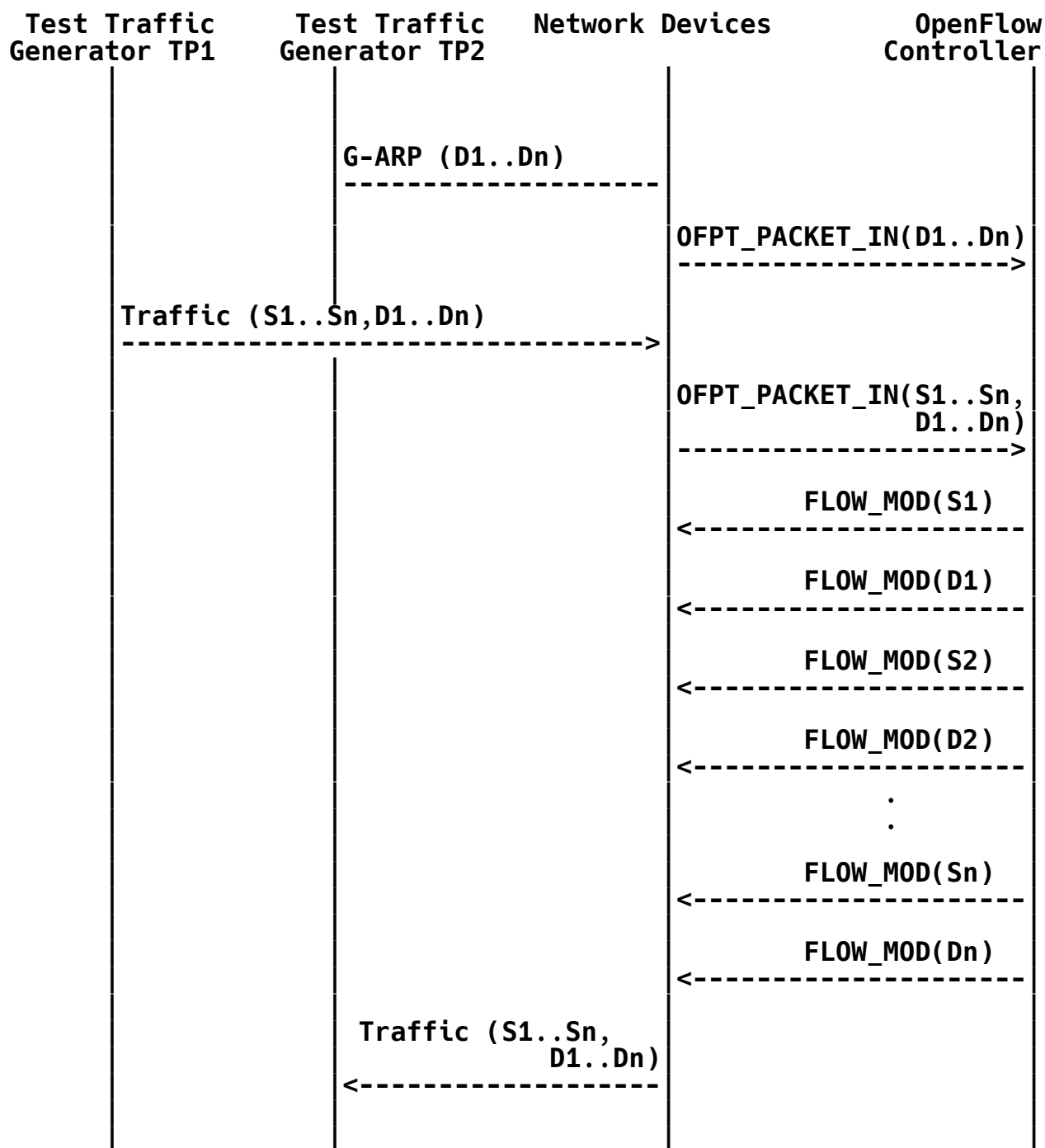
G-ARP: Gratuitous ARP message  
Tsf1: Time of first frame sent from TP1  
Tdf1: Time of first frame received from TP2

Discussion:

The Proactive Path Provisioning Time can be obtained by finding the time difference between the transmit and receive times of the traffic (Tsf1 - Tdf1).

## A.4.6. Reactive Path Provisioning Rate

## Procedure:



**Legend:**

G-ARP: Gratuitous ARP message

D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ...,  
Destination Endpoint n

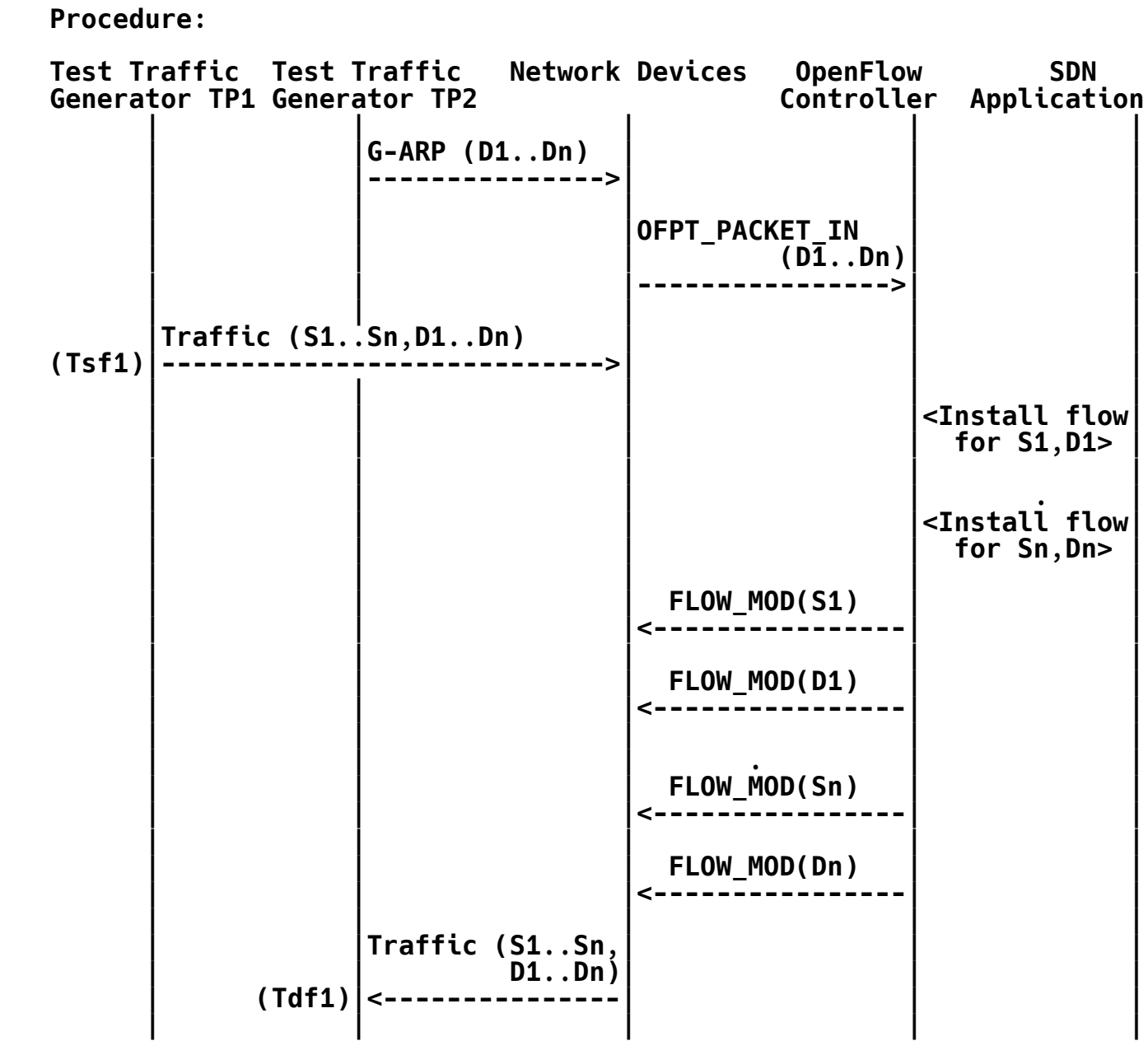
S1..Sn: Source Endpoint 1, Source Endpoint 2 ...,  
Source Endpoint n

**Discussion:**

The Reactive Path Provisioning Rate can be obtained by finding the total number of frames received at test traffic generator TP2 after the Trial Duration.



A.4.7. Proactive Path Provisioning Rate



**Legend:**

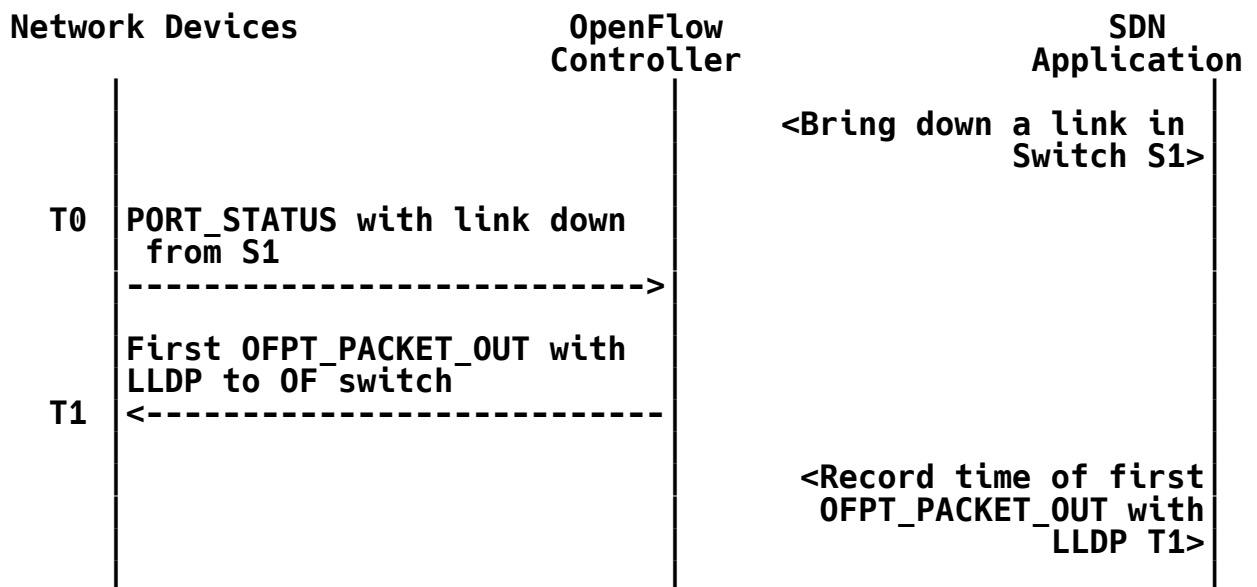
G-ARP: Gratuitous ARP message

D1..Dn: Destination Endpoint 1, Destination Endpoint 2 ...,  
Destination Endpoint n

S1..Sn: Source Endpoint 1, Source Endpoint 2 ...,  
Source Endpoint n

**Discussion:**

The Proactive Path Provisioning Rate can be obtained by finding the total number of frames received at test traffic generator TP2 after the Trial Duration.

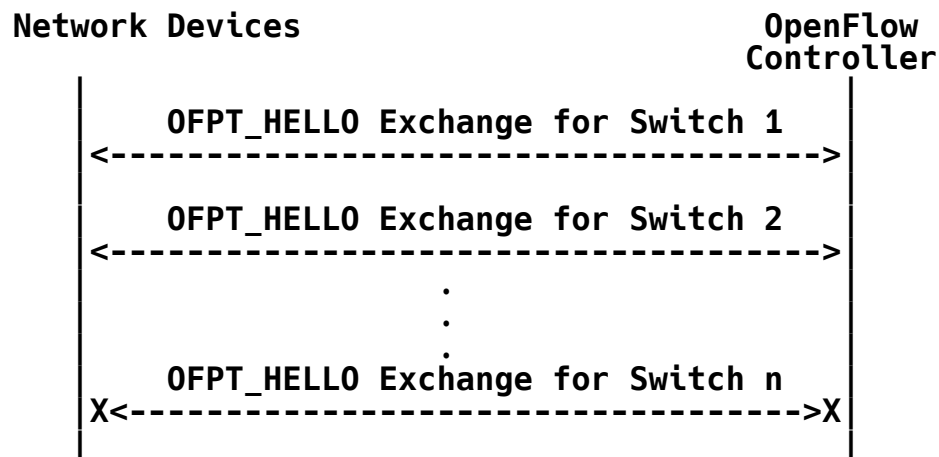
**A.4.8. Network Topology Change Detection Time****Procedure:****Discussion:**

The Network Topology Change Detection Time can be obtained by finding the difference between the time that OpenFlow Switch S1 sends the PORT\_STATUS message (T0) and the time that the OpenFlow controller sends the first topology rediscovery message (T1) to OpenFlow switches.

## A.5. Scalability

### A.5.1. Control Sessions Capacity

#### Procedure:



#### Discussion:

The value of Switch (n - 1) will provide the Control Sessions Capacity.

## A.5.2. Network Discovery Size

## Procedure:

Network Devices	OpenFlow Controller	SDN Application
		<Deploy network with given no. of OF switches N>
OFPT_HELLO Exchange	<----->	
OFPT_PACKET_OUT with LLDP to all switches	<----->	
OFPT_PACKET_IN with LLDP rcvd from Switch 1	----->	
OFPT_PACKET_IN with LLDP rcvd from Switch 2	----->	
:		
OFPT_PACKET_IN with LLDP rcvd from Switch n	----->	
		<Wait for the expiry of the Trial Duration (Td)>
		Query the controller for discovered n/w topo. (N1)
		<----->
		<If N1==N, repeat Step 1 with N + 1 nodes until N1<N >
		<If N1<N, repeat Step 1 with N=N1 nodes once and exit>

**Legend:**

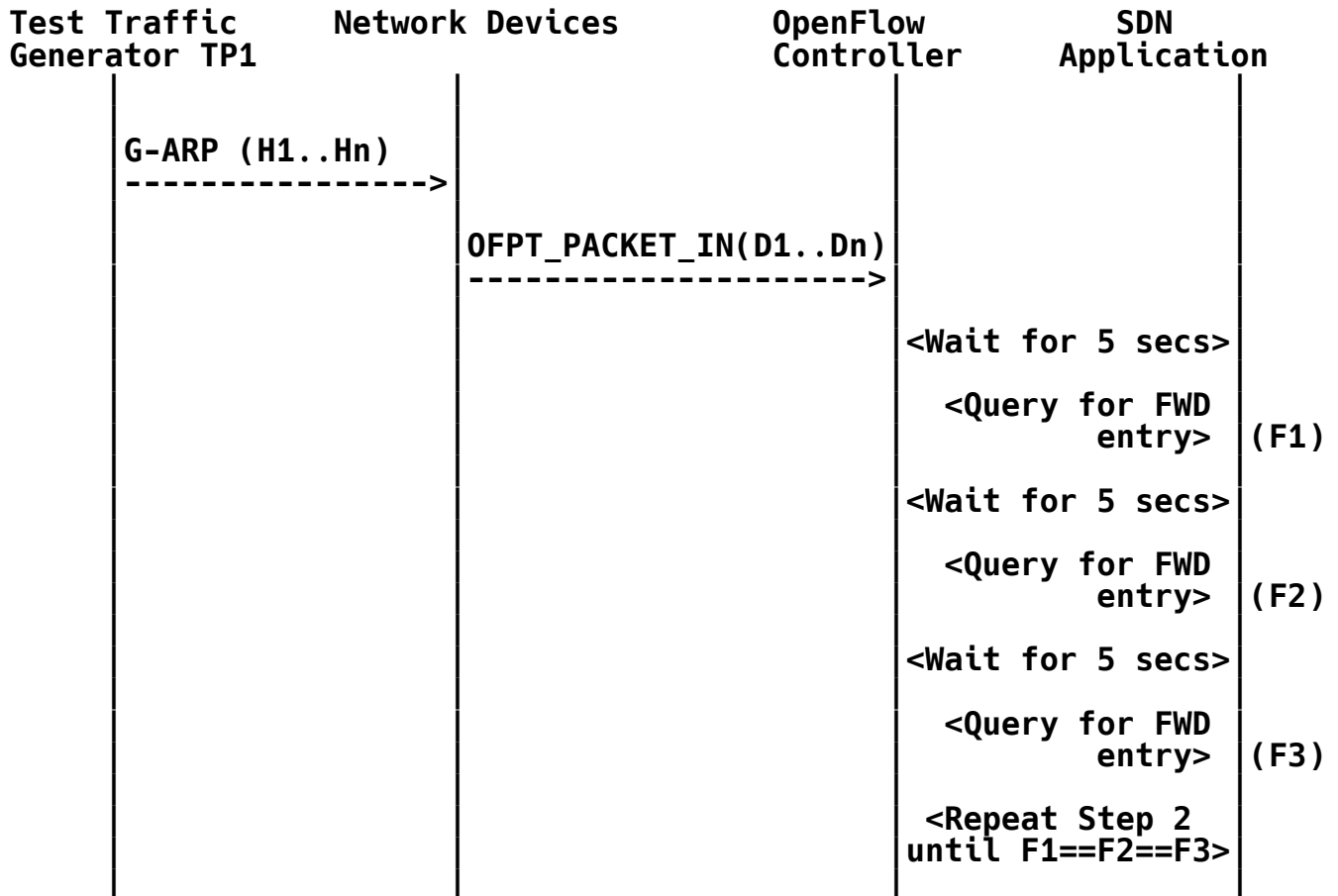
n/w topo: Network topology  
OF: OpenFlow

**Discussion:**

The value of N1 provides the Network Discovery Size value. The Trial Duration can be set to the stipulated time within which the user expects the controller to complete the discovery process.

## A.5.3. Forwarding Table Capacity

## Procedure:



## Legend:

G-ARP: Gratuitous ARP message  
H1..Hn: Host 1 .. Host n  
FWD: Forwarding Table

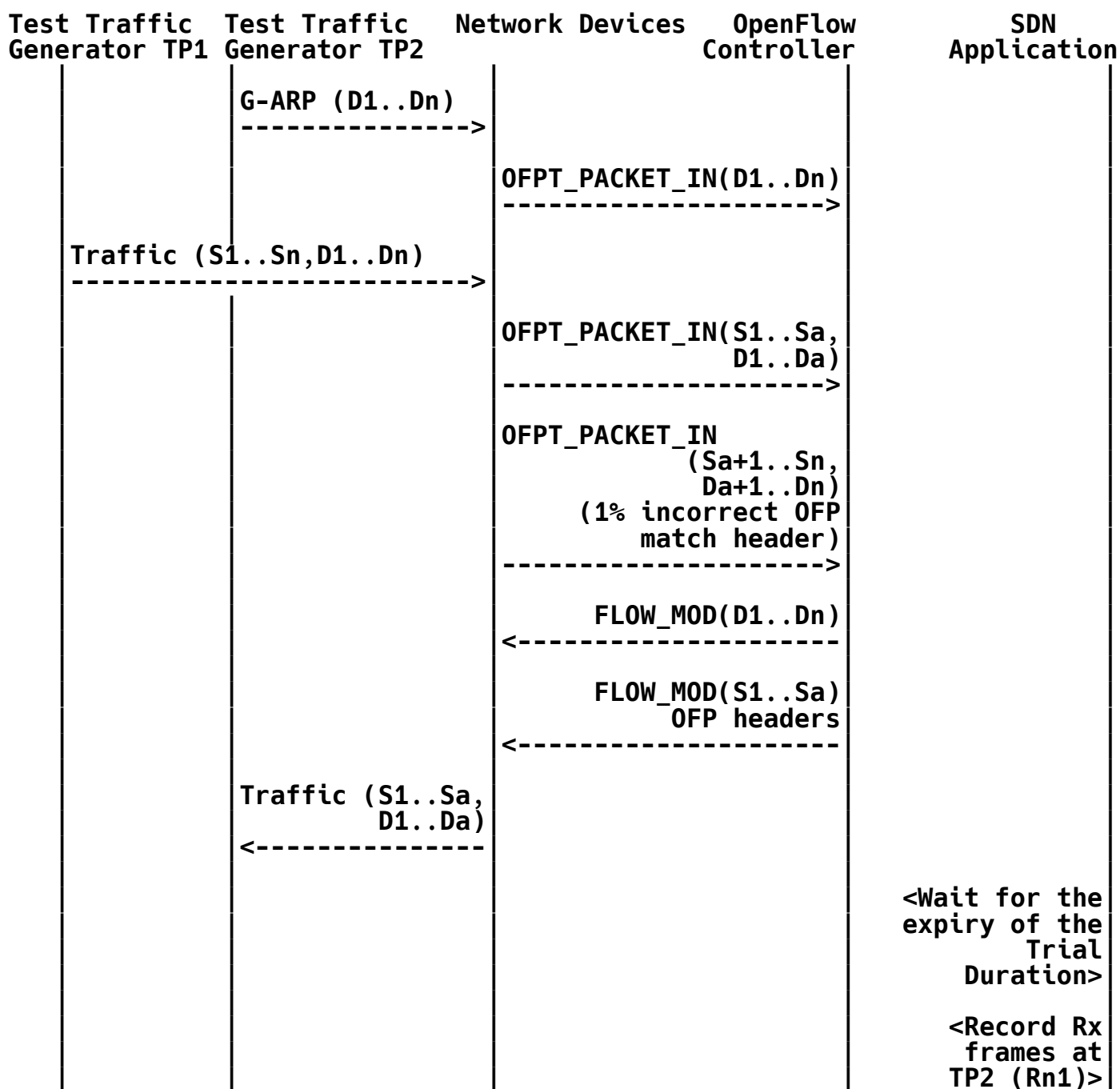
## Discussion:

Query the controller's Forwarding Table entries multiple times, until three consecutive queries return the same value. The last value retrieved from the controller will provide the Forwarding Table Capacity value. The query interval is user configurable. The interval of 5 seconds shown in this example is for representational purposes.

## A.6. Security

## A.6.1. Exception Handling

Procedure:



			<Repeat Step 1 with 2% incorrect OFPT_PACKET_INs>
			<Record Rx frames at TP2 (Rn2)>

**Legend:**

G-ARP: Gratuitous ARP message

OFPT\_PACKET\_IN(Sa+1..Sn,Da+1..Dn): OFPT\_PACKET\_IN with  
 wrong version number

Rn1: Total number of frames received at Test Port 2  
 with 1% incorrect frames

Rn2: Total number of frames received at Test Port 2  
 with 2% incorrect frames

**Discussion:**

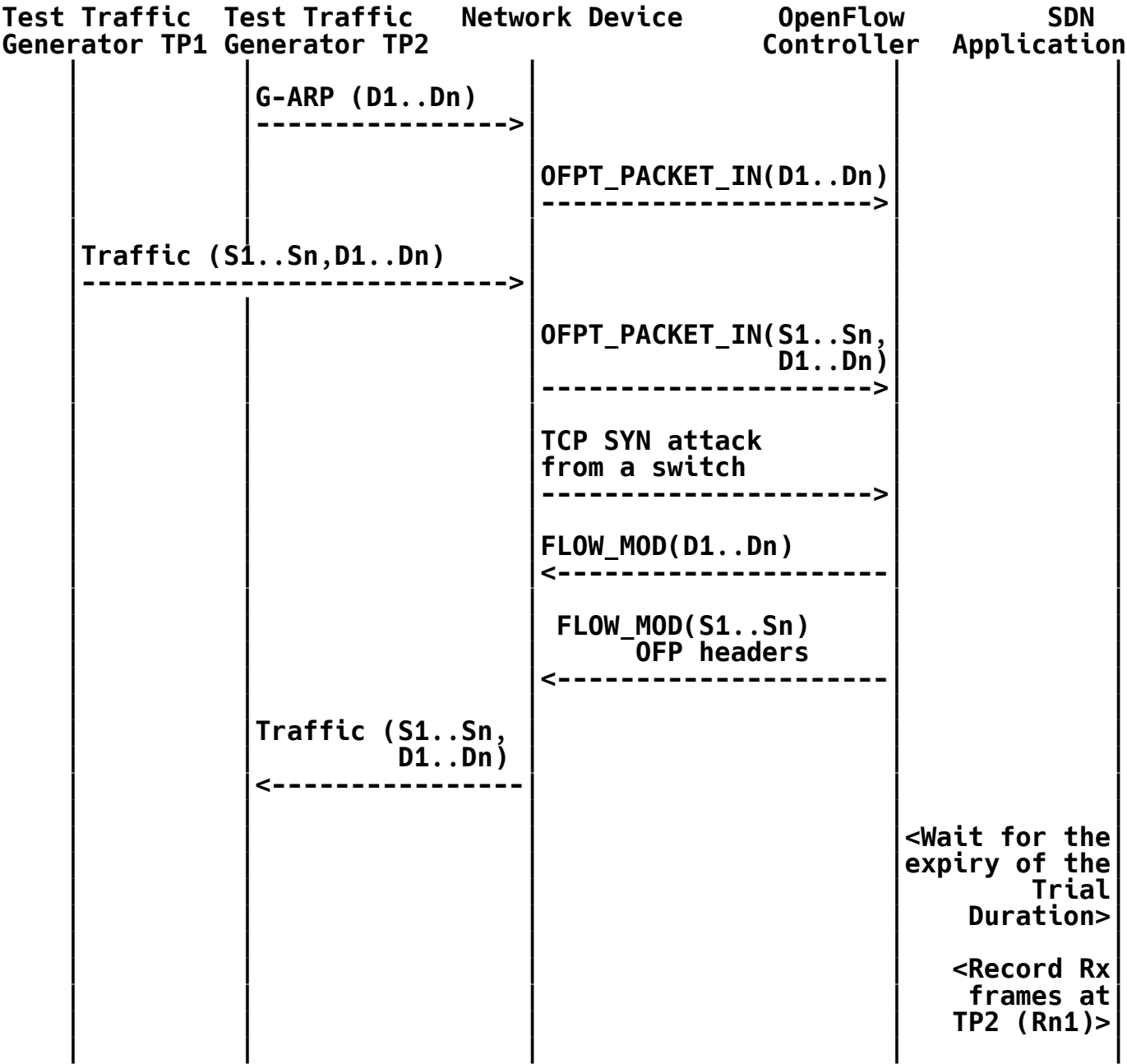
The traffic rate sent towards the OpenFlow switch from Test Port 1 should be 1% higher than the Path Programming Rate. Rn1 will provide the Path Provisioning Rate of the controller when 1% of incorrect frames are received, and Rn2 will provide the Path Provisioning Rate of the controller when 2% of incorrect frames are received.

The procedure defined above provides test steps to determine the effects of handling error packets on the Path Programming Rate. The same procedure can be adapted to determine the effects on other performance tests listed in this benchmarking test.



A.6.2. Handling Denial-of-Service Attacks

Procedure:



**Legend:**

**G-ARP: Gratuitous ARP message**

**Discussion:**

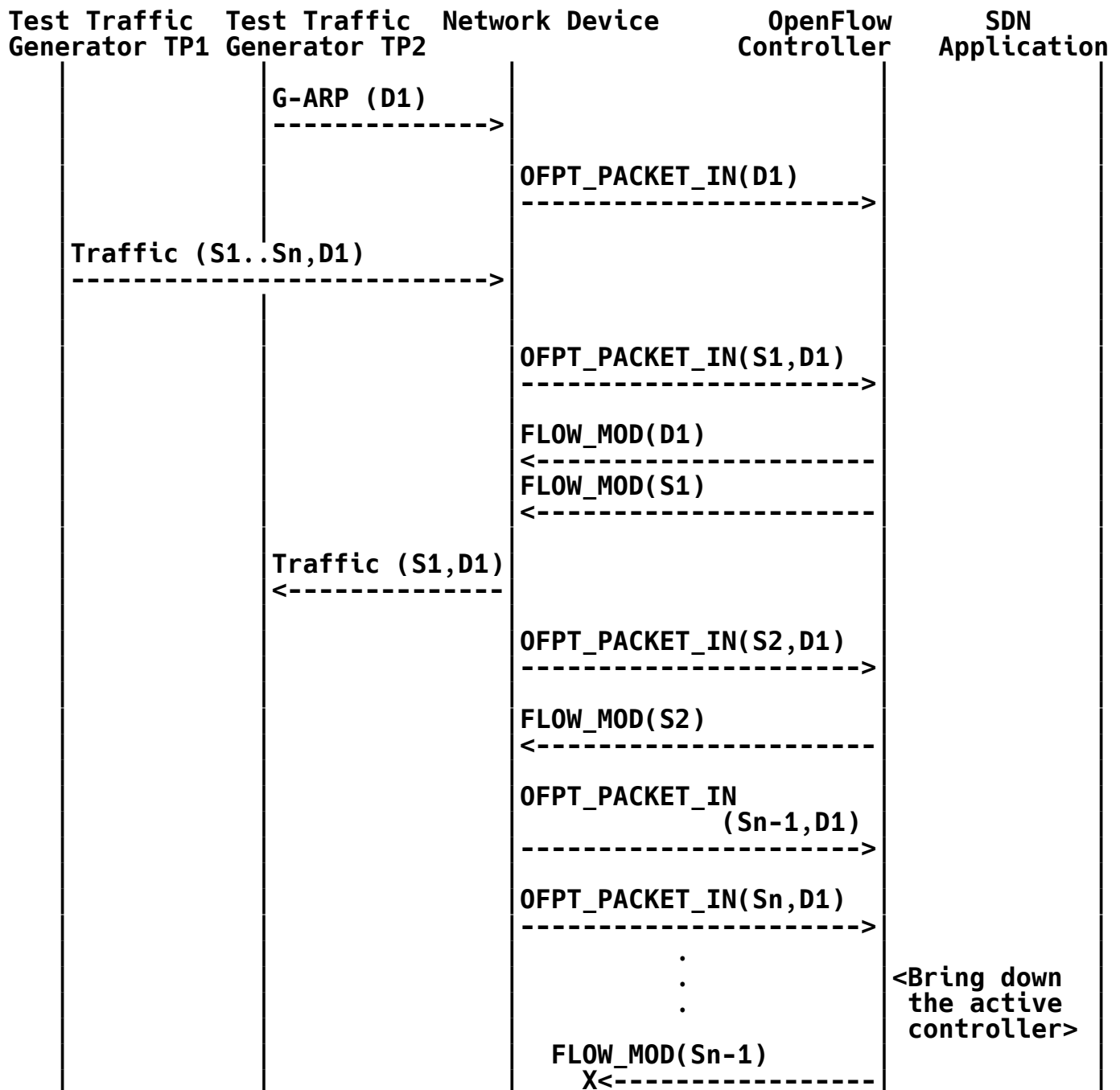
A TCP SYN attack should be launched from one of the emulated/simulated OpenFlow switches. Rn1 provides the Path Programming Rate of the controller upon handling a denial-of-service attack.

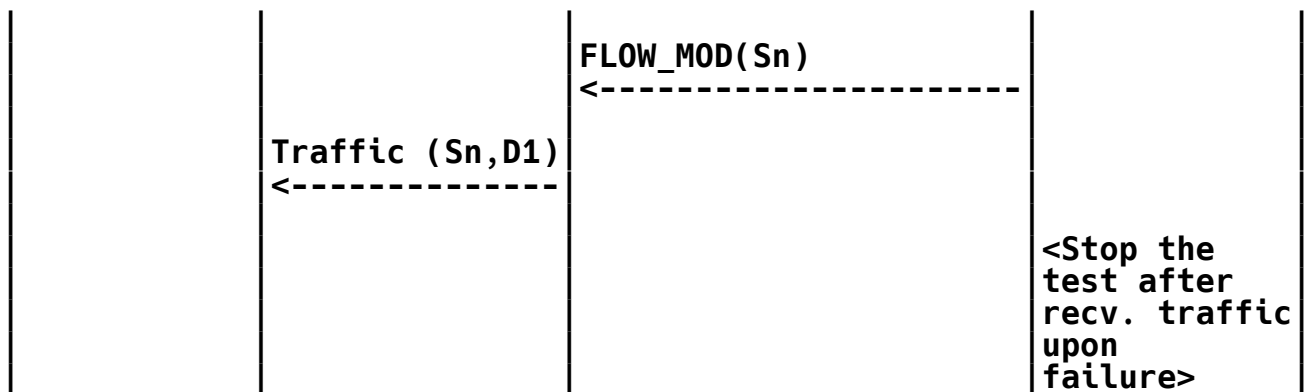
The procedure defined above provides test steps to determine the effects of handling denial of service on the Path Programming Rate. The same procedure can be adapted to determine the effects on other performance tests listed in this benchmarking test.

## A.7. Reliability

### A.7.1. Controller Failover Time

Procedure:



**Legend:**

G-ARP: Gratuitous ARP message

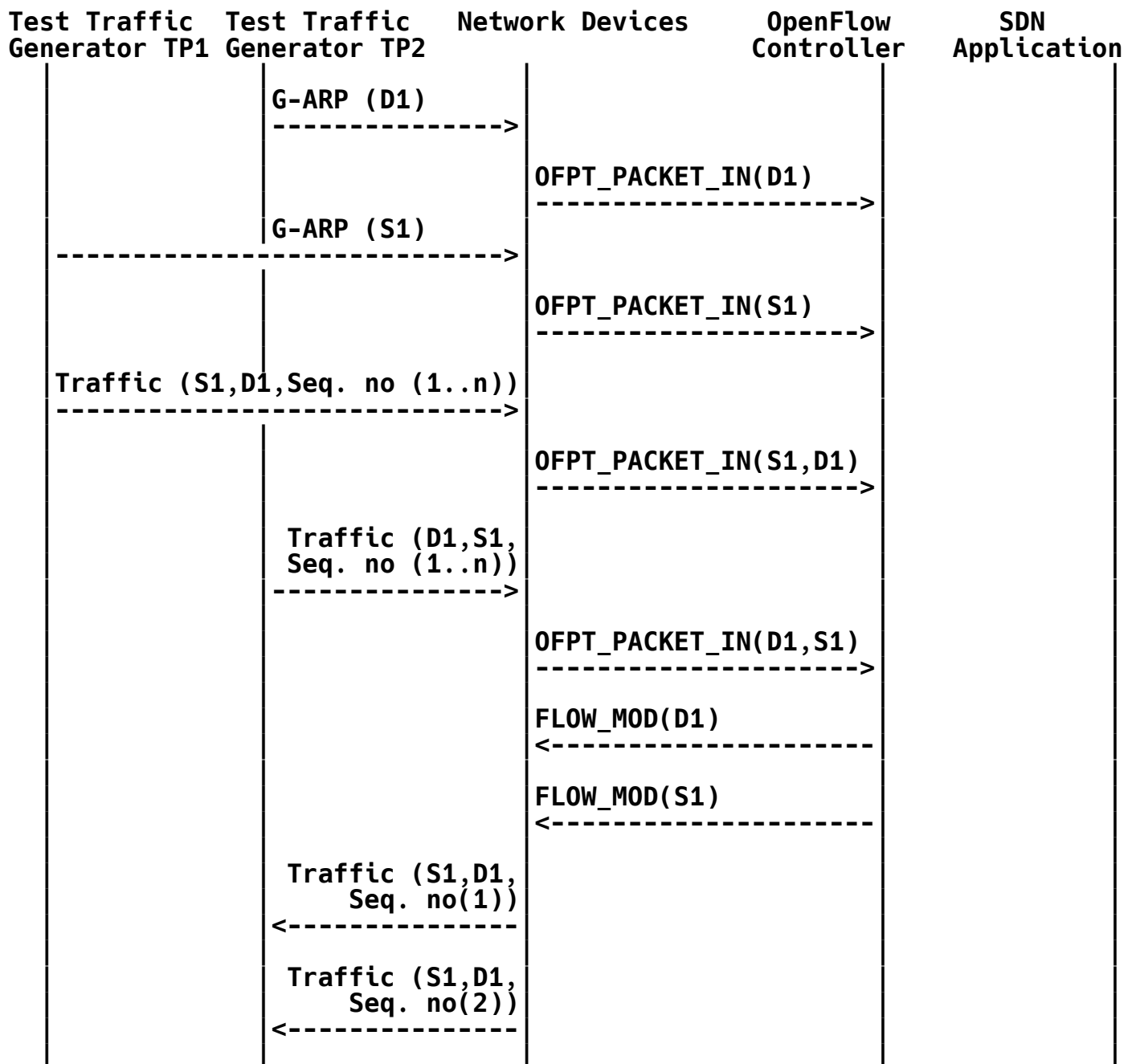
**Discussion:**

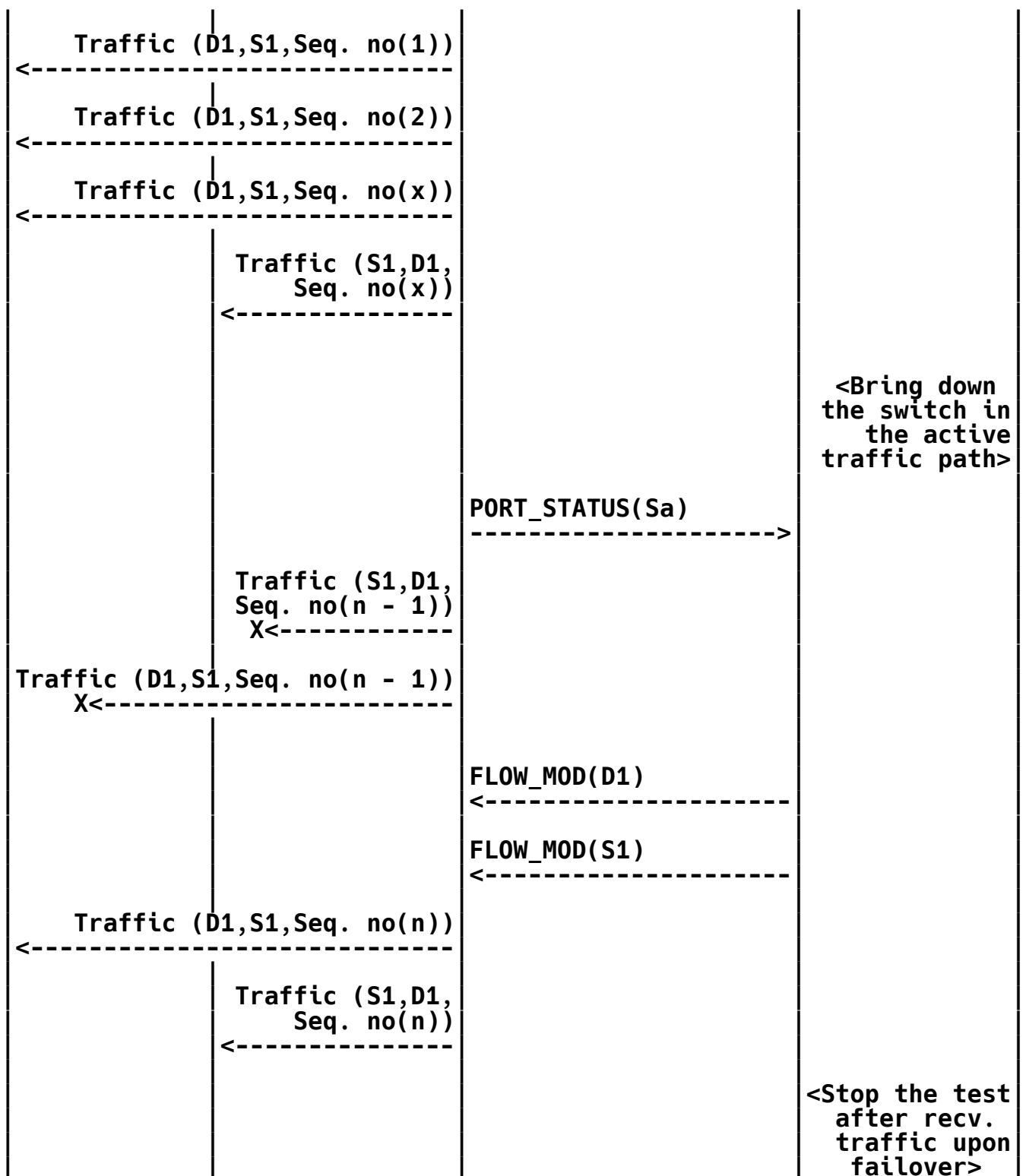
The time difference between the last valid frame received before the traffic loss and the first frame received after the traffic loss will provide the Controller Failover Time.

If there is no frame loss during the Controller Failover Time, the Controller Failover Time can be deemed negligible.

## A.7.2. Network Re-provisioning Time

## Procedure:





**Legend:**

G-ARP: Gratuitous ARP message

Seq. no: Sequence number

Sa: Neighbor switch of the switch that was brought down

**Discussion:**

The time difference between the last valid frame received before the traffic loss (packet with sequence number x) and the first frame received after the traffic loss (packet with sequence number n) will provide the Network Re-provisioning Time.

Note that the trial is valid only when the controller provisions the alternate path upon network failure.

**Acknowledgments**

The authors would like to thank the following individuals for providing their valuable comments regarding the earlier draft versions of this document: Al Morton (AT&T), Sandeep Gangadharan (HP), M. Georgescu (NAIST), Andrew McGregor (Google), Scott Bradner, Jay Karthik (Cisco), Ramki Krishnan (VMware), Boris Khasanov (Huawei), and Brian Castelli (Spirent).

**Authors' Addresses**

Bhuvaneshwaran Vengainathan  
Veryx Technologies Inc.  
1 International Plaza, Suite 550  
Philadelphia, PA 19113  
United States of America

Email: [bhuvaneshwaran.vengainathan@veryxtech.com](mailto:bhuvaneshwaran.vengainathan@veryxtech.com)

Anton Basil  
Veryx Technologies Inc.  
1 International Plaza, Suite 550  
Philadelphia, PA 19113  
United States of America

Email: [anton.basil@veryxtech.com](mailto:anton.basil@veryxtech.com)

Mark Tassinari  
Hewlett Packard Enterprise  
8000 Foothills Blvd.  
Roseville, CA 95747  
United States of America

Email: [mark.tassinari@hpe.com](mailto:mark.tassinari@hpe.com)

Vishwas Manral  
NanoSec Co  
3350 Thomas Rd.  
Santa Clara, CA 95054  
United States of America

Email: [vishwas.manral@gmail.com](mailto:vishwas.manral@gmail.com)

Sarah Banks  
VSS Monitoring  
930 De Guigne Drive  
Sunnyvale, CA 94085  
United States of America

Email: [sbanks@encrypted.net](mailto:sbanks@encrypted.net)