March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

# A Comparative Study for ONOS Reactive and Intent Forwarding Technique in Network Performance

[1]Fathul Arif Kamarudin, [2]Megat Norulazmi Megat Mohamed Noor, [3]Fuead Ali

[1]MIIT IoT Research Group, Univesiti Kuala Lumpur, Kuala Lumpur, Malaysia
[2,3] Dept. Computer Enginering, Univesiti Kuala Lumpur, Kuala Lumpur, Malaysia
[1]fathul.kamarudin@s.unikl.edu.my, [2]megatnorulazmi@unikl.edu.my, [3]fuead@unikl.edu.my

***Article Info***
***Volume 83***
***Page Number: 8205 - 8214***
***Publication Issue:***
***March - April 2020***

***Article History***
***Article Received****: 24 July 2019*
***Revised****: 12 September 2019*
***Accepted****: 15 February 2020*
***Publication****: 09 April 2020*

## Abstract

The standard of network instability performed by numerous undertakings from telco enterprises rises a new type of hurdle in dealing with the ever-expanding traffic over a similar WAN. The probability of risk and venture involves with regards to such service frequently decreased the chance to grasp its performance in a proper environment. Additionally, the need to physically change the policies into low-level command often results to human error. This study is to address the impacts of SDN (Software-Defined Network) on network performance towards the distinctions between the reactive and intent forwarding technique based on the evaluation matrix. The testbed is performed under Mininet test system which hooked with ONOS (Open Network Operating System) controller. Mesh type topology with a fix parameter matrix is set; utilizing a pseudo-random topology generator. The expected result from this study is to demonstrate the changes on the network performance under two classes of forwarding; reactive and intent based on the evaluation matrix. This study contributes towards acknowledging the viability of SDN design on network performance and to improve future benchmarking on SDN testbed.

**Keywords;** *Intent forwarding, network, performance, reactive forwarding, software defined network, testbed.*

## I. INTRODUCTION

Telco administrators need to tailor their systems to be nimble, productive and capable to handle the operational expense at moderate level. For that, a testbed is a reasonable technique that can be adopted by mimicking the requirementproposed by the R&D of Telco enterprise to reap the criticism and feedbacks one may get in a dynamicsituation. The motivation behind this study is to give proper analysis and discussion on the preliminarydata acquired from the initial experimental testbedperformed with respect to the change-over of network through the usage of specific techniques accessible in software-defined network. In this study, the dataobtain from the final result depends on the correlation between the two type of forwardingmethods; intent and reactive technique. The reactiveforwarding utilized in the study is a host-explicit technique which depends on entries for each destination host associated with the same virtual network. The intenttechniqueinstead applies ONOS's intentframework which act as a policy based command. Both form of forwarding is tested under a list of parameter matrix with explicit parameter assessment. The test is controlled under Ubuntu environment with Mininet and ONOS implemented. The testbed comprises of a virtual network topology utilizing Mininet that is connected to ONOS controller. From the testbed, we anticipate changes in the networkperformancewhen tested under these two types of forwardingtechnique and



*Published by: The Mattingley Publishing Co., Inc.*

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

validates the reasons behind the methods in whichsuch behaviour occurred.

## II. RELATED WORK

The term legacy is a typical way to portray an old system components that either has been irrelevant or can possibly be one in the future. The responsibilities to build up revenue via comprehensive strategies such as Bandwidth on Demand or Pay for Network Features falls under Telco operators. The way that the computer networks are designed however allow for collaborative amount of network elements, ranging from servers, routers, switches to middle boxes. Furthermore, in view of the present pattern of technological expansion, a colossal challenge must be addressed for network service providers to constantly keep vigilante with the exponential development of web data traffic. Concern has been put forth by Hakiri [2] expressing that the demand for traffic usage progression is corresponded to network architecture such that the present network infrastructures capability to adhere a large growth of data traffic is restricted.

### A. Network Challenges and Solutions

One of the significant challenges that ISP should be focused on with respect to the network services as pointed out by fellow researchers is the over-dependent towards equipment to deal with the multifaceted nature of network design [3], [1], [4]. According to researchers [6], the recent operational strategy utilized in network services is incapable to rout the intricacy in network infrastructure. When the data transfer capacity request grows, so does the network infrastructure ability to resist the resources require to handle such service. These challenges should be tended to, and a few solutions have been given in a manner that hypothetically able to handle the issues pointed out by the current practice of network services. The issues and its conceivable solutions for each challenge can be condense into Table 1 as clarified below.

| **Network Services** | |
|---|---|
| ***Issues*** | ***Possible Solutions*** |
| Absence of programming driven control to optimize network multifaceted nature [6], [8]. | Deploy a unify controller that cover the internal details of network control and functions as programmatic interface to the application [7], [9]. |
| Cost and operational issue for current network services influencing the Opex and Capex of an enterprise [12],[10]. | A cost - sparing methodology for structuring versatile network on demand to oversee topology and dynamically control network resources [11]. |
| Enormous data transfer can obstruct the switches with packets [11] | Open Network Operating System implement a centralize network design of adaptability and scale-out. [13]. |

**TABLE I. The challenges and possible solutions of bandwidth-on-demand -on-demand**

### B. Software-Defined Network Concept

Software-defined networking (SDN) is a computer methodology where its primary functions allow for logical control of the network and enable the possibility for programmable and adaptable computer network. In its fundamental, the SDN architecture separate the data plane and the control plane where the SDN software components oversees the control plane of the network; enabling researchers and software engineers to implement network application between the gap of the planes through its network wide deliberation. In SDN network management, a vendor-explicit interface application for overseeing network device overalls is supplanted with logical centralize SDN controller. As indicated by the researchers [7], they talked about the possibility of "platform as a service" model for network administration. It is considered as a typical inclination to isolate the foundation of network architecture from the service management and concealing the basic physical network and the topology to the client. The researchers further elaborate that the client is instead keen on having the option to design policies and characterizing how packets are dealt with.

Fig. 1 below demonstrates the SDN layer is isolated into three sections; application, control and device layers. The principle idea of SDN is to consolidate



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

the isolated control layer into one single network controller. What it means is that for this type of design, the network device is required for dealing with the data layer and sending the data packet from one point to another based on SDN controller's policies. Therefore, one explicit controller which is coordinated through application in application layer oversees each switch by utilizing application programming interface (API).

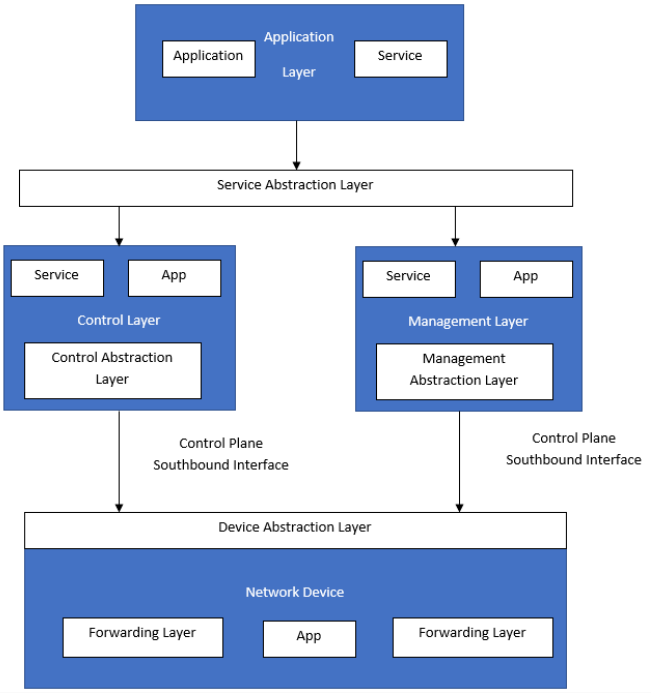


**Fig. 1. The layers in SDN architecture**

## III. METHOD OF STUDY

### A. Study Design

This study is performedusing anexperimentaltestbedby utilizing virtual devices to determine the distinction of performance in term of a fixed assessmentmatrix between reactive and intent forwarding techniques. There are two type of techniques accessible in ONOS for forwarding packets. One is simplereactiveforwarding application (onos-application fwd) and the other one is intent based reactiveforwarding (onos-application ifwd). Reactive forwarding alludes to the mechanism utilized for introducing network switch forwardingentries. These entries are introduced on request after a sender has sent the packets. The accompanying processes are performed when a packets enters the input interface. 1) The fields of the packets header are assessed against the table. 2) If no match (no table-miss section) is incorporated, the packets is erased. 3) If no match exists and a table-miss section exists, execute the defined table-miss action. 4) If the match is accurate, update the counters, run the directives and sent to a table in the pipeline or transmitted from the exit port [13].

Reactive forwarding can be performed by installing the application onos-ap-fwd in ONOS which is utilized in the experiment. Intentforwarding is an unchangeable model object that portrays application's solicitation to change the network's conduct in ONOS that may depict the network assets, constraints or criteria [13]. Fig. 2 below demonstrates the flowchart of intentforwarding in ONOS. Noticed that the shaded states are transitional and expected to last temporarily while the rest of the states are space states where the intent may require some time.

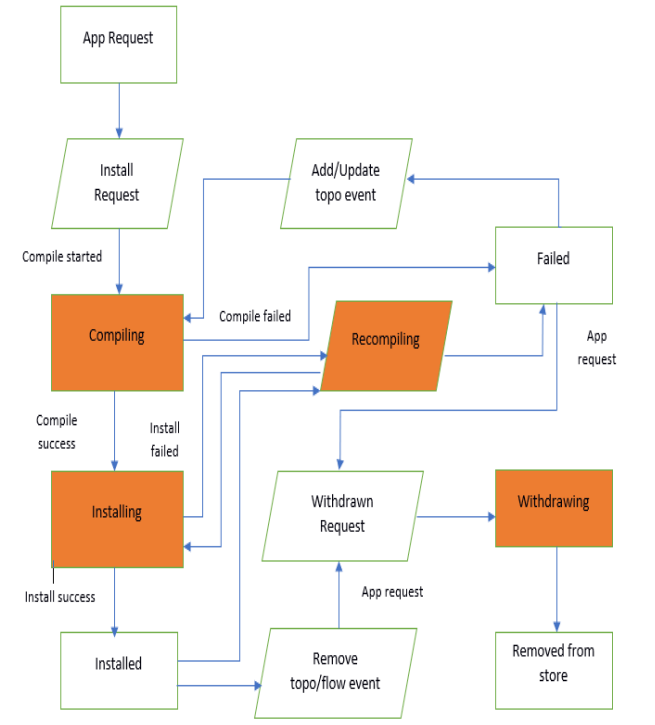


**Fig. 2. The flow of intents compilation in ONOS**



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

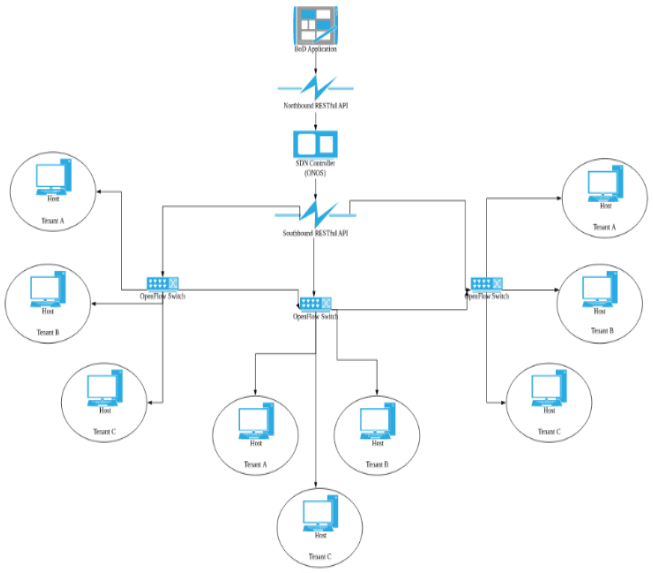

**Fig. 3. The overview of the experimental testbed**

Fig. 3 above demonstrates the outline of the testbed. The test is performed by utilizing Ubuntu 18.04 under localhost environment. The information for this study is gathered using experimental testbed with a few parameter matrix as appeared in Table 2 utilizing the predetermined assessment parameter. The parameter matrix is comprised of a few switches, links, has hosts. The parameter framework is then assessed by the assessment matrix; throughput, RTT (relay time trip), setup and teardown. The testbed is constructed via the usage of Mininet pseudorandom topology generator script to produce a mesh type topology with a number of parameter.

The test set is dictated by the maximum number of switches which is 25. When the quantity of switches arrive at 25, the host is increased by 10 and the following test set is led. The links is constantly augmented by 15 as indicated by the incrementation of the switch. The test for every method is viewed as finished once the number hosts arrive at 50. It is to be noticed that the experiment sets 1 Gigabytes or 1000 Megabytes of the total number of bandwidth. This number was intentionally selected to enable probability of changes to the conduct of the transfer speed with connection to the throughput. Transfer speed or bandwidth can be viewed as the total limit that a link can carry starting from one point onto the next while throughput is the actual limit of successful completion starting with one point then onto the next. This experiment required a virtual partition of network to keep a host from pinging to every random set of hosts. This can be accomplished by making a network situation that have a specific number of hosts in their particular tenants. The tenant is suppeosedly to act as a detachment between the alloted host on assigned tenant by utilizing the method called network slicing by means of ONOS. Notwithstanding, this experiment did not feature a network slicing condition because of the host able to ping each other regardless of host conseqeuently bereft of tenant-specifics. It is mandatory that the future experiment performed the capacity as expected for proper analysis.

## B. Data Collection

All mininet contents are set under one explicit folder for simpler execution. The scripts are name-dependent using the order of parameter matrixnumbers acquired from the pseudorandom topology generator, for example 5-10-10.py (reactiveforwarding strategy for a topology with 5 switches, 10 links and 10 hosts) and i5-10-10.py (intent technique for topology with 5 switches, 10 links and 10 hosts). Every one of the script contained a number of tests purposely for the experiment. The first test is acquired the setup time of the topology. The setup time is taken using the summation of all finished RTT during the dumping procedure of host connectiononce the switches performed its complete initialization. The summation of this RTT are later averaged. The procedure of network connection is performed via the ping-all-full command. This stage which is required to decide the average RTT of the network through mininet utilized an estimation equation as follow:

$$F^{RTT} = \sum(Avr_{RTT}/N^{RTT}) \square \qquad (1)$$

Where the summation of the average of RTT sample is partitioned by the total number of RTT rate. The



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

result of the last RTT is then averaged. The following test is performed to decide the throughput of the data transfer capacity. This is finished by utilizing the iperf command from the mininet that indicate two hosts in which one turn as a server while the other turn as the client. The equation for figuring the throughput is:

$$\sum(Throughput \lessapprox RWIN/RTT) \quad (2)$$

Where the RWIN signify the TCP Received Windows to be isolated by RTT. The TCP window size sent by the iperf device might be shifted based on the condition of the network resource utilization. This is on the grounds that the experiment is performed under the consistent flood of pinging and numerous data packet move from each host inside the network. The scope of TCP window size is between 2.50 Mbytes - 85.3 Kbytes. The outcome from this is averaged. Each test is performed multiple times to acquire the final average. The test is performed consecutively by following the expanding number of the host by utilizing one single command, for example 5-10-20, 10-25-20… 5-10-30, 10-25-30, etc.

The experiment began with reactive forwarding techniques. The mininet is required to be in clean state for each arrangement of tests to free up the port by utilizing the mininet command mc clean. The intent test is started once the reactive forwarding experiment is finished. Intents are then being appointed through ONOS by means of ssh via utilizing python scripts to produce host-based intents. The command utilized is akin to the reactive forwarding with slight abbreviation regarding the shortening of the python named class. The result of every experiment is dumped into a text records under one folder as referenced earlier. The outcome is then recompiled into Excel sheet and the information is arranged into categorical matrix based on the assessment framework. All final averages are determined in Excel and organized.

Table 2 below demonstrates a bit of organized data procured from the reactive forwarding experiment. The initial five column of the table signify to its parameter grid which is comprised of switch, links and host. The first column demonstrates a few switches with a beginning number of 5 and a maximum number of 25 with the cycle of five in the middle. The subsequent column demonstrates the quantity of links required for the arrangement of mesh-type topology. The number is chosen dependent on the accomplishment of the pseudorandom topology generator to produce a semi-mesh type topology. From the test, the base number for the mesh topology to be created effectively is 25. The first row of the column which began at 5 is depend on the quantity of switches whereas the first row of the subsequent column which began at 10 represent to the quantity of links. The maximums number of the links to be produced is 70 with the cycle of 25. The third column demonstrates the quantity of hosts utilized in the experiment which is 10. The test began with 10 hosts and stayed consistent all through the test until the first column scopes to number 25. When the number reached, the hosts is iterated with 10 while the first and second column reset back to its underlying number which is 5 and 10 respectively. The emphasis number continued as before all through experimentation. The tabulation procedure for intent forwarding is akin.

**TABLE II. Snippet Of Reactive Forwarding Result**

| *Switch* | *Link* | *Host* | *Throughput(mbps)* | *RTT(ms)* | *Setup(ms)* | *Teardown(sec)* |
|---|---|---|---|---|---|---|
| 5 | 10 | 10 | 423.78 | 0.215 | 6.49 | 561.378 |
| 10 | 25 | 10 | 444.89 | 0.246 | 1.96 | 555.519 |
| 15 | 40 | 10 | 465.77 | 0.239 | 16.82 | 560.919 |
| 20 | 55 | 10 | 465.14 | 0.242 | 2.24 | 560.919 |
| 25 | 70 | 10 | 464.68 | 0.245 | 12.23 | 565.57 |
| 5 | 10 | 20 | 458.27 | 0.133 | 29.99 | 590.214 |
| 10 | 25 | 20 | 456.82 | 0.190 | 21.99 | 573.575 |
| 15 | 40 | 20 | 447.28 | 0.299 | 28.96 | 642.413 |
| 20 | 55 | 20 | 439.71 | 0.266 | 8.301 | 572.494 |
| 25 | 70 | 20 | 482.68 | 0.277 | 15.02 | 573.57 |



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

## IV. RESULTS AND DISCUSSION

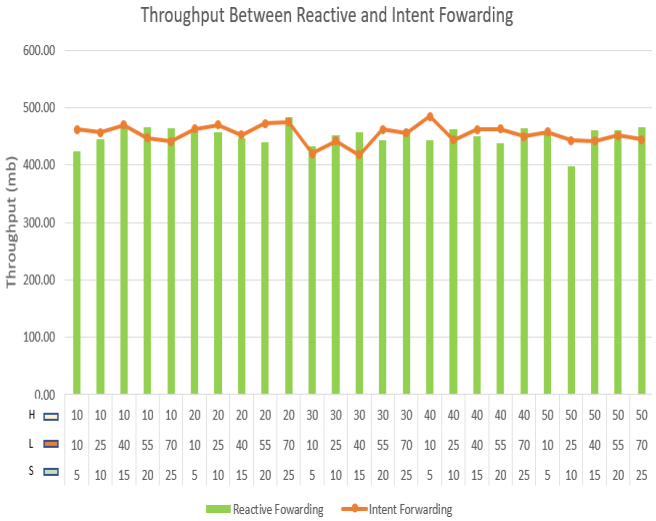


**Fig. 4. The comparison between reactive and intent forwarding in term of throughput (mb)**

In the Fig. 4 over, the throughput of reactive forwarding (RF) and intentforwarding (IF) are being analysed. The rightaxis (vertical) signifies the throughput in megabytes persecond (mbps), while the left axis (horizontal) signifies to the quantity of switches, links and hosts with the capitalized letter of S, L and H individually. In light of the figure over, the chart shows that the throughput of IF yielded the maximumtotal of attainable throughput which is 484.48mbps out of 1000 Megabytes per seconds with the combo number of 10-25-40 IF. The pattern demonstrates that at whatever point the two techniques are being comparedwithin the spectrum of predefined number of S, L and H respectively, IF and RF is by all accounts consistently neck-in-neck in accomplishing highest conceivable throughput. The distinctions of margin between the two techniques seems, by all accounts, to be miniscule. This pattern is unswerving all throughout the length of the experiment even with an exceptionally miniscule distinction for the maximum throughput as appeared by the 10-25-40 IF which is 484.48mbps against 5-10-40 RF of 482.68mbps.

As the analysis for each SLH rehashed multiple times before the total summationof averages for the throughput is taken, this distinction showed by the diagram could be further augment with increasingly repeatable test. The lowest feasible throughput is showed by 15-40-30RF where it oversaw 397.50mbps contrasted with 417.82 mbps of 10-2510 IF. This appears to demonstrate a rationale behavior in connection to the quantity of hosts and the quantity of switches. Consistently, 10-25-10 IF can accomplish higher maximum throughput compared with 15-4030RF as it has minimal number of hosts which is 10 contrasted with 30 when appended to 10 switches of equivalent number instead of 30 hosts connected to 15 switches resulting to the performance of IF hosts to be lesser than RF. It is a similar likelihood between 20-55-50RF and 10-25-40RF compared with 15-40-40 IF and 5-10-20 IF with a comparable distinction in term of quantities of hosts and switches; acquiring better throughput for a lesser number of hosts regardless of whether the fluctuation is in two decimal spots. Nonetheless, the estimatedbehaviour does not show on 5-10-40 RF and 10-25-40 IF. It appears that for this situation, the lower the quantity of switches; for example 5 and 10, the better the exhibition. Besides, it creates the impression that the number 40 of hosts is by all accounts the integral factor as both offer comparative attribute. The main clarification that can be derived from this is the abrupt burst of packets in a smallergroup of switches may influence the behaviour of throughput performances. This clarification can be upheld by taking a gander at the quantity of switches for the lowest measured of throughput in 5-10-30 RF. Thus, 5-10-30 IF likewise show comparable behavior as in RF. For RF, the average measurement of feasible throughput is of 451.59mbps whereas for IF is 453.86mbps. This demonstrate IF throughput by average is better.



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

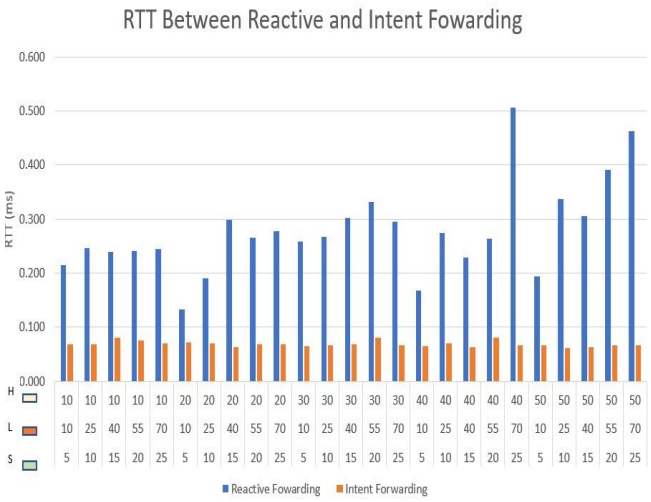


**Fig. 5. The comparison between reactive and intent forwarding in term of RTT (ms)**

Fig. 5 above demonstrates the RTT (Round-trip defer time) of reactive forwarding (RF) and intents forwarding (IF). The rightaxis (vertical) signifies the RTT in milliseconds (ms), while the left axis (horizontal) represents the quantity of switches, connections and hosts with the relegated capitalized letter of S, L and H respectively. RTT is the term used to portray the summation time it takes for a packet to be sent and be recognize that the packet has been received. Thus, it is accepted that the lesser the RTT, the better.

In view of Fig. 5 above, it is demonstrated that RF yielded the highest RTT with 0.506 milliseconds for 25-70-40 RF pursued by 0.463 milliseconds from 25-70-50 RF. The outcome obtained from IF notwithstanding, are essentially lower where even the highest RTT for IF is beneath 0.1 milliseconds. The highest RTT for IF is 0.081 milliseconds under 20-55-40 IF. From the Fig. 5 above, the pattern of RTT for RF is seemingly sporadic yet maintained an increasing manner of sets for each incremented number of hosts. There are some dip of RTT recorded from the RF where the most significantdip happened once the quantity of switches is reset at 5. Furthermore, the dip began to demonstrate its substantial when the number of hosts began to arrive at 40 or more. Along these lines, when the switches reset back to the lowest number, the dip occurred. The pace of the dip is conversely corresponding to the quantity of hosts. Then again, the RTT for IF are majorly distinct when comparedto the RTT of RF. As the outcome demonstrated, the RTT of IF are within 0.1 milliseconds. This demonstrates an significantperformance in term of response time.

The purpose behind this is on the grounds that in RF, the broadcast address is used to send the packet. This effect the likelihood for the packet to be duplicated as each deviceexposed in the network are attempting to tune in. That is the reason the bigger the number of destinationsneeded to be communicated, the higher the quantity of identical packets to be delivered. This behaviour is seen in RF by utilizing Wireshark where the duplication of packet is regularly occurring. This behaviour fundamentally influences the RTT because of the sifting of the copied packets. In addition, in the experiment, the duplicatedpackets for the incrementing number of hosts capable to reach up to thousands; facilitating the influence of RTT for RF. The inverse is seen in IF where the plausibility of repetition of packet to happen are nonexistence. The cause for this result is on the grounds that the intents are constantly being distinguished by the application that presented the intent; for example hosts and its specificintent id,generated during the making of the network topology. After the intents are put together by the application, it will be sent instantaneously into a compiling stage before starting the installation stage to accomplish the installed state. That is the reason behind a concise snapshot of time is consumed during the primary initialisation of network topology as the intent are being aggregated and installed. When the intents are in its installedstate, the packet motioning between its destination for the RTT is prompt, adding to the outcome in Fig. 5 above.



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

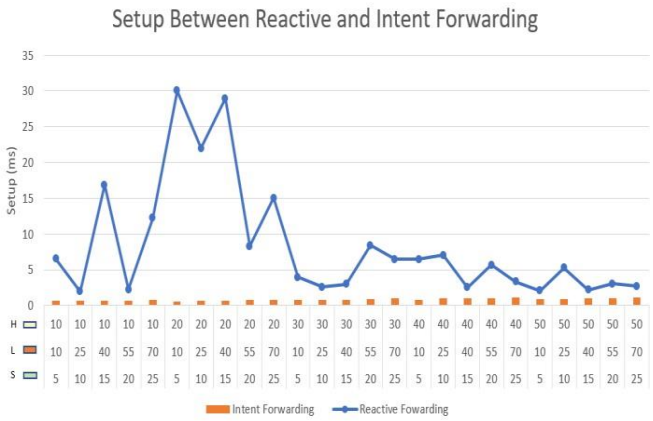


**Fig. 6. The comparison between reactive and intent forwarding in term of setup (ms)**

Fig. 6 above demonstrates the setup time (sec) for reactive forwarding (RF) and intentforwarding (IF). The left vertical axis speaks to the setup time in seconds while the horizontalaxis speaks to the quantity of switches, links and host; appointed each with the capitalized letter of S, L, H individually. For this experiment, the setup time is taken to gauge the pace of the network topology being made based on the impact of number of parameters; switches, links and hosts. The setup time is important to be taken into the experiment on the grounds that for each performing network topology, the setup time must be quick with the goal that the scalability of the network can be improved. The closer the time of setup with 0.00 milliseconds the better. The outcome would mean the adjustment of the network or the setup of the topology can be completed without affecting the downtime of the network framework.

In view of the Fig. 6, the setuptime for RF demonstrates a vacillated pattern with few spikes as the number of hosts began to increment by 5. The diagram demonstrates that the highest for RF is 29.994 milliseconds in 5-10-20 RF. The spike demonstrated its peak when the number of hosts are at around 20 and 15 - 25 for switches. The number of switches appears to impact the graph just as the switches with the quantity of 15 and 25 appears to constantly linked with the highestspike for everyset of switches number. The setup time for IF is immensely difference to that of RF. The figure for IF shows constant pattern of setup time with the lowest conceivable value. The lowest recorded time for IF is 0.602 milliseconds as appeared on Fig. 6. IF setup time instead continually incremented between the scope of 0.6 to 1.0 milliseconds with the highest recorded spike of 1.098 milliseconds. The purpose behind the significant diverge from respects to the arrangement time of RF and IF is because of a similar reason to that expressed in RTT.

In RF, the device utilized broadcast address to listen each other different devicesaffecting to the plausibility of duplicated packets. When the duplication of packet happened, the gadgets will continue sending the duplicatedpackets until an appropriate association from the correspondence of the targeteddevice is found. The forwarding of the duplicatedpackets normally exceeded to thousands beforeobtaining the intendedpacket. The behaviour was seen by utilizing Wireshark during the analysis with RTT and a similar behaviour is reflected during the setup time experiment. This consequently would prompt extreme expanded of setup time as the underlying point for a setup to be considered as substantial is the point at which a legitimate packet is being received by both intendeddevices to form the firstcommunication. In IF by contrast, the duplicated packets are nonexistence because of the consistency of the intents to recognize its network assets. For example, hosts and its intent id; as referenced during the RTT experimentation. Strictly, the setup time for IF is much longer contrasted with RF if the procedure for the setup is taken for deliberation to the point of the first execution of the topology scripting. This is on the grounds that before the correspondence of the devices happens, the intents should be compiled and installed which would take additional time contrasted with that of distinctive RF technique. Instead, the state of setup time taken in this experiment is logged during its first unhingedcommunication between the device.



Published by: The Mattingley Publishing Co., Inc.

March - April 2020
ISSN: 0193-4120 Page No. 8205 - 8214

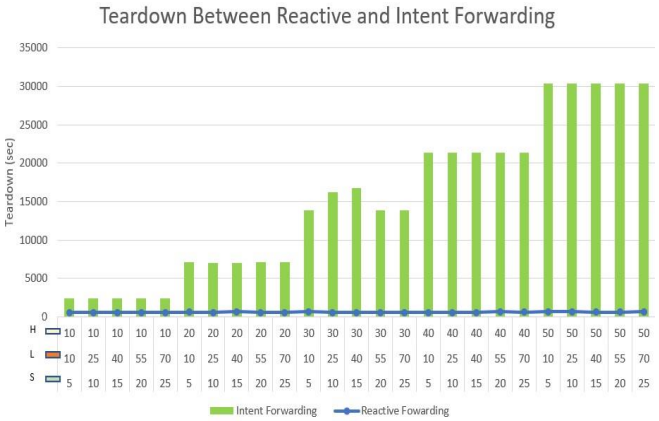


**Fig. 7. The comparison between reactive and intent forwarding in term of teardown (sec)**

Fig. 7 above demonstrates the teardown time (sec) for reactive forwarding (RF) and intent forwarding (IF). The left vertical axissignifies the teardown time in seconds while the horizontal axissignifies the quantity of switches, links and hosts; doled out each with the capitalized letter of S, L, H individually. For this experiment, the teardown time is taken to determine how much of time consumption for the semi mesh topology constructed by the script being dissimulateduring each methods and the relationship of the setup and teardown concerning the adaptability of network infrastructure.

In view of the Fig. 7, the teardown time for RF demonstrates a relatively steady pattern with minor spikes on two occasions during the test. The teardown time appears in the graph continually incremented at the scope of 555-680 seconds during the period of experimentation for RF while spikes happened at 15-40-20RF and 5-1030RF with the estimation of 642 seconds and 643 seconds individually. From the figure above, it demonstrates that the scope of teardown time for the devices with the incrementing number of hosts from 10 to 50 included a gradual estimation of 10 seconds. The abrupt event of the spike is seemingly due to the unforeseen utilization of resources in mininet running on Ubuntu 18.04 causing the mininet to end up inert for a period which enough to influence the teardown time. The RTT and consequently the setuptime may likewise add to a portion of the spikes happened on RF. Then again, the teardown time for IF is altogether higher contrasted with RF.

In view of the figure above, the teardown of IF demonstrates an upward pattern with consistent development. The teardown time is seemingly relative with the quantity of hosts. As the quantity of hosts began to increment by 10, the teardown began to augment 5000, 6000, 7000 and 8000 seconds in regard to the expanding set of hosts. Conversely, between the change of the quantity of hosts, the teardown time of IF seeminglypersisted in fixed state aside from when the host number is at 30 which can be consequential from the unstable behaviour of mininet. This behaviour of teardown is to be expected from IF as the intents during the initialisation of mesh topology required to be installed and the procedure consumes times. This consequently influenced the teardown. When the intentsin its installed state, the teardown is consistently at steady rate because of the instantaneous reaction given by the intents from the devices during the setup. Thus as a result, despite the teardown of IF is slower in contrast to RF, the IF's scalability potential is manifested. The IF has the attribute of perhaps enabling a network to be scaled out quicker which is a requirement for telecom suppliers.

## V. ACKNOWLEDGMENT

This study is bolstered by Universiti Kuala Lumpur Center of Research and Innovation (CoRI). We thank our honourcolleagues for giving the genuinely understanding and aptitude which tremendously helped the researchdespite the distinctive construal may have on this paper. We express gratitude toward Megat, Dr from UniKL MIIT for the help with the procedure and testbed plan and Fuead Ali for the remarks that consequently improved the paper.

We might likewise want to demonstrate our appreciation to the postgraduates of IoTSmartXMIIT for imparting their point of view to us over the span of this manuscript.

*March - April 2020*
*ISSN: 0193-4120 Page No. 8205 - 8214*

**REFERENCES**

[1]. Yin, C., Kuo, T. C., Li, T. Y., Chang, M. C., and Liao, B. H. (2014). Mediating between OpenFlow and legacy transport networks for bandwidth on-demand Services. APNOMS 2014 - 16th Asia-Pacific Network Operations and Management Symposium,6996529.https://doi.org/10.1109/APNOMS.2014.6996529 https://doi.org/10.1145/1721654.1721672

[2]. Hakiri, A., Berthou, P., Gokhale, A., \&Abdellatif, S. (2015). Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications. IEEE Communications Magazine,53(9),48–54. https://doi.org/10.1109/MCOM.2015.7263372

[3]. Benzekki, K., El Fergougui, A., \&ElbelrhitiElalaoui, A. (2016). Software-defined networking (SDN): a survey. Security and Communication Networks, 9(18), 5803–5833. https://doi.org/10.1002/sec.1737

[4]. Simon, A. S. (2018). Bandwidth-on-Demand Motion Control, 26(1), 265–273

[5]. Omar, N. (2017). Transport- Software Defined Network T-SDN Specification Requirement for Bandwidth on Demand Service. TMRND, (Julai).

[6]. Sadasivarao, A., Syed, S., Pan, P., Liou, C., Monga, I., Guok, C., \& Lake, A. (2013). Bursting data between data centers: Case for transport SDN. Proceedings - IEEE 21st Annual Symposium on High-Performance Interconnects, HOTI 2013, 87–90. https://doi.org/10.1109/HOTI.2013.20

[7]. Thyagaturu, A. S., Mercian, A., McGarry, M. P., Reisslein, M., \&Kellerer, W. (2016). Software Defined Optical Networks (SDONs): A Comprehensive Survey. IEEE Communications Surveys and Tutorials (Vol. 18). https://doi.org/10.1109/COMST.2016.2586999

[8]. Benzekki, K., Fergougui, A. El, El, A., \& El, B. (2016). A Secure Cloud Computing Architecture Using Homomorphic Encryption, 7(2), 293–298.

[9]. Mendiola, A., Astorga, J., Ortiz, J., Vuleta-Radoiĉić, J., Juszczyk, A., Stamos, K., … Higuero, M. (2017). Towards an SDN-based bandwidth on demand service for the European research community. 2017 International Conference on Networked Systems, NetSys 2017. https://doi.org/10.1109/NetSys.2017.7903965

[10]. Zhou, W., Li, L., \& Chou, W. (2014). SDN Northbound REST API with Efficient Caches. Web Services (ICWS), 2014 IEEE International Conference On.

[11]. Knoll, T. M. (2014). A combined CAPEX and OPEX cost model for LTE networks. 2014 16th International Telecommunications Network Strategy and Planning Symposium, Networks 2014. https://doi.org/10.1109/NETWKS.2014.6958531

[12]. Yassine, A., Shirehjini, A. A. N., \&Shirmohammadi, S. (2016). Bandwidth On-demand for Multimedia Big Data Transfer across Geo-Distributed Cloud Data Centers. IEEE Transactions on Cloud Computing, X(X), 1. https://doi.org/10.1109/TCC.2016.2617369

[13]. OnosProject.org. (2017). ONOS - A new carrier-grade SDN network operating system designed for high availability, performance, scale-out. Retrieved from https://onosproject.org



*Published by: The Mattingley Publishing Co., Inc.*