Intent-Based Path Selection for VM Migration Application with Open Network Operating System

Dimas A. Marenda[1], Galura M. Suranegara[2], Aris C. Risdianto[3], Rifqy Hakimi[4], Eueung Mulyana[5]
School of Electrical Engineering and Informatics, Institut Teknologi Bandung
Email : dimasagil@students.itb.ac.id[1], galurams@students.itb.ac.id[2], aris@onos-ambassador.org[3]
rifqyhakimi@stei.itb.ac.id[4], eueung@stei.itb.ac.id[5]

Abstract— **Application of virtual machine (VM) migration is one of the key applications in the management of a data center. The migration task is required for several reasons such as load balancing, maintenance, or power management. Migration task need to be optimized by improving networking (e.g. end-to-end path) between VM hypervisors. Migrating VM on the best path, will certainly give a smaller migration time. Intent (i.e., intent networking) is used as a basis in path selection on the SDN to identify better networking performance (i.e., higher bandwidth). The migration measurement is taken for three parameters which are migration time, downtime, and overhead time. The measurement result shows an improvement for three different bandwidth variation.**

Keywords: Intent, path-selection, migration, ONOS

I. INTRODUCTION

The use of virtual machine migration application has become the common sense in the management of a data center. A virtual machine needs to be shifted to another location for several reasons. In a data center, a server may contain a number of virtual machines close to its maximum capacity. While on the other hand, there are servers still not fully loaded. In this condition, some virtual machines deserve to be migrated so that each server load can be balanced [7].

In other cases, a virtual machine needs to be relocated for regular maintenance reasons [6]. After the maintenance process is complete, then the virtual machine is returned to its original location. Some data center service providers even apply virtual machine migration to other data centers that are always in the state of the night to save electricity usage [11].

The whole process of migration requires a short time. So this migration process can be optimized to shorten the time through path selection process. Paths that can be passed in the data transmission during migration can be various both hop and bandwidth performance. The success of using a path with good performance will certainly improve the data delivery time.

The process of path selection is then the main topic in this research. Intent utilization which is a high level policy on SDN is used as a basis in path selection process. This is because the character of the network on the SDN through its abstraction provides the ease in viewing network components globally. In case of transferring data, SDN utilizes two concepts which are reactive and proactive forwarding. Intent-based path selection works technically as proactive forwarding to neglect the limitation of reactive forwarding so it can effectively use for productive data center.

This paper is organized as follows: Section II describes literature survey and related works in intent-based application; Section III outlines the design of the aplication; Section IV interprets the simulation results; Section V concludes the paper.

II. LITERATURE SURVEY

A. Virtual Machine Migration

Virtual machine migration is the process of migrating a virtual machine to a new location for several reasons in order to manage a data center. In general, migration is divided into two, namely live and non-live migration [1]. Live migration is a migration process that has very little downtime or even close to zero so the services of a virtual machine need not be stopped [8].

Different from live migration, non-live migration requires that the virtual machine is turned off before the migration process starts. Furthermore, the non-live migration process does not require shared storage so that the transmission process can be done immediately. In addition, the migration non-live migration mechanism can still be done on two virtual machines despite having a different processor architecture.

Each mechanism has specific advantages and uses. Non-live migration is commonly used on applications that support user mobility [1]. A virtual machine is switched off before the user switches, then it will be re-enabled on the new location.

B. SDN with ONOS

Software Define Networking is a new frame of mind in network management. Network is managed as three main layers, namely infrastructure, control, and applications. The infrastructure layer is a layer that performs the data forwarding function. One of the most commonly used software to see its performance is open vswitch.

The abstraction process performed on the SDN separates the control plane from its forwarding plane. Control plane works like an operating system on a computer. In traditional networks, the devices used put these two layers in a device. With this abstraction then the complexity of the network can be seen to be simpler.

The functions and services desire to build on a network are further realized through applications built over the control plane. With this mechanism the network becomes a programmable entity. Applications built to be very diverse as well as specific according to the desired needs.

978-1-5386-6163-5/18/$31.00 ©2018 IEEE

For the communication needs between layers then designed some interfaces. The protocol used for communication between control plane and forwarding plane is openflow [10]. Openflow provides the provision of data splitting mechanism as well as its communication model. An incoming data packet to the forwarding plane will be checked first in accordance with the field on the open flow before the action is applied on the package.

Communication between control plane and application is built using REST API [9]. This interface provides information about the network managed by the control plane. Thus users can access and view the network simpler.

Open Network Operating System (ONOS) is one network operating system that is open-source. ONOS provides information about the network through the REST API provided. Some of the APIs used are Host, Device, Link, Path, and Intent which each have specific network related information.

C. Related Work

Intent is a top-level policy that facilitates vision in network utilization. Intent is a readable policy that can be understood by the user easily. In ONOS, the intent is a customizable API. Commands received by intent will be translated by intent compiler into a set of forwarding rules that can be used to support the application designed.

Some utilization of intent-based networking is in the issue of network virtualization based on SDN [4]. In this scenario, the intent will communicate with the orchestrator network so as to form multiple virtual networks that are dedicated to the same physical infrastructure. This automation can be done by intent so that virtual network management can be done on a global level in an easy way. Previously, virtual network management and configuration such as protocol, address, and control rules were performed on each virtual network so that it took longer time.

Intent utilization also extends to security issues. The more demand for an encrypted data to support network security, the intent provides one alternative solution. In [2] intents are used in encryption processes called in-flight encryption. In this mechanism, encryption is performed on the traffic between a trusted site on the media layer (IP / MAC / PHY) through the ACINO project.

A data that wants to be encrypted will give effect to service request. Requests with different types and levels of encryption will surely add complexity to the network because the configuration carried out on the traffic is manual. ACINO attempts to automate its configuration so that weaknesses are slow service and error rate can be overcome.

In other scenarios the intent is also used in application-centric networks [3]. In this scenario it is known that an application basically has a network-specific request service. This can be bandwidth, latency, and also availability is quite diverse. Intent is used to negotiate between applications and networks. At certain levels when the SR cannot be met, then some SR options are given so the application can still run.

In this study, intent is used in the path selection task just before migration is done. Path selection is useful to provide a choice of paths that can be used so that it can improve the efficiency of data transmission.

III. DESIGN AND IMPLEMENTATION

A. Application Design

This application is designed with three functions running on it, namely main, check, and excecution function. The main function is the first display of the application that is used to find out information regarding the VM to be moved. The information needed includes the HostID in the form of a mac address and also information regarding the VM list contained in a host.

Check function is used to provide information regarding the path to be passed from a source host to its destination. Technically this module will provide a maximum of two paths that consist of the switchID and the port that is traversed. The next stage is the path selection process itself which will end with the installation of intent on the selected path to be used.

The excecution function as the name implies will run the VM migration function from the specified host. At the time of execution, the system will simultaneously calculate the transmission time used in the migration process of a selected VM.

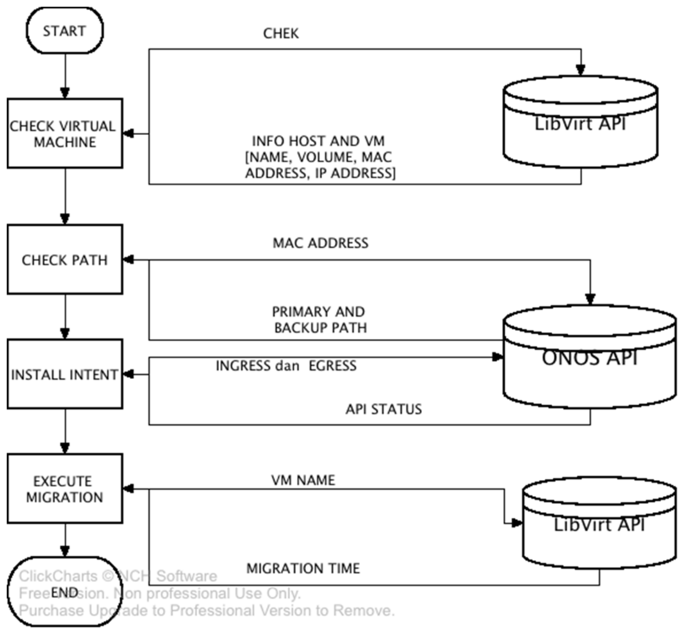


Fig1. Global view of the application vm migration

B. VM Migration Implementation

Migrating VM on a physical machine is actually a complex process. In this study the migration process was carried out using KVM QEMU. KVM QEMU provides an API that can be used to carry out the migration process, namely LibVirt. The scheme used for its management is Hypervisor Native Transport. This mechanism provides benefits because it has minimal computational cost character due to effectiveness in the amount of data copied.

LibVirt API is able to see a number of VMs that are located on a host and is able to provide the required information. By using LibVirt, volume of each VM that will be moved including the memory contained in a VM can easily be informed. This information will then be used as consideration for the migration process from the source host to the destination.

Technically, the VM migration process is done by modifying flag that will give effect to the domain remaining when the migration process is carried out. The next stage, it is necessary to ensure that the VM has moved to the desired destination host. Furthermore, the VM that has been successfully moved will be turned on again as the deadline for calculating the migration time. This process is a major part of the excecution function

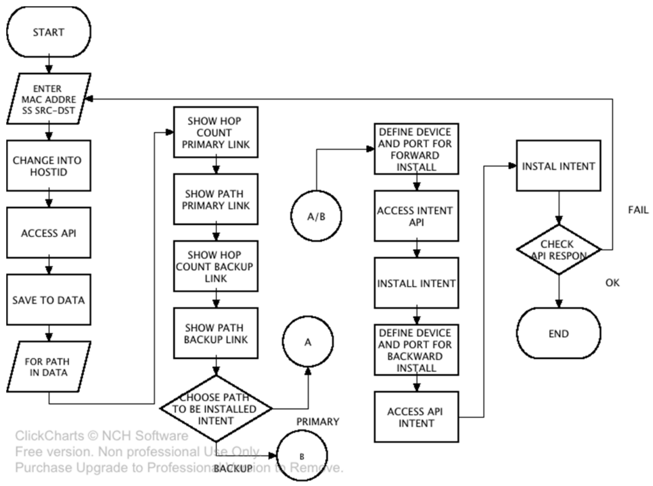
C. *SDN Path Selection Implementation*

Task of path selection is done by utilizing two APIs provided by ONOS. Both APIs are disjoint path and intent. As the name implies, the disjoint path relates to the path that can be used while intent is used to prepare the path so that it can be used in the data transmission process.

PATH is an API used to display a usable path from a source host to a destination. This means that PATH will display information about the number of hops and links it passes. There are two types of APIs that can be used, namely PATH and disjoint PATH. The difference between them lies only in the number of paths that can be shown to the user. In this application used disjoint path so that users get information about two paths that are useful in the selection process.

Determination of two paths that can be used basically based on hop count. The path shown will always provide the path with the two least hop count. Therefore, the two available paths need to be selected for use in the data transmission process. The next selection is done by considering the bandwidth on each path. Paths that have better bandwidth will then be installed intent so that the path can be used.

The last phase in the path selection process is the intent installation on the selected path. Intent is a policy that can be translated into a set of forwarding rules. Technically this forwarding rules will add flow to the flow table on each open vswitch.


START
ENTER MAC ADDRESS SRC-DST
CHANGE INTO HOSTID
ACCESS API
SAVE TO DATA
FOR PATH IN DATA
SHOW HOP COUNT PRIMARY LINK
SHOW PATH PRIMARY LINK
SHOW HOP COUNT BACKUP LINK
SHOW PATH BACKUP LINK
CHOOSE PATH TO BE INSTALLED INTENT
PRIMARY
BACKUP
A
B
A/B
DEFINE DEVICE AND PORT FOR FORWARD INSTALL
ACCESS INTENT API
INSTALL INTENT
DEFINE DEVICE AND PORT FOR BACKWARD INSTALL
ACCESS API INTENT
INSTAL INTENT
CHECK API RESPON
FAIL
OK
END
ClickCharts © NCH Software
Free version. Non professional Use Only
Purchase Upgrade to Professional Version to Remove.
Fig2. Flowchart of the path selection

Path selection process begins by entering mac host source and destination information. This information then needs to be converted to hostID by adding a VLAN variable. This process will be executed to access the PATH disjoint API whose data is still formatted JSON. This data can be parsed to be displayed in the user interface readable by the user.

The path information contains the number of hop counts and the links that make up each path. The links on each path contain information about the number of open vswicth involved and the port details used as ingoing and outgoing ports.

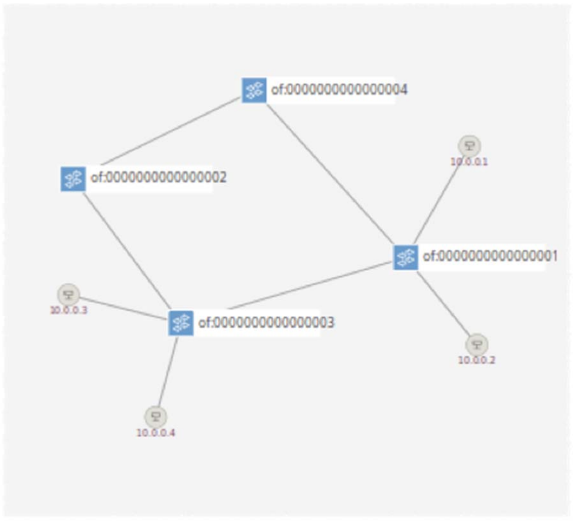
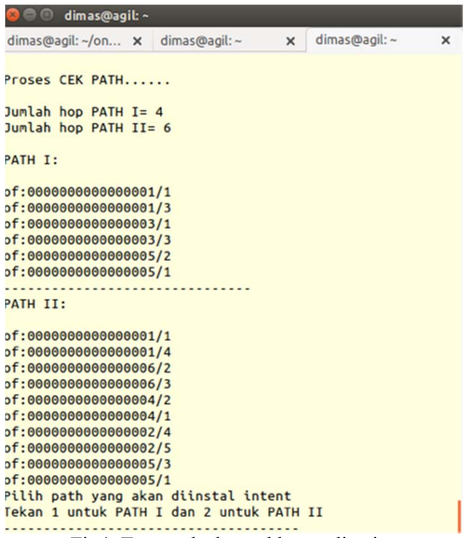


Fig3. Topology used for first test

In the test topology, path which has a larger hop count will be set to have better bandwidth. So by this predefine information, the path with a lower hop count is not necessarily used in the transmition process. The path used in the intent mechanism will also always be on the path with a longer hop count.

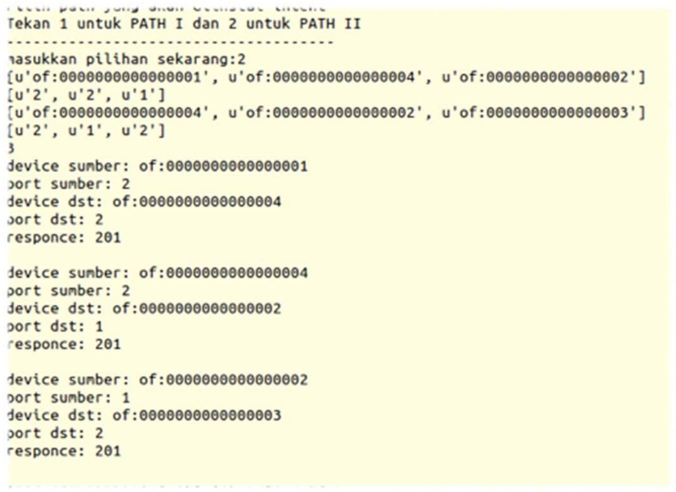


```
Proses CEK PATH......

Jumlah hop PATH I= 4
Jumlah hop PATH II= 6

PATH I:

of:0000000000000001/1
of:0000000000000001/3
of:0000000000000003/1
of:0000000000000003/3
of:0000000000000005/2
of:0000000000000005/1
------------------------------
PATH II:

of:0000000000000001/1
of:0000000000000001/4
of:0000000000000006/2
of:0000000000000006/3
of:0000000000000004/2
of:0000000000000004/1
of:0000000000000002/4
of:0000000000000002/5
of:0000000000000005/3
of:0000000000000005/1
Pilih path yang akan diinstal intent
Tekan 1 untuk PATH I dan 2 untuk PATH II
------------------------------------
```

Fig4. Two path showed by application

The next phase is to install the intent. The intent in this study has been set to have a certain priority. Priority will affect the fast slew of data packets executed by open vswitch. The greater the priority value, the execution will be prioritized. Intent is installed by involving the first port of a connected source host and the destination host's final port is located. Another variable is the outgoing port on each open vswitch constructing a path.

Intent is installed in both directions, forward and backward. Forward installation means the intent is installed with the open vswitch parameter and the port from the source host to the destination. While the backward installation is in the opposite direction.



```
Tekan 1 untuk PATH I dan 2 untuk PATH II
-------------------------------------
nasukkan pilihan sekarang:2
[u'of:0000000000000001', u'of:0000000000000004', u'of:0000000000000002']
[u'2', u'2', u'1']
[u'of:0000000000000004', u'of:0000000000000002', u'of:0000000000000003']
[u'2', u'1', u'2']
3
device sumber: of:0000000000000001
port sumber: 2
device dst: of:0000000000000004
port dst: 2
responce: 201

device sumber: of:0000000000000004
port sumber: 2
device dst: of:0000000000000002
port dst: 1
responce: 201

device sumber: of:0000000000000002
port sumber: 1
device dst: of:0000000000000003
port dst: 2
responce: 201
```

Fig5. Intent installation and its responce code

D. *Hardware Specification and Topology*

The study was conducted using ONOS 1.12.0 (Magpie) as a network controller. Furthermore, the virtual machine is built using KVM / QEMU which consists of two types, namely Host1 and Host2. Host1 will be the source while Host2 will act as destination. Host1 contains a 5GB virtual machine that is transmitted to Host2.

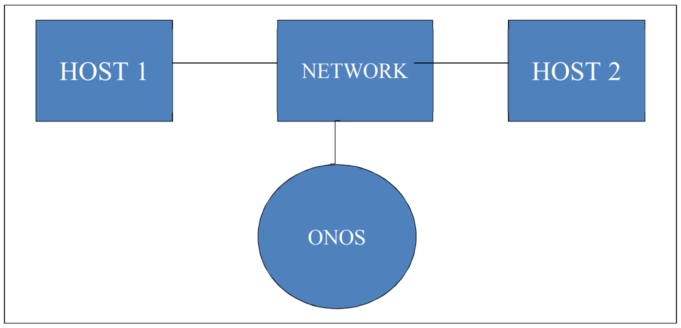


Fig6. Server configuration to run application

To run this scenario at least 4 virtual machines (VM) are used on a server. The first VM is used to run ONOS. ONOS runs on a VM with 4 processors and an 8192 MB memory allocation. Host1 and Host2 run on the same VM specification each with 8 processors and 16384 MB of memory allocation.

The VM that runs the network function contains the topology used in the test. To facilitate the testing process with a good topology, then use mininet 2.2.0 which is connected with Host1 and Host2 and ONOS as network controller.

The topology used in this test consists of 3 open vswitch, sw1, sw2, and sw3. Sw1 is connected to Host1 while sw3 is connected to Host2. In this test Host1 has a static IP of 192.168.1.11 while Host2 is 192.168.1.22. The transmission process is done from Host1 to Host2.

Bandwidth set on sw1-sw3 link half times than bandwidth on sw1-sw2-sw3. This is done to support the process of path selection as described above. Furthermore, sw1-sw2-sw3 bandwidth limit bandwidth is always changed from 200 Mbps, 800 Mbps, and 1000 Mbps.

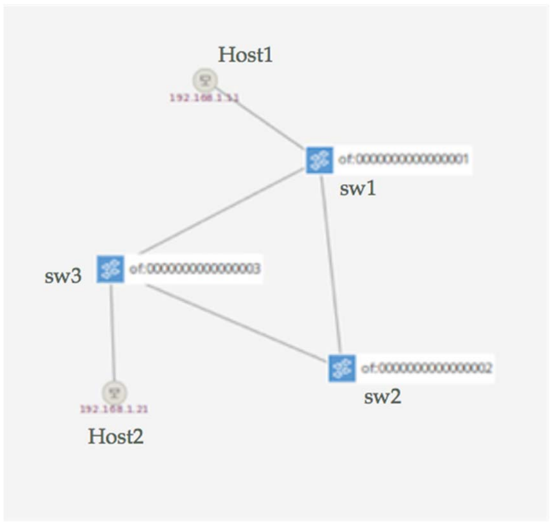


Fig7. Topology of the network used for migration

To verify the performance of the application being designed, it is tested against the final results of the migration process. The test is done by moving a VM with a volume of 5 GB to the topology above to see the suitability of the path used and the time of migration needed.

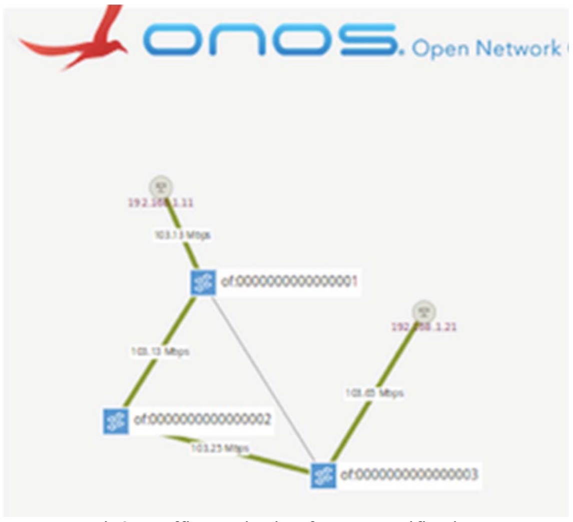
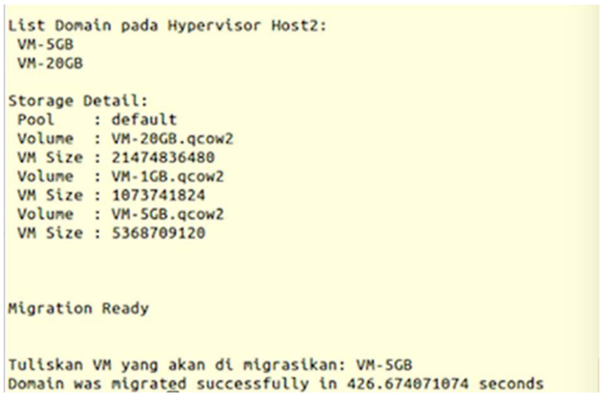


Fig8. Traffic monitoring for apps verification

Based on Fig8 it is known that data traffic during the migration process is passed to a path that has a longer hop count in accordance with the intent usage scenario. This can be seen from the traffic monitoring facilities provided by ONOS. Traffic starts from sw1, sw2, and ends with sw3 before arriving at the destination host.



```
List Domain pada Hypervisor Host2:
 VM-5GB
 VM-20GB

Storage Detail:
 Pool     : default
 Volume   : VM-20GB.qcow2
 VM Size  : 21474836480
 Volume   : VM-1GB.qcow2
 VM Size  : 1073741824
 Volume   : VM-5GB.qcow2
 VM Size  : 5368709120


Migration Ready


Tuliskan VM yang akan di migrasikan: VM-5GB
Domain was migrated successfully in 426.674071074 seconds
```

Fig9. Measurement on migration time

IV. Measurement Results

Measurements are done by migrating a virtual machine with a volume of 5 GB. The migration is done from Host1 to Host2 through a topology that allows the application to make a selection of the two available paths. Furthermore, it will be compared the fowarding mechanism used, namely reactive and proactive forwarding.

Reactive forwarding is the default mode of data transmission from a host to a destination. In this mechanism, the package management is determined by the algorithm contained in it. If a packet has a field entry corresponding to the flow table, then the packet will be immediately sent to the destination. However, if the entry is not available then the process will be delayed until the controller changes the flow table.

The addition of the flow table is closely related to the availability of a usable path. If the path has been established, then the path will be directly used on reactive forwarding mechanism. Conversely, if no path is available, then the packet will get flooding instructions. Packages will be transmitted out through all ports where a host is connected to open vswitch to reach the destination to get the response. The response will be sent back to the source host while the path used in this mechanism will be used in the reactive forwarding. For this reason the built path always has the smallest hop count.

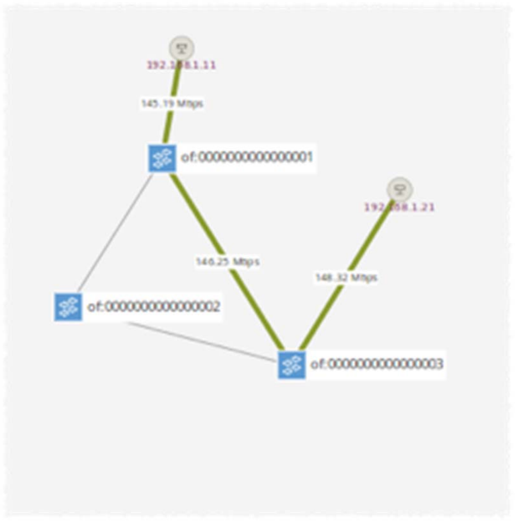


Fig10. Traffic monitoring for reactive forwarding

Clearly this mechanisme will introduce greater time delay on transmission phase. The disadvantage of this mechanism other than the path delay is the inability to consider the bandwidth. Despite having a slight hop count, but bandwidth performance is not necessarily the best. Therefore, another mechanism is used, namely proactive forwarding.

Proactive forwarding in the research is formed by installing the intent. Before installation is done, bandwidth is considered so that the path used has better performance despite having a bigger hop count. This mechanism is compared in the test by involving three parameters, namely migration time, downtime, and overhead.

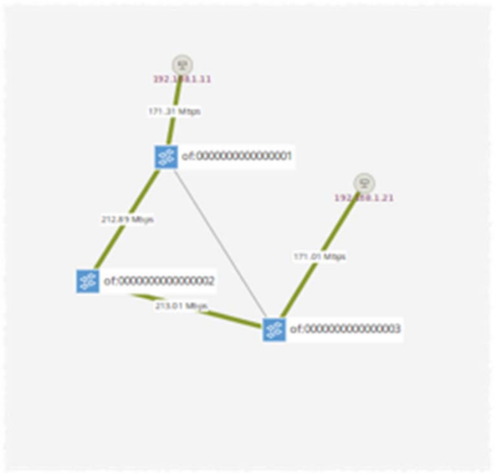


Fig11. Traffic monitoring for proactive forwarding

Migration time is the total time spent on the migration process. Migration time measurement is done in three phases, namely suspend, copy, and restart. Suspend is the time used to turn off the vitrtual machine. After the virtual machine is suspended, the virtual machine will be copied to the destination. Then in the last stage, the virtual machine will be restarted until it can be reused.

Based on the measurement results, the greater the bandwidth limit used then the required migration time will be faster. This occurs in both reactive and proactive forwarding mechanisms. Furthermore, at each migration time measurement in the intent scenario, it has a shorter time than reactive forwarding. This shows that the path selection can be used and gives effect to the migration time.

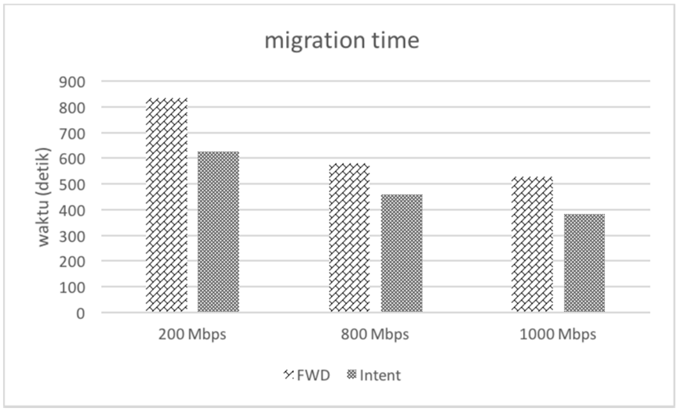


Fig12. Migration time comparison for several bandwidth

Further measurements are made to determine downtime. Downtime is the time virtual machine cannot run the service. The measurement is done after the virtual machine is turned off until the virtual machine returns to life in the destination location. Technically the mechanism is done by ping test and evaluating icmp packet leap that is not detected.

Based on the test results note that downtime is affected by the bandwidth used during the data transmission process. The greater the bandwidth used the downtime becomes shorter. This also occurs in the comparison of reactive forwarding and intent scenarios. On three different bandwidths, the intent scenario has a shorter downtime result.

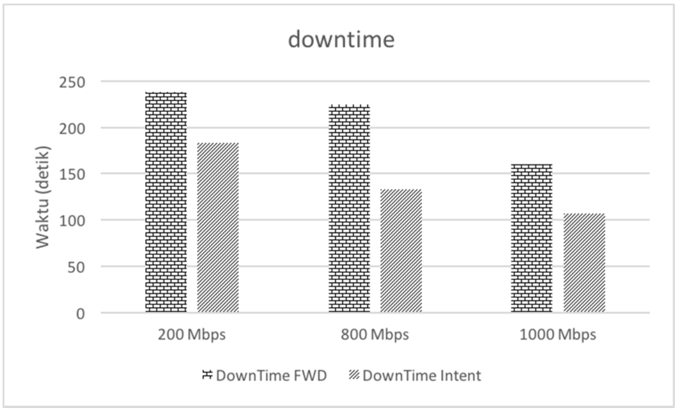


Fig13. Downtime comparison on several bandwidth

The third measurement is done to see the overhead. Overhead is the difference between migration time and transmission time. Transmission time is calculated by comparing the volume of data transmitted with available bandwidth on the path used. Thus, overhead can also be referred to as the overall processing time used during the migration.

Based on the measurement results known that the decreased overhead is directly proportional to the increase in bandwidth used in transmission. Overhead values in intent scenarios also tend to be smaller when compared to overheads in reactive forwarding scenarios.

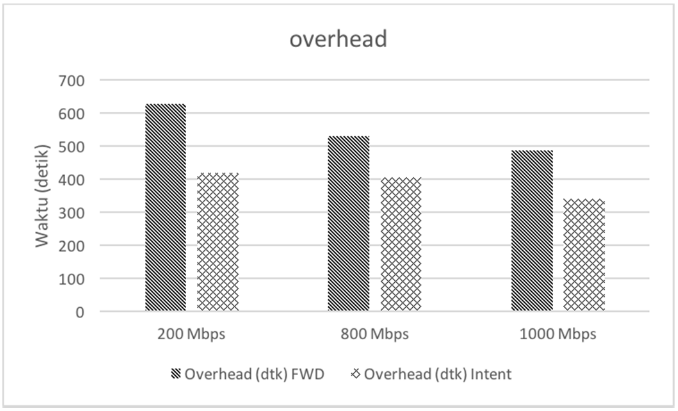


Fig14. Overhead comparison

Delay that occurs on the migration process really affect the performance of the application. By those all measurement, it is clearly informed that proactive forwarding has a better performance due to less time consumption for handling transmission process. It will get more benefit if it success to select a bigger bandwidth to be used as path for transmission.

V. Conclusion

Path selection on virtual machine migration application can be developed on SDN framework using an open source network controller that is Open Network Operating System at least with two main API both are disjoint path and intent. Path selection contribute to better the performance of the virtual machine migration for the three parameters, migration time, downtime, and overhead. It penetrates the time consuming for three parameters on the different bandwidth that are 200 Mbps, 800 Mbps, and 1000 Mbps. It benefits for average reduction 24.69 % of migration time, 32.36 % of the downtime, and 28.86 % overhead time respectively.

Future Works

The path selection process can be shortened not only take advantage of the two paths provided by the ONOS API disjoint path. Certain algorithms can be applied so that the selection process can perform more accurate bandwidth calculations rather predefine mechanism.

References

[1] Zhang F., Liu G., Fu X., Yahyapour R. "Survey on Virtual Machine Migraton: Challenges, Techniques, and Open Issues". IEEE Communication Surveys & Tutorial, VOL 20, NO.2 Second Quarter 2018.

[2] Marsico A., Santuari M., Ghafour A, Junique S., et al. "An Interactive Intent-Based Scheme for Application-Centric Network". 978-1-5090-6008-5/17 2017 IEEE

[3] Chamania M., Szyerkowiec T., Santuari M., Siracusa D., et al "Intent-Based In-Flight Service Ebcryption in Multi-Layer Transport Networks". 978-1-943580-23-1/17 2017 Optical Society of America

[4] Han Y., Li J., Hoang D., Yoo J., Hong J.W. "An-Intent Based Network Virtualization Platform for SDN". 978-3-901882-85-2 2016 IFIP

[5] C. Ge., Z Sun, N. Wang, K.Xu, J. Wu. "Energy Management in cross-domain content delivey networks: A Theoritical perspective". IEEE Transaction Network Services Management, vol 11 no.3 pp. 264-277 2014.

[6] L. Youseff, Butrico M., Da Silva D. "Towards a unified ontology of cloud computing" Grid Computing Workshop, Austin USA, 2008

[7] L. Cheng and T. Li. "Efficient data redistribution to speed up big data analytics in large systems" Proceeding IEEE 23rd International Conference High Performance Computing, 2016, pp 91-100.

[8] Kortas N., Youssef H. "Performance Evaluation of Virtual Machine Migration Planning in Software Define Networking". DOI 10.1109/TCC.2017.2710193. IEEE Transaction on Cloud Computing

[9] Zhou W., Li L., Luo M., Chuo W. "REST API Design Patter for SDN Northbound API". 978-1-4799-2652-7/14 IEEE DOI 10.1109/WAINA.2014.153

[10] Openflow Switch Spesification Versio 1.0.0. Open Networking Foundation (ONF).

[11] K. Takahashi, K. Sasada, and T. Hirofuchi. "A fast virtual machine storage migration technique using data deduplication" in Proc. Cloud Comput, 2012, pp. 57-64