ASSESSMENT OF THE IMPACT OF THE OPENFLOW PROTOCOL ON THE PERFORMANCE OF NETWORK DEVICES

DOI: 10.36724/2072-8735-2023-17-2-56-60

Denis O. Yakupov,
Povolzhskiy State University of Telecommunications and Informatics, Samara, Russia, ***d.yakupov@psuti.ru***

Sergey V. Malakhov,
Povolzhskiy State University of Telecommunications and Informatics, Samara, Russia, ***malakhov-sv@psuti.ru***

Manuscript received 11 January 2023;
Accepted 20 February 2023

Keywords: *OpenFlow protocol, performance, switch, router, network operating system, hash table, line search, packet generation, experimental installation*

A very promising area of computer network technologies is the use of Software Defined Network (SDN). The principles of SDN first emerged in the Stanford and Berkeley research laboratories and are currently being developed as part of the Open Network Foundation consortium and the European project OFELIA. The openness of the OpenFlow standard and the introduction of control logic into a separate controller simplifies the software and hardware of active network equipment, which will allow manufacturers to reduce its cost in the future, and therefore reduce the cost of creating computer networks. Therefore, research in this area is very promising. This work is devoted to the study of the performance of SDN, namely, how the implementation of the OpenFlow protocol affects the quality of data transfer between network devices. Using a single stream, the maximum throughput is estimated. To do this, three experimental installations of a network with different network devices, a switch, a router and an OpenFlow device, as well as a dedicated Linux server with support for the OpenFlow protocol, are assembled. The essence of the experiments is to change the size of packets and the size of the forwarding table to assess the performance of SDN with different network devices. The results of the analysis showed that the implementation of OpenFlow on Linux systems is able to offer very good performance. OpenFlow shows good performance with a large number of threads.

Information about authors:

Denis O. Yakupov, *Povolzhskiy State University of Telecommunications and Informatics, Assistant of the Department of Software Engineering, Samara, Russia*

Sergey V. Malakhov, *Povolzhskiy State University of Telecommunications and Informatics, Associate Professor of the Department of Software Engineering, Samara, Russia*

Для цитирования:
Якупов Д.О., Малахов С.В. Оценка влияния протокола Openflow на производительность сетевых устройств // T-Comm: Телекоммуникации и транспорт. 2023. Том 17. №2. С. 56-60.

For citation:
Yakupov D.O., Malakhov S.V. (2023). Assessment of the Impact of the Openflow protocol on the performance of network devices. *T-Comm*, vol. 17, no.2, pp. 56-60. (*in Russian*)

Introduction

The needs of enterprises set the need for networks that can adapt to constantly changing business, enterprise growth and service development. Therefore, it is important to understand the principles of what the corporate network is and how it is formed and adapted to meet the needs of society. An enterprise network is initially the interconnection of objects belonging to a specific functional group or organization, which primarily ensures the sharing of resources such as printers and file servers. Modern networks have practically exhausted their capabilities, increased routing time, and difficulties appeared in configuring the network and managing flows in it. Network virtualization and routing, mapping multiple logically isolated networks with independent resources, do not help meet today's needs. The cost of deploying such networks is incomparably high.

Modern corporate networks are built on the basis of many network devices, such as routers, switches, firewalls, etc. They can work independently of other devices, independently determine the rules for processing network packets on devices. Service information between such devices is determined by rules, as well as its application methods are determined by many different protocols. Service protocols solve some network administration tasks.

However, complex architecture, modern design approaches and service protocols do not make network administration easy. The solution to the existing problems can be considered in the new concept of software-configurable networks based on the open OpenFlow protocol. The concept will speed up routing, switching, and improve the ease of administration and maintenance of networks [1].

The main goal of organizing software-configurable networks is the ability to manage data when sent and received using OpenFlow. For the correct operation of the OpenFlow protocol, a network operating system (NOS) is required, thanks to which all network devices are combined into one whole. Also, NOS enables applications to manage the network at different levels. Controllers in such operating systems use common rules to send and receive data packets, which is a centralized system.

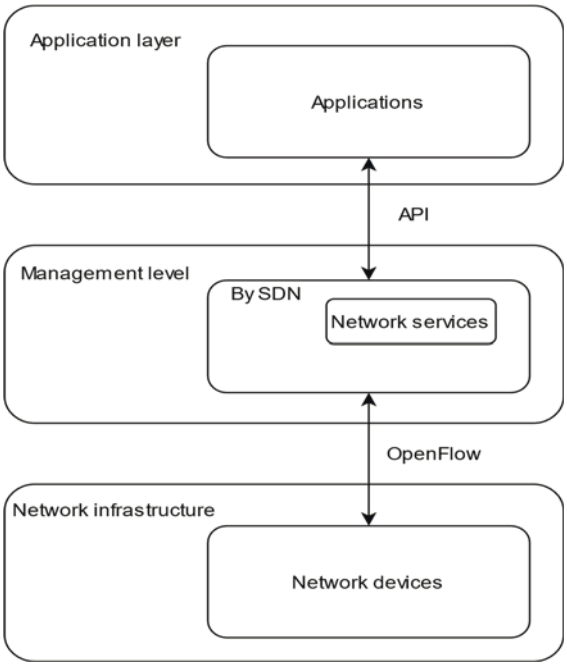

Fig. 1. SDN architecture

According to generally accepted rules, in the architecture of software-configurable networks [2] they represent three levels (Fig. 1):

- the layer responsible for the network infrastructure. It is a collection of switches and communication channels;
- the level responsible for network management. It is an operating system that provides different services, as well as software tools for managing the network and different network devices;
- the level responsible for the correct operation of applications.

Visual representation of OpenFlow

OpenFlow is a protocol (technology) that controls the process of processing and transmitting data, using routers and switches in computer networks. The OpenFlow protocol enables rapid scaling of networks, allowing a remote controller to change the actions of different network devices using precisely set routing rules. OpenFlow is a special feature of its kind for routers and switches from different manufacturers (Fig. 2).

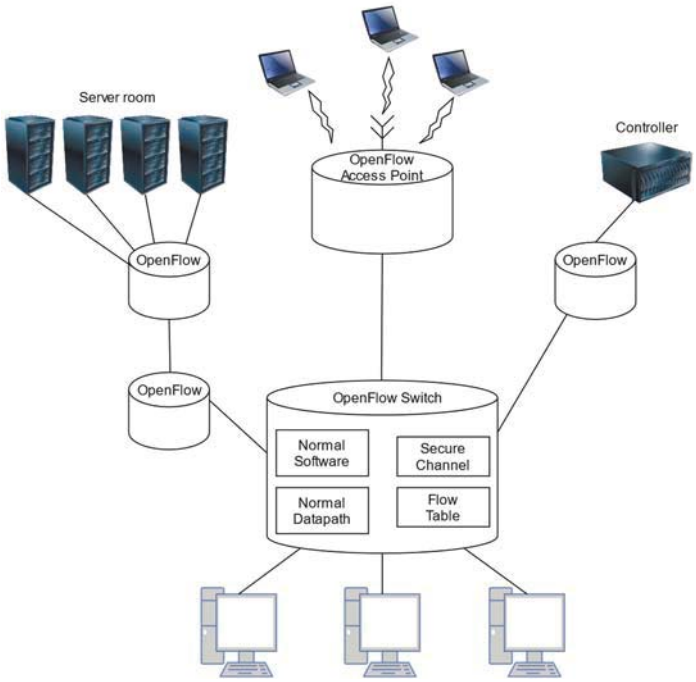

Fig. 2. Visual representation of OpenFlow

Due to the fact that it is possible to directly manage the forwarded data in the network, it is very profitable to use software-configured networks in large LANs. SDN make it possible to implement data transmission along several paths at once, as well as build priorities, thanks to which flows can be controlled. They support network virtualization and allow you to divide the load into several nodes of the network without obvious loss.

Logically centralized network data management allows all network management functions to be delegated to a single physical server called a controller. The controller has the ability to control one or more OpenFlow devices. Also, the controller has the property of controlling the network operating system, which provides various low-level services for managing the network, individual

parts of the network, network components, including applications that are carried out by high-level management of network data and flows [3].

NOS provides programs with access to low-level and high-level network parameters and constantly monitors the status of network devices. NOS is interpreted as a system that provides continuous monitoring of the state of the network. It manages and has full access to network resources [4].

An OpenFlow switch is an abstraction of a data link in the form of a table in which each entry contains information about packet flows, a port forwarding action, and a change or delete field. If a previously unknown packet arrives on the switch and there is no corresponding entry in the switching table, the packet will be sent to the controller. After that, the controller will decide what to do next with the new package.

It can accept the packet by adding an entry to the route table or discard and ignore it. If a packet is received by the controller, this route will be the forwarding rule for the following similar packets.

Installing and Configuring the OpenFlow Protocol on the Network Operating System

For the network operating system to support the protocol, it must be installed.

Start by installing SSH. SSH is a protocol that allows you to remotely connect to the operating system via TCP connections [5]:

*sudo apt-get -y install ssh

Installing Git, Distributed File Version Control [6]:

*sudo apt-get install git-core automake m4 pkg-config libtool

*git clone git://openflow.org/openflow.git

*cd openflow

*./boot.sh

Loading the necessary packages in order to compile the OpenFlow protocol:

*sudo apt-get install gcc

Unpacking and installing the OpenFlow environment:

*./configure

*make

*sudo make install

After the operations, the virtual machine will support the OpenFlow protocol and you can start installing other software.

After installing a special plug-in OpenFlow Wireshark Dissector, it will be possible to decode packets and present hierarchical data in a readable form:

*sudo apt-get install wireshark libgtk2.0-dev

*cd utilities/wireshark_dissectors/openflow

*make

*sudo make install

To conduct tests of the experiment, you need to put some packages. Installation:

*cd ~/<your openflow-dir>

*sudo regress/scripts/install_deps.pl

To prevent interruptions during testing, you need to disable IPv6 support:

*sudo apt-get remove avahi-daemon

or

*sudo apt-get install sysv-rc-conf

*sudo sysv-rc-conf avahi-daemon off

*sudo nano /etc/sysctl.conf

*net.ipv6.conf.all.disable_ipv6 = 1

*net.ipv6.conf.default.disable_ipv6 = 1

*sudo vi /etc/modprobe.d/blacklist.conf

*blacklist net-pf-10

*blacklist ipv6

After successfully installing all packages, you need to restart the VM:

*sudo shutdown -r now

It is important to update the OF_ROOT:

*cd ~/

*cp <openflow-dir>/regress/scripts/env_vars .

*vim env_vars

At this point, the preparation of the virtual machine is completed. It is necessary to check the correct installation. This is done using the superuser profile under which the OF_ROOT is loaded:

*source ~/env_vars

To automatically load the environment variable when the system starts, you need to add the ~/.bashrc command.

Virtual Ethernet connection:

*veth_setup.pl // the veth0 list will appear... 7

*/sbin/ifconfig | more // Ethernet Connection Check

Wireshark launch:

*wireshark &

After starting Wireshark, on the top panel, click Filter and enter:

*of || tcp.flags.reset == 1

To start generating packages, go to

*Capture->Interface и выбрать lo (Loopback).

Now you can start testing:

*of_user_veth_test.pl

After pre-configuration, packets will be generated in the network between the switch and the controller. Each test, with a certain packet size, will be considered complete after all generated packets have ended. Also, it is possible to view information about packets in the OpenFlow logs of the switch.

Experiment using OpenFlow switch

A pilot installation was assembled. It consists of: a personal computer (ATX motherboard, Intel Core i7 processor, 16 GB RAM, Linux Ubuntu OS), a NIC with a bandwidth of at least 1 Gb/s, a gigabit switch (Fig. 3), a gigabit router (Fig. 4), an OpenFlow device (Fig. 5).

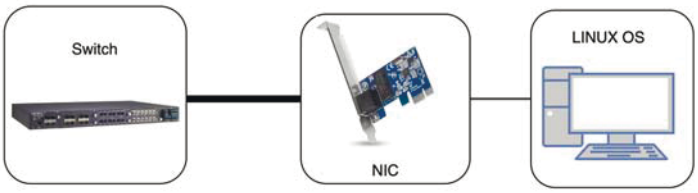

Fig. 3. Experimental installation with switch

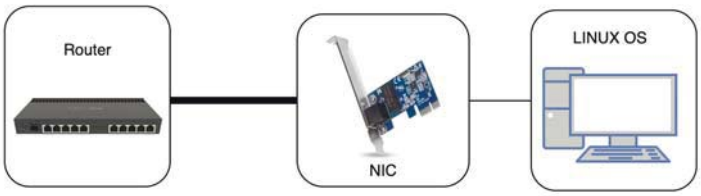

Fig. 4. Experimental installation with router

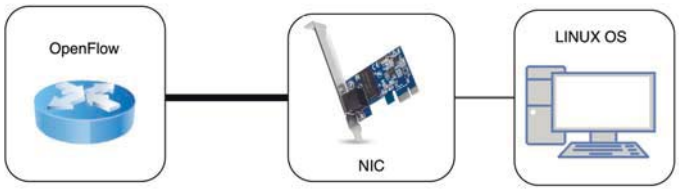

Fig. 5. Experimental installation with OpenFlow device

The throughput of the assembled experimental plant will be measured using a single system flow. Since packets will increase in size, performance should also increase. Since the system has one single thread, then IP routing will use the internal cache in its implementation. This means that the IP header testing process will occur by testing the headers without the need to do any complex search operations throughout the table.

Based on this, the difference between routing performance without OpenFlow and routing with OpenFlow will be small. This fact is explained by the fact that the OpenFlow device in the routing table will have an additional column with the field's header. The essence of the experiment will be to measure the performance [7] of devices with different packet sizes.

Results of the experiment

Figure 6 shows the efficiency of the forwarding table size. Table sizes are 8K (s), 64K (m) and 128K (l). OpenFlow mappings are formed by accessing destination ports and UDP traffic sources. The mapping table is created and populated automatically. New entries are added to it thanks to the look-back algorithm, that is, records from the table are deleted after the timeout has expired [8]. A stop time of 60 minutes is assigned to the forwarding table. Packets are sent through the switch with different source addresses.

Fig. 6. Flow Performance Results

With small packet sizes, the performance of the switch and the OpenFlow router is low [9]. The switch has the lowest performance with a packet size of 96 bytes. The OpenFlow device and router produce results that indicate a lack of performance. For OpenFlow, it is 0.09%, 6.1%, 9.0%, and for router 0.59%, 4.8%, 12.2%. For comparison with the switch, these values are 14.1%, 71.5%, 90.3%. And the bandwidth of the switch is less than 10%. As the package size increases, performance increases.

The OpenFlow router requires a 256-byte packet for maximum performance and a 1024 switch. Also, the performance is zero at:

- 256 and 512 bytes with 8k table size on OpenFlow and Routing devices;
- 256 and 512 bytes with table sizes 64k and 128k on the Routing device;
- 1024 and 1500 bytes with table sizes 8k, 64k, 128k on OpenFlow and Routing devices.

Until now, only exact rule matches for OpenFlow have been considered. If you organize mask matching rules in OpenFlow, the key figures change [10, 11]. Let's consider two types of organization of the hash table and with linear search in the form of rule records (up to 100). OpenFlow is implemented on the basis of the second type of tables. Figure 7 shows that there is a 33% performance disadvantage on the 96-byte packet

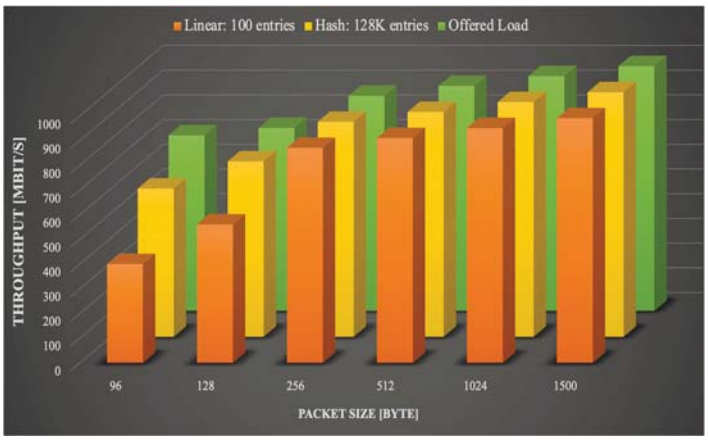

Fig. 7. Results of the experiment on the rule of matching by mask

The principle of operation of the hash table: constant sending of data, independent of the size of the table. Linear table: Sending data and search operation depends on size [12]. Formula for calculating the average time for processing a T_p packet:

$$T_p = L - 2\,(S/C + D) \qquad (1)$$

where, L – packet transmission delay (measured by the router), S – packet size, C – channel performance, D – propagation delay.

The results of the experiment calculation are presented in Table 1:

Table 1

Results of the experiment

	Linear Table Size					Hash Table Size				
Packet Size [byte]	1	25	50	75	100	1	32k	64k	96k	128k
96	28,21	28,21	28,21	28,21	28,21	28,21	28,21	28,21	28,21	28,21
128	29,19	30,05	29,19	29,19	29,19	29,19	29,19	29,19	29,19	27,16
256	28,73	29,73	29,73	30,29	29,73	29,73	29,73	29,73	29,73	29,73
512	28,64	28,64	28,64	28,64	28,64	28,64	28,64	28,64	28,64	28,64
1024	28,44	28,44	27,44	28,44	28,44	29,97	28,44	26,44	28,44	28,44
1500	24,18	26,18	26,18	26,18	26,18	26,18	26,18	26,18	26,18	26,18

As you can see from the data obtained, the results of the calculation for linear search and for hashes of the table are almost the same.

Conclusion

The implementation of OpenFlow on Linux systems is able to offer very good performance. OpenFlow shows good performance with a large number of threads. If you compare the results of different tables up to 128K and up to 100 rules with a large number of threads, OpenFlow has better latency and performance.

It follows from this that it is better to use it in production networks. When switching the second layer, which is carried out in Linux, the low throughput is described about 40% for the router, and about 30% for the OpenFlow device.

References

1. K.E. Samuilov, I.A. Shalimov, I.G. Buzhin, Y.B. Mironov (2018) Model of functioning of telecommunication equipment of software-configurable networks. *Modern information technologies and IT education*. Vol. 14. No. 1, pp. 13-26.
2. Open Networking Foundation. SDN ARCH. *Technical Reference normative*. 2014. No. 2. P. 68.
3. Openflow [Electronic resource] – Access mode: www.openflow.org (04.12.2022).
4. Open systems [Electronic resource] – Access mode: http://www.osp.ru/os/2012/09/13032491/ (04.12.2022).
5. Wikipedia SSH [Electronic resource] – Access mode: ru.wikipedia.org/wiki/SSH (04.12.2022).
6. Gi everything [Electronic resource] – Access mode: git-scm.com/ (04.12.2022).
7. .V. Malakhov, V.N. Tarasov, I.V. Kartashevsky (2015) Theoretical and experimental study of delay in software-configurable networks. *Infocommunication technologies*. No. 4, pp. 409-413.
8. .V. Malakhov, V.N. Tarasov, N.F. Bakhareva, I.V. Kartashevsky (2016) The effect of the size of the TCP window on the distribution of intervals between traffic packets in software configurable SDN networks. *Infocommunication technologies*. No.4, pp. 384-389.
9. .V. Malakhov, V.N. Tarasov, N.F. Bakhareva, I.V. Kartashevsky (2016) Analysis of intervals between traffic packets in SDN networks depending on the size of the TCP window. *3rd International Scientific-Practical Conference Problems of Infocommunications Science and Technology*. No.10, pp. 15-17.
10. G.A. Bashilov (2011) Software and hardware idyll, or OpenFlow. *Log of network solutions*. No. 9, pp. 2-5.
11. R.I. Smelyansky (2012) Software-configurable networks//Open systems. No. 9, pp. 34-36.
12. M.F. Buranova, V.G. Kartashevsky (2022) g/g/1 queue analysis for software-configurable networks based on OpenFlow. *T-Comm*. No. 4, pp. 4-13.

ОЦЕНКА ВЛИЯНИЯ ПРОТОКОЛА OPENFLOW НА ПРОИЗВОДИТЕЛЬНОСТЬ СЕТЕВЫХ УСТРОЙСТВ

Якупов Денис Олегович, *Поволжский государственный университет телекоммуникаций и информатики, г. Самара, Россия,* ***d.yakupov@psuti.ru***

Малахов Сергей Валерьевич, *Поволжский государственный университет телекоммуникаций и информатики, г. Самара, Россия,* ***malakhov-sv@psuti.ru***

Аннотация

Очень перспективной областью компьютерных сетевых технологий является использование программно-конфигурируемых сетей (SDN). Принципы SDN были впервые разработаны в исследовательских лабораториях Стэнфорда и Беркли, а в настоящее время разрабатываются консорциумом Open Network Foundation и европейским проектом OFELIA. Благодаря открытости протокола OpenFlow и перенос логики управления на отдельный контроллер упрощает программно-аппаратное обеспечение активного сетевого оборудования. Данный факт позволяет производителям в будущем снизить стоимость на оборудование, а значит и стоимость создания компьютерных сетей. Поэтому исследования в данной области является весьма перспективным. Данная работа посвящена исследованию производительности ПКС, а именно как влияет внедрение протокола OpenFlow на качество пересылки данных между сетевыми устройствами. При помощи единственного потока оценивается максимальная пропускная способность. Для этого собраны три экспериментальная установка сети с разными сетевыми устройствами, коммутатором, маршрутизатором и устройством OpenFlow, а также выделенным сервером под ОС Linux с поддержкой протокола OpenFlow. Суть экспериментов заключается в изменение размера пакетов и размера таблицы переадресации для оценки производительности ПКС с разными сетевыми устройствами. Результаты анализа показали, что, реализация OpenFlow в системах Linux в состоянии предложить очень хорошую производительность. OpenFlow показывает хорошую производительность при большом количестве потоков.

Ключевые слова: *протокол OpenFlow, производительность, коммутатор, маршрутизатор, сетевая операционная система, хэш таблица, линейный поиск, генерация пакетов, экспериментальная установка.*

Литература

1. *Самуйлов К.Е., Шалимов И.А., Бужин И. Г., Миронов Ю.Б.* Модель функционирования телекоммуникационного оборудования программно-конфигурируемых сетей // Современные информационные технологии и ИТ-образование. 2018. Т. 14. No 1. С. 13-26.
2. Open Networking Foundation. SDN ARCH 1.0 06062014 (Technical Reference), non-normative, type 2, 2014. 68 p. URL: https://www.opennetworking.org/wp-content/uploads/2013/02/TR_SDN_ARCH_1.0_06062014.pdf (дата обращения: 10.02.2018).
3. Openflow URL: www.openflow.org (Дата обращения 01.12.2022).
4. Открытые системы URL: http://www.osp.ru/os/2012/09/13032491/ (Дата обращения 01.12.2022).
5. Википедия SSH URL: ru.wikipedia.org/wiki/SSH (Дата обращения 01.12.2022).
6. git everything URL: git-scm.com/ (Дата обращения 01.12.2022).
7. *Малахов С.В. Тарасов В.Н., Карташевский И.В.* Теоретическое и экспериментальное исследование задержки в программного-конфигурируемых сетях // Инфокоммуникационные технологии. 2015. №4. С. 409-413.
8. *Малахов С.В. Тарасов В.Н., Бахарева Н.Ф., Карташевский И.В.* Влияние размера TCP-окна на распределение интервалов между пакетами трафика в программно-конфигурируемых сетях SDN // Инфокоммуникационные технологии. 2016. №4. С. 384-389.
9. *Малахов С.В. Тарасов В.Н., Бахарева Н.Ф., Карташевский И.В.* Анализ интервалов между пакетами трафика в сетях SDN в зависимости от размера окна TCP // 3rd International Scientific-Practical Conference Problems of Infocommunications Science and Technology. 2016. №10, pp. 15-17.
10. *Башилов Г.А.* Программно-аппаратная идиллия, или OpenFlow // Журнал сетевых решений. 2011. № 9. С. 2-5.
11. *Смелянский Р.И.* Программно-конфигурируемые сети // Открытые системы. 2012. № 9. С 34-36.
12. *Буранова М.Ф., Карташевский В.Г.* Анализ очереди типа G/G/1 для программно-конфигурируемых сетей на основе OpenFlow // T-Comm: телекоммуникации и транспорт. 2022. № 4. С. 4-13.

Информация об авторах:

Якупов Денис Олегович, *Поволжский государственный университет телекоммуникаций и информатики, ассистент кафедры ПрИ, г. Самара, Россия*

Малахов Сергей Валерьевич, *Поволжский государственный университет телекоммуникаций и информатики, доцент кафедры ПрИ, к.т.н., г. Самара, Россия*