СПбГУТ)))
ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ТЕЛЕКОММУНИКАЦИИ
Март 2016 Том 4 № 1 ISSN 2307-1303 http://www.itt.sut.ru
TELECOM IT
March 2016 Vol. 4 Iss. 1 ISSN 2307-1303 http://www.itt.sut.ru

ТЕСТИРОВАНИЕ КОНТРОЛЛЕРОВ ПРОГРАММНО-КОНФИГУРИРУЕМОЙ СЕТИ НА БАЗЕ МОДЕЛЬНОЙ СЕТИ

А. Г. Владыко[1], Н. А. Матвиенко[1], М. И. Новиков[1], Р. В. Киричек[1]

[1] СПбГУТ, Санкт-Петербург, 193232, Российская Федерация
Адрес для переписки: ruslan.stk@gmail.com

Информация о статье
УДК 621.391
Язык статьи – русский.
Поступила в редакцию 29.01.16, принята к печати 29.02.16.

Ссылка для цитирования: Владыко А. Г., Матвиенко Н. А., Новиков М. И., Киричек Р. В. Тестирование контроллеров программно-конфигурируемой сети на базе модельной сети // Информационные технологии и телекоммуникации. 2016. Том 4. № 1. С. 17–28.

Аннотация
Предмет исследования. Исследуются контроллеры программно-конфигурируемые сети. **Метод.** В качестве метода исследования применяется натурный эксперимент. В работе оценивается производительность контроллеров в зависимости от различных параметров (количество потоков, количество ядер и др.). **Основные результаты.** В статье приведены результаты оценки производительности контроллеров программно-конфигурируемых сетей Floodlight, OpenDaylight, Mul, Beacon, Maestro, Ryu на базе серии нагрузочных тестов. **Практическая значимость.** Полученные результаты позволят выбрать необходимый тип контроллера при планировании сетей с различным количеством подключаемых абонентов.
Ключевые слова
программно-конфигурируемая сеть, SDN-контроллер, модельная сеть, тестирование.

SDN-CONTROLLERS BENCHMARKING BASED ON MODEL NETWORK

A. Vladyko[1], N. Matvienko[1], M. Novikov[1], R. Kirichek[1]

[1] SPbSUT, St. Petersburg, 193232, Russian Federation
Corresponding author: ruslan.stk@gmail.com

Article info
Article in Russian.
Received 29.01.16, accepted 29.02.16.

For citation: Vladyko A., Matvienko N., Novikov M., Kirichek R.: SDN-Controllers Benchmarking Based on Model Network // Telecom IT. 2016. Vol. 4. Iss. 1. pp. 18–28 (in Russian).

Abstract
Object of research. Software-defined networking controller platforms. **Method.** As a research method is used full-scale experiment. In the estimated performance of controllers depending on various parameters (the number of threads, number of cores, and others.). **Core results.** The article presents the results of the evaluation of software and configurable controllers performance networks Floodlight, OpenDaylight, Mul, Beacon, Maestro, Ryu on the basis of a series of stress tests. **Practical relevance**. The results will allow to select the desired type of controller in the planning of networks with different number of connected subscribers.
Keywords
Software-defined Networking, SDN-controller, model network, benchmarking.

Введение

В настоящее время остро стоит проблема перегруженности сетей связи, основанных на традиционных технологиях [1, 2]. Учитывая тот факт, что к 2020 году к сетям общего пользования будут подключено более 50 миллиардов устройств Интернета вещей и Промышленного интернета, есть риск, что сети не справятся с трафиком, генерируемым таким количеством узлов [3, 4]. Кроме этого, появится сложность в управлении и конфигурировании сетей такого масштаба.

Отмеченные недостатки современных сетей позволяет компенсировать технологии программно-конфигурируемых сетей (*Software-Defined Networks,* SDN) [5, 6].

SDN предполагает логически централизованное управление сетью за счет программирования. Программируемые сети позволять гибко распределять и обрабатывать большие потоки трафика, устранять узкие места и резервировать ресурсы сети, предотвращая перегрузку. Централизованный метод управления позволяет сделает контроль и настройку большого количества устройств гораздо проще.

Концепция SDN предполагает:

- разделение процессов передачи данных и управления данными;
- логически централизованный уровень управления данными;
- виртуализация физических ресурсов сети;
- единый и унифицированный интерфейс (*openflow*) между плоскостью управления и плоскостью передачи данных (не зависит от вендора).

Архитектура SDN состоит из трех уровней:

Уровень сетевых приложений: на данном уровне реализуются различные функции управления сетью: управление потоками данных в сети, управление безопасностью, мониторинг трафика, управление QoS, управления политиками и так далее [7, 8].

Уровень управления: на уровне управления отслеживается и поддерживается глобальное представление сети (топология сети). Также на этом уровне реализуется программный интерфейс (API) для сетевых приложений.

Уровень инфраструктуры сети: включает в себя сетевые устройства SDN (коммутаторы *OpenFlow*) и каналы передачи данных [9].

SDN-контроллер

Контроллер является ключевым элементом SDN, он выступает в роли «мозга» всей сети. Производительность и возможности сети напрямую связаны с характеристиками контроллера. Сам контроллер представляет собой сетевую операционную систему, установленную на выделенном физическом сервере.

К основным функциям контроллера относится:

- управление устройствами сети;
- управление топологией (построение топологии сети, обработка добавления/удаления новых элементов сети);
- управление приложениями;
- управление доступными ресурсами сервера (потоками, ядрами).

Основные характеристики SDN-контроллера

Производительность:

- скорость обработки потоков – количество запросов от коммутаторов, обрабатываемых контроллером в секунду – (потоки/секунду);
- задержка – время, затрачиваемое контроллером на обработку одного запроса – (миллисекунды).

Масштабируемость:

- изменение показателей производительности при увеличении числа соединений с коммутаторами;
- изменение показателей производительности при увеличении числа конечных узлов в сети;
- изменение показателей производительности при увеличении числа ядер процессора.

Ресурсоемкость:

- загрузка ядер процессора;
- использование физической памяти.

Надежность:

- количество отказов за время тестирования;
- время безотказной работы при заданном профиле нагрузок.

Обоснование выбора критериев тестирования

Как известно, модельные сети нашли широкое распространение при тестировании NGN [10].

Тестирование SDN-контроллеров проводилось на базе модельной сети Лаборатории Интернета Вещей СПбГУТ [3, 11, 12].

Используя модельную сеть можно провести комплексное тестирование контроллера, как в штатном режиме работы, так и под нагрузкой с применением стрессовых режимов, которые позволят более качественно и объективно оценить его характеристики.

Выбор критериев оценки характеристик контроллера основан на ранее разработанных методиках тестирования фрагментов программно-конфигурируемой сети [9] с учетом положений RFC 2889 [13] и RFC 2544 [14].

Контроллеры, задействованные в исследовании

– *Floodlight.* Контроллер с открытыми исходными кодами, поддерживаемый открытым сообществом разработчиков. Разработан на основе платформы контроллера Beacon, на языке Java. Имеет лицензию Apache, т. е. может использоваться для любых целей.

– *OpenDaylight.* OpenDaylight является открытым проектом с модульной и гибкой платформой. Этот контроллер реализован программно, внутри своей собственной виртуальной java-машины *(jvm*). Таким образом, он может быть развернут на любом оборудовании и платформе операционной системы, которая поддерживает Java.

– *Mul.* Разработан на языке *C*, имеет многопоточную инфраструктуру на уровне ядра. Он поддерживает многоуровневый интерфейс для сетевых приложений. Главной задачей при разработке этого контроллера было обеспечение производительности и надежности, что необходимо при развертывании высоконагруженных сетей.

– *Beacon.* Кросс-платформенный, модульный OpenFlow контроллер на Java. Beacon используется во многих научно-исследовательских проектах и тестовых внедрениях. Beacon применяется в экспериментальном ЦОДе Стэнфорда, в котором он управляет 100 виртуальными и 20 физическими коммутаторами. Работает на многих платформах, начиная от высокопроизводительных многоядерных Linux-серверов до смартфонов на Android.

– *Maestro*. Сетевая операционная система, разработанная в Rice University. Maestro предоставляет интерфейсы для реализации модульных приложений управления сетью для доступа и изменения состояния сети, а также координации их взаимодействия. Несмотря на то, что этот проект направлен на создание OpenFlow контроллера, Maestro не ограничивается только OpenFlow-сетями. Maestro разработана на Java (и сама платформа, и ее компоненты), является универсальной для различные ОС и архитектур.

– *Ryu*. Реализован на языке Python. Предоставляет API, которые упрощают создание новых приложений сетевого управления. Поддерживает различные протоколы управления сетевыми устройствами, такие как OpenFlow, NETCONF, OF-config. Полностью поддерживает OF v.1.0, 1.2, 1.3, 1.4, 1.5.

Утилита для бенчмаркинга контроллеров Cbench

Режимы работы:

– режим измерения скорости обработки потоков;

– режим измерения задержки.

Алгоритм работы:

– Cbench имитирует работу *N* коммутаторов OpenFlow;

– создает *N* OpenFlow сессий к контроллеру.

Результат работы:

– одного теста: количество потоков, устанавливаемых контроллером в секунду (потоков/с) (суммируется по всем коммутаторам);

– Cbench: минимальное, максимальное и среднее значение запросов в секунду (запросов/с) по всем тестам.

Пример использования утилиты:

На рис. 1. Представлен пример использования утилиты тестирования CBench.

```
root@sdn:/home/sdn/oflops/cbench# cbench -c 192.168.0.1 -p 6653 -m 10000 -l 2 -s 1 -M 10000 -t
cbench: controller benchmarking tool
   running in mode 'throughput'
   connecting to controller at 192.168.0.1:6653
   faking 1 switches offset 1 :: 2 tests each; 10000 ms per test
   with 10000 unique source MACs per switch
   learning destination mac addresses before the test
   starting test with 0 ms delay after features_reply
   ignoring first 1 "warmup" and last 0 "cooldown" loops
   connection delay of 0ms per 1 switch(es)
   debugging info is off
13:50:57.311 1   switches: flows/sec:  758115   total = 75.770910 per ms
13:51:07.412 1   switches: flows/sec:  257941   total = 25.792235 per ms
RESULT: 1 switches 1 tests min/max/avg/stdev = 25792.24/25792.24/25792.24/0.00 responses/s
```

Рис. 1. Пример использования утилиты Cbench

В таблице 1 перечислены опции командной строки утилиты CBench.

Таблица 1.

Опции командной строки, утилиты Cbench

Параметр	**Описание**
-c	имя/IP адрес сервера, на котором запущен контроллер
-p	порт контроллера
-m	длительность одного теста (в миллисекундах)
-l	ЧИСЛО тестов, проводимых за один запуск
-s	число эмулируемых коммутаторов
-M	количество хостов с уникальными MAC-адресами на один коммутатор
-t	запуск в режиме тестирования пропускной способности

Экспериментальное исследование

Физическая схема экспериментального стенда представлена на рис. 2.

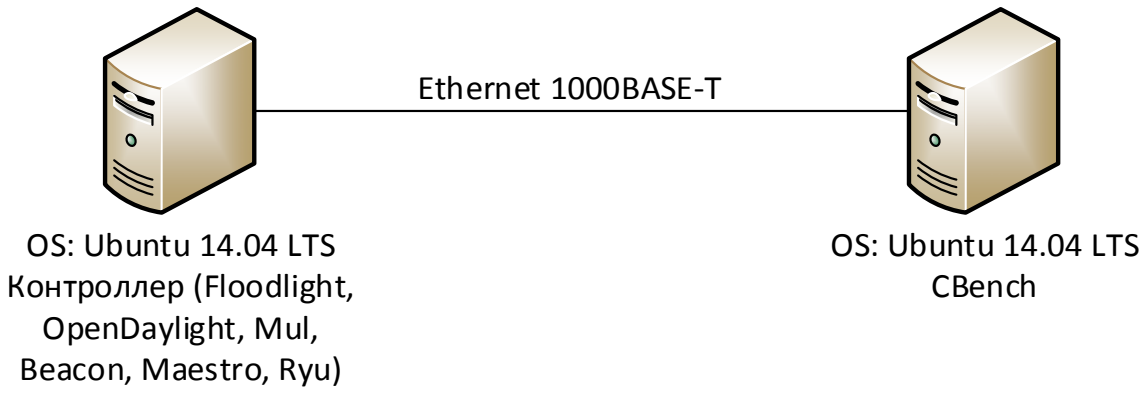

Рис. 2. Схема лабораторного стенда

Аппаратная составляющая сервера:

1. Процессор: Intel Xeon E3-1220 V2 3.10GHz (4 cores, 4 threads).

2. Оперативная память: 3 платы Foxline FL1600D3U11-4G DDR3 4GB DIMM.
3. Жесткий диск: Hitachi HDS721010CLA332 1000GB.

Программная составляющая сервера:

1. Средство для тестирования Cbench.
2. Контроллеры:

– OpenDaylight – Lithium SR3;
– Floodlight – v 1.0;
– Mul – v 3.2.7;
– Beacon – v 1.0.2;
– Maestro – v 0.2.0;
– Ryu – v 4.1.

Сценарии тестирования

Производительность контроллера:

1. Зависимость производительности от количества подключенных коммутаторов (1, 5, 10, 20, 50, 100), при фиксированном числе хостов (10^4).

2. Зависимость производительности от числа оконечных узлов (10^3, 10^4, 10^5, 10^6, 10^7), при фиксированном количестве коммутаторов (20).

3. Зависимость производительности от количества используемых ядер процессора.

Задержка контроллера:

1. Зависимость величины задержки от количества подключенных хостов (10^3, 10^4, 10^5, 10^6, 10^7), при фиксированном числе коммутаторов (1).

Результаты тестирования

Для качественного отображения результатов, контроллеры были разделены на высокопроизводительные (*Maestro, Mul*), средней производительности (*Floodlight, OpenDaylight*) и непроизводительные (*Ryu, Beacon*).

1. Зависимость производительности контроллеров от количества коммутаторов представлена на рис. 3–5.

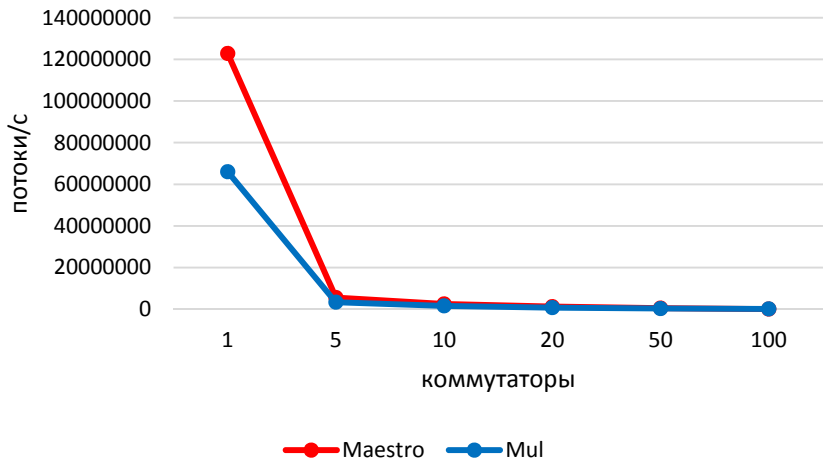

Рис. 3. Зависимость производительности контроллеров (*Maestro, Mul*) от количества коммутаторов

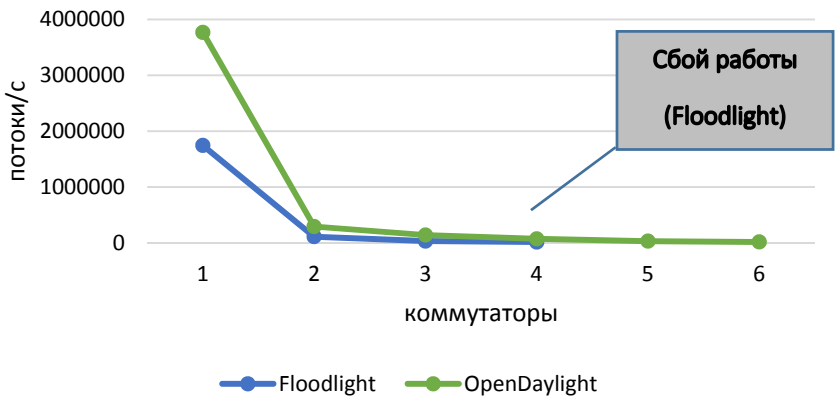

Рис. 4. Зависимость производительности контроллеров *(Floodlight, OpenDaylight*) от количества коммутаторов

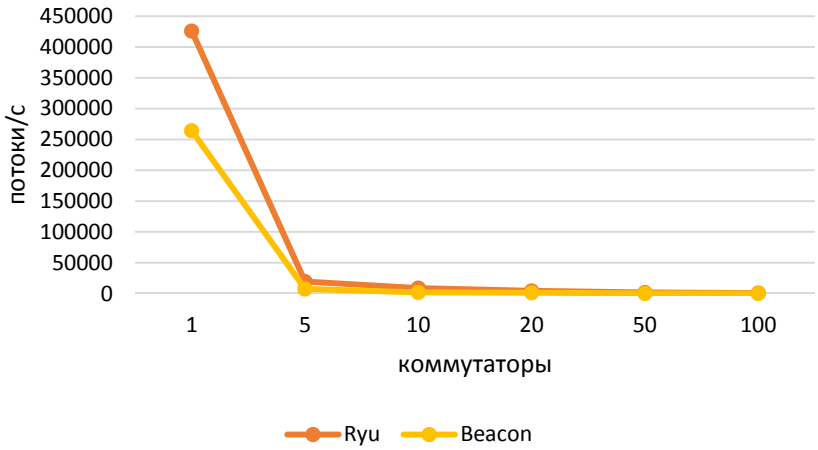

Рис. 5. Зависимость производительности контроллеров (*Ryu, Beacon*) от количества коммутаторов

Увеличивая число подключенных коммутаторов, количество обрабатываемых потоков, поступающих от каждого коммутатора уменьшалось;

2. Зависимость производительности контроллера от количества подключенных оконечных узлов представлена на рис. 6–8.

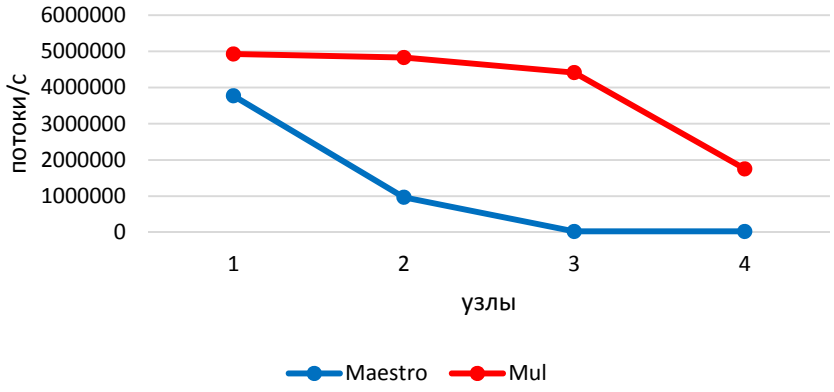

Рис. 6. Зависимость производительности контроллеров (*Maestro, Mul*) от количества подключенных оконечных узлов

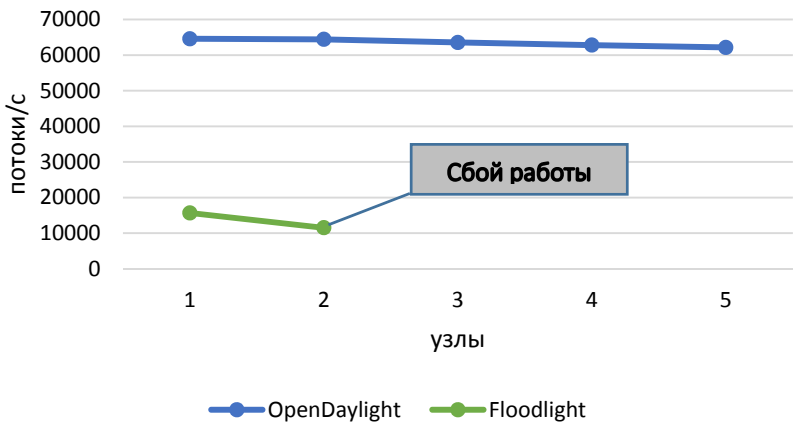

Рис. 7. Зависимость производительности контроллеров (*Floodlight, OpenDaylight*) от количества подключенных оконечных узлов

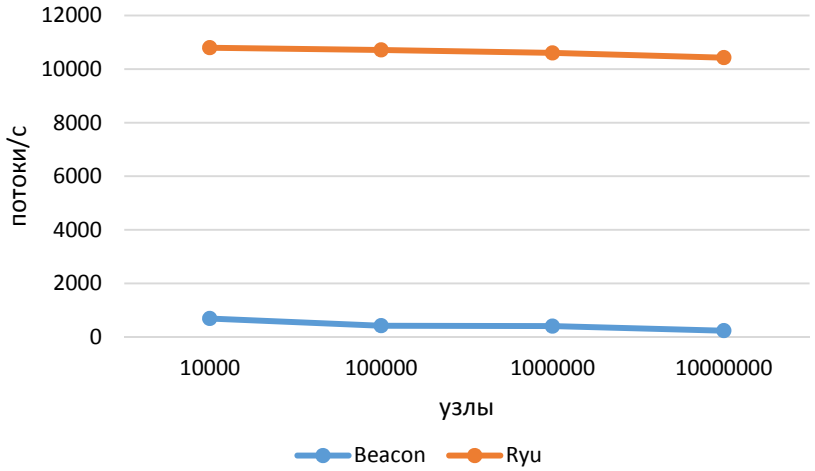

Рис. 8. Зависимость производительности контроллеров (*Beacon, Ryu*) от количества подключенных оконечных узлов

Повышая количество подключенных оконечных узлов, наблюдалось снижение числа обрабатываемых контроллером потоков.

3. Зависимость производительности контроллера от количества используемых ядер (потоков) процессора представлена на рис. 9, 10.

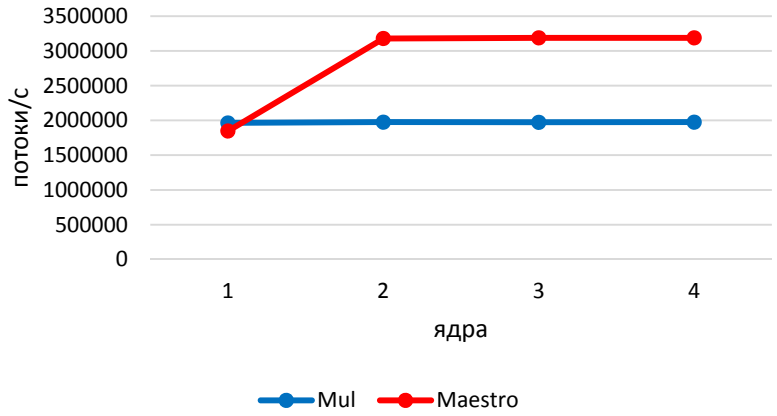

Рис. 9. Зависимость производительности контроллеров (*Maestro, Mul*) от количества используемых ядер (потоков) процессора

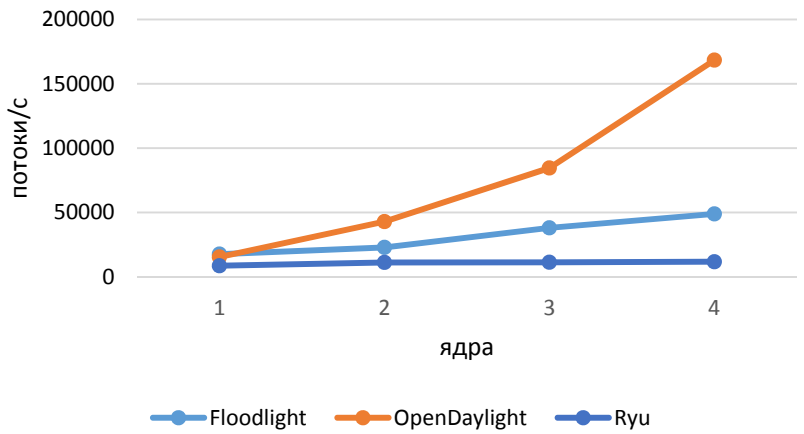

Рис. 10. Зависимость производительности контроллеров (*Floodlight, OpenDaylight, Ryu*) от количества используемых ядер (потоков) процессора

Увеличивая число используемых ядер процессора количество обрабатываемых контроллером потоков возрастало.

4. Зависимость задержки контроллера от количества подключенных оконечных узлов представлена на рис. 11, 12.

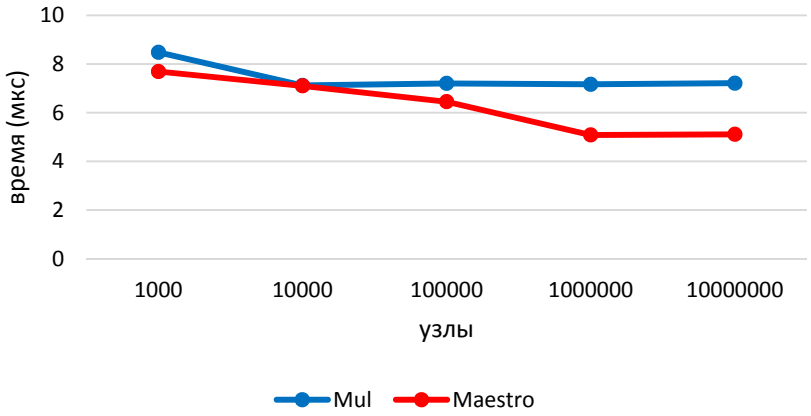

Рис. 11. Зависимость задержки контроллеров (*Maestro, Mul*) от количества подключенных оконечных узлов

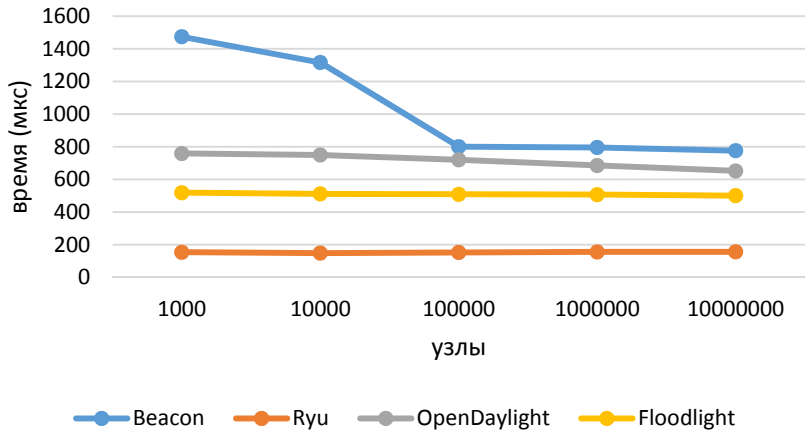

Рис. 12. Зависимость задержки контроллеров *(Floodlight, OpenDaylight, Ryu, Beacon)* от количества подключенных оконечных узлов

При увеличении количества оконечных узлов, учитывая фиксированное число коммутаторов время обработки контроллером потоков уменьшалось.

Выводы

Методика позволила оценить основные характеристики SDN-контроллера:

Производительность и масштабируемость:

1. При увеличении числа подключенных коммутаторов, количество обрабатываемых потоков, поступающих от каждого коммутатора уменьшалось.

2. Повышая количество оконечных узлов, число обрабатываемых контроллером потоков снижалось, но незначительно.

3. Увеличивая число используемых сетевой операционной системой ядер процессора, количество обрабатываемых контроллером потоков возрастало.

Задержка.

Время обработки потоков уменьшалось, так как с учетом повышения числа оконечных узлов (хостов), генерировалось большее количество пакетов, которые контроллер в рамках заданного времени обрабатывал с большей скоростью, за меньшее время.

Ресурсоемкость:

1. Оперативная память. Использование оперативной памяти контроллерами представлена в таблице 2.

Таблица 2.

Ресурсоемкость контроллеров

Контроллер	**OpenDaylight**	**Floodlight**	**Mul**	**Maestro**	**Ryu**
Количество используемой оперативной памяти сервера, Гб	2,1	7,3	0,8	1,2	4,1

2. Загрузка ядер процессора. Загрузка ядер процессора контроллерами представлена в таблице 3.

Таблица 3.

Загрузка ядер процессора

Контроллер	**OpenDaylight**	**Floodlight**	**Mul**	**Maestro**	**Ryu**
Загрузка ядер процессора	3 ядра 96–99 %, 1 ядро 85–87 %	4 ядра 97–100 %	4 ядра 90–98 %	4 ядра 70–98 %	3 ядра 7–18 %, 1 ядро 99 %

В ходе исследований лучшие результаты по производительности и стабильности работы продемонстрировали контроллеры Maestro и Mul, так же данные контроллеры лучше оптимизированы в использовании физических ресурсов сервера. Данные контроллеры могут быть использованы для управления большими сетями.

Контроллеры OpenDaylight, Floodlight и Ryu показали низкую-среднюю производительность. Данные контроллеры могут быть использованы для организации небольших сетей.

Контроллер Beacon работал нестабильно, что является недопустимым при реализации SDN.

Заключение

При проведении дальнейших исследований предполагается провести комплексное тестирование проприетарных SDN-контроллеров, надежности их работы при высокой нагрузке в течении длительного времени, а также реакцию на некорректно сформированные сообщения. На основе имеющихся программных решений по бенчмаркингу SDN-контроллеров разработать собственное средство тестирования с расширенным функционалом и интуитивно понятным графическим интерфейсом.

Литература

1. Гольдштейн Б. С., Кучерявый А. Е. Сети связи пост-NGN. СПб.: БХВ-Петербург, 2013. 160 с.

2. Кучерявый А. Е., Киричек Р. В., Парамонов А. И., Прокопьев А. В. Эволюция исследований в области беспроводных сенсорных сетей // Информационные технологии и телекоммуникации. 2014. № 4. С. 29–41.

3. Kirichek R., Koucheryavy A. Internet of Things Laboratory Test Bed // Lecture Notes in Electrical Engineering. 2016. Vol. 348. pp. 485–494.

4. Kirichek R., Golubeva M., Kulik V., Koucheryavy A. The home network traffic models investigation // 18th International conference on advanced communication technology (ICACT). 2016. pp. 97–100.

5. Kreutz, D., Ramos, F., Verissimo, P., Rothenberg, C., Azodolmolky, S., Uhlig, S. Software-Defined Networking: A Comprehensive Survey // Proceedings of the IEEE. 2015. Vol. 103. Iss. 1. pp. 14–76.

6. Xia, W., Wen, Y., Foh, C., Niyato, D., Xie, H. A Survey on Software-Defined Networking // IEEE Communications Surveys &Tutorials. 2015. Vol. 17. Iss. 1. pp. 27–51.

7. Летенко И. Д. Нечеткая модель динамического управления трафиком в программируемых сетях // Системы управления и информационные технологии. 2015. Т. 62. № 4.1. С. 179–184.

8. Dotcenko S., Vladyko A., Letenko I. A fuzzy logic-based information security management for software-defined networks // 16th International Conference on Advanced Communication Technology (ICACT). 2014. pp. 167–171.

9. Владыко А. Г., Киричек Р. В., Великоречин М. А., Думин Д. И. Комплексная методика тестирования фрагмента программно-конфигурируемой сети // Информационные технологии и телекоммуникации. 2015. № 2. С. 20–29.

10. Васильев А. Б., Тарасов Д. В., Андреев Д. В., Кучерявый А. Е. Тестирование сетей связи следующего поколения. М. : Изд-во ФГУП ЦНИИС, 2008. 140 с.

11. Киричек Р. В., Владыко А. Г., Захаров М. В., Кучерявый А. Е. Модельные сети для Интернета вещей и программируемых сетей // Информационные технологии и телекоммуникации. 2015. № 3. С. 17–26.

12. Kirichek R., Vladyko A., Zakharov M., Koucheryavy A. Model networks for Internet of Things and SDN // 18th International conference on advanced communication technology (ICACT). 2016. pp. 76–79.

13. Mandeville R., Perser J. RFC 2889. Benchmarking Methodology for LAN Switching Devices. 2000.

14. Bradner S., McQuaid J. RFC 2544. Benchmarking Methodology for Network Interconnect Devices. 1999.

References

1. Goldstein B. S., Koucheryavy A. E. Seti svyazi post-NGN (Post-NGN Telecommunication Networks). Saint-Petersburg: BHV-Petersburg, 2013. 160 p.

2. Koucheryavy A. E., Kirichek R. V., Paramonov A. I., Prokopiev A. V. The Evolution in the Field of Wireless Sensor Networks Research // Telecom IT. 2014. no 4. pp. 29–41.

3. Kirichek R., Koucheryavy A. Internet of Things Laboratory Test Bed // Lecture Notes in Electrical Engineering. 2016. Vol. 348. pp. 485–494.

4. Kirichek R., Golubeva M., Kulik V., Koucheryavy A. The Home Network Traffic Models Investigation // 18th International conference on advanced communication technology (ICACT). 2016. pp. 97–100.

5. Kreutz, D., Ramos, F., Verissimo, P., Rothenberg, C., Azodolmolky, S., Uhlig, S. Software-Defined Networking: A Comprehensive Survey // Proceedings of the IEEE. 2015. Vol. 103. Iss. 1. pp. 14–76.

6. Xia, W., Wen, Y., Foh, C., Niyato, D., Xie, H. A Survey on Software-Defined Networking // IEEE Communications Surveys &Tutorials. 2015. Vol. 17. Iss. 1. pp. 27–51.

7. Letenko I. D. Fuzzy Model Control of Network Traffic in SDN // Sistemy upravleniya i informacionnye tekhnologii. 2015. Vol. 62. no 4.1. pp. 179–184.

8. Dotcenko S., Vladyko A., Letenko I. A Fuzzy Logic-Based Information Security Management for Software-Defined Networks // 16th International Conference on Advanced Communication Technology (ICACT). 2014. pp. 167–171.

9. Vladyko A. G., Kirichek R. V., Velikorechin M. A., Dumin D. I. Benchmarking Methodology of Software-Defined Networks // Telecom IT. 2015. no 2. pp. 20–29.

10. Vasil'ev A. B., Tarasov D. V., Andreev D. V., Koucheryavy A. E. Testirovanie setej svyazi sleduyushchego pokoleniya (Testing of Next Generation Networks). Moscow: FGUP ZNIIS, 2008. 140 p.

11. Kirichek R. V., Vladyko A. G., Zakharov M. V., Koucheryavy A. E. Model Networks for Internet of Things and Software-Defined Networks // Telecom IT. 2015. no 3. pp. 17–26.

12. Kirichek R., Vladyko A., Zakharov M., Koucheryavy A. Model Networks for Internet of Things and SDN // 18th International conference on advanced communication technology (ICACT). 2016. pp. 76–79.

13. Mandeville R., Perser J. RFC 2889. Benchmarking Methodology for LAN Switching Devices. 2000.

14. Bradner S., McQuaid J. RFC 2544. Benchmarking Methodology for Network Interconnect Devices. 1999.

Владыко Андрей Геннадьевич – кандидат технических наук, начальник управления, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, vladyko@sut.ru

Матвиенко Никодим Александрович – студент, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, matvienko.na@spbgut.ru

Новиков Максим Игоревич – студент, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, novikov.mi@spbgut.ru

Киричек Руслан Валентинович – кандидат технических наук, доцент, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, ruslan.stk@gmail.com

Vladyko Andrei – Ph.D., the head of Department, SPbSUT, St. Petersburg, 193232, Russian Federation, vladyko@sut.ru

Matvienko Nikodim – student, SPbSUT, St. Petersburg, 193232, Russian Federation, matvienko.na@spbgut.ru

Novikov Maksim – student, SPbSUT, St. Petersburg, 193232, Russian Federation, novikov.mi@spbgut.ru

Kirichek Ruslan – Ph.D., assistant professor, SPbSUT, St. Petersburg, 193232, Russian Federation, ruslan.stk@gmail.com