Министерство науки высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

Факультет «инфокоммуникационных технологий»

Направление подготовки «11.04.02 Инфокоммуникационные технологии и системы связи»



**Реферат**

На тему: «SDN-контроллеры. Сравнительный анализ»

По дисциплине: Основы технологии программно-конфигурируемых сетей

Выполнил:

Студент гр. К41101c:

Фёдоров Никита Константинович

Проверил:

Шкребец Александр Евгеньевич

г. Санкт-Петербург

2021 г.

**Оглавление**

[**ВВЕДЕНИЕ** 3](#_Toc73444154)

[**1. ПОНЯТИЕ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ (SDN)** 5](#_Toc73444155)

[**1.1 Архитектура программно-конфигурируемых сетей** 6](#_Toc73444156)

[**2. SDN КОНТРОЛЛЕРЫ** 10](#_Toc73444157)

[**2.1 Сравнительный анализ существующих контроллеров** 11](#_Toc73444158)

[**2.1.1 Контроллер NOX/POX** 12](#_Toc73444159)

[**2.1.2 Контроллер Beacon** 12](#_Toc73444160)

[**2.1.3 Контроллер OpenDaylight** 13](#_Toc73444161)

[**2.1.4 Контроллер ONOS** 13](#_Toc73444162)

[**2.1.5 Контроллер Floodlight** 14](#_Toc73444163)

[**ЗАКЛЮЧЕНИЕ** 17](#_Toc73444164)

[**СПИСОК ЛИТЕРАТУРЫ** 18](#_Toc73444165)

# **ВВЕДЕНИЕ**

В наши дни большинство устройств связаны друг с другом через Интернет, что приводит к увеличению объема информации, проходящей через сеть. Также развиваются облачные технологии, увеличивается роль мобильных устройств. Центры обработки данных, в которых хранятся данные пользователей, расширяются. Все это побуждает компьютерные сети иметь высокую пропускную способность, повсеместную доступность, мобильность и динамическое управление. Организациям необходимо постоянно обновлять свои устройства, такие как маршрутизаторы и коммутаторы, а также конфигурировать крупномасштабные сети. При этом классическое управление сетями – настройка каждого подключаемого устройства к сети через командную строку и конфигурационные файлы, становиться ограничивающим фактором развития вычислительной инфраструктуры. Возникают трудности при изменении или принятия каких-либо политик, так как администратору необходимо делать изменения на каждом устройстве в сети отдельно, что занимает длительное время.

Ответом на сформулированные выше проблемы стала технология или подход к построению компьютерных сетей – Software Defined Networking (SDN) или программно-конфигурируемые сети.

SDN – это новая сетевая архитектура, в которой управление сетью отделено от пересылки и напрямую программируется. SDN предоставляют согласованное, относительно быстрое управление сетями, разрешая изменения во всей сети с единственной консоли управления. Такая миграция управления, ранее тесно связанного в каждом сетевом устройстве с доступными вычислительными устройствами, позволяет приложениям и сетевым службам абстрагироваться от базовой инфраструктуры и рассматривать сеть как логическую или виртуальную сущность.

Основными идеями при использовании SDN в сравнении с традиционными компьютерными сетями являются следующие [1]:

* разделение процессов передачи и управления данными;
* единый, унифицированный, не зависящий от поставщика интерфейс между уровнем управления и уровнем передачи данных;
* логически централизованное управление сетью, осуществляемое с помощью контроллера с установленной сетевой операционной системой (NOS) и реализованными поверх сетевыми приложениями;
* виртуализация физических ресурсов сети.

# **1. ПОНЯТИЕ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ (SDN)**

Программно-конфигурируемая сеть (Software-defined networking) – это подход к управлению сетью (уровень управления сетью отделён от устройств передачи данных), который обеспечивает динамическую, программно эффективную конфигурацию сети с целью повышения производительности и мониторинга сети.

Ключевые принципы программно-определяемых сетей – разделение процессов передачи и управления данными, централизация управления сетью при помощи унифицированных программных средств, виртуализация физических сетевых ресурсов.

SDN позволяет использовать индивидуальные методы и настройки. При правильной реализации этот процесс дает бизнесу возможность иметь более оперативную сеть, которая реагирует на меняющиеся потребности бизнеса. Это обещает упростить общий контроль над корпоративной сетью. Вдобавок ко всему, реализация программно-определяемых сетей дает еще несколько преимуществ [2]:

1. Снижение эксплуатационных расходов: SDN позволит выполнять многозадачность в сетевых операциях. Это требует меньшего количества дорогого сетевого оборудования. SDN также хорошо адаптируется к виртуализации, что снижает потребность в дополнительном оборудовании.
2. Повышение производительности: так как уровень управления сетью отделён от устройств передачи данных, SDN позволяет гарантировать более высокое качество взаимодействия с пользователем. SDN дает возможность этим устройствам направить все свои ресурсы на ускорение перемещения трафика.
3. Подход к безопасности: гораздо проще контролировать и отслеживать функции безопасности с одной централизованной консоли, чем с нескольких приложений, разбросанных по различным системам. Есть одна центральная точка, с которой можно будет проводить политику безопасности и распространять информацию.
4. Администрирование: возможность управления с одной консоли дает возможность более эффективно контролировать сетевой трафик и распределение задач. С помощью централизованной консоли повышается общая производительность систем и сети.
5. Гарантированная доставка контента: Возможность формировать трафик данных и управлять им – одно из основных преимуществ SDN. Возможность направлять и автоматизировать трафик данных упрощает реализацию качества услуг (QoS) для передачи голоса по IP и мультимедиа.
6. Унификация облака: облако и его ресурсы будут только брать на себя все больше и больше наших повседневных операций. SDN упростят объединение облачных ресурсов несмотря на то, что они находятся в возможно удаленном центре обработки данных. Платформа с массивными компонентами и множеством рабочих частей может управляться с одной централизованной консоли.

## **1.1 Архитектура программно-конфигурируемых сетей**

Компьютерная сеть включает в себя различное количество устройств, таких как маршрутизатор и коммутаторы различных производителей, отвечающих за передачу данных по сети. Традиционные сетевые устройства включают в себя два основных элемента: уровень данных (Data Plane) и уровень управления (Control plane) (Рисунок 1). Уровень данных (уровень пересылки) – функционал, обеспечивающий физическую передачу данных, уровень микросхем и сетевых пакетов. Уровень управления (программный уровень) – интеллект устройства; зависит от конфигурации и типа устройства; знает о топологии; это различные алгоритмы и функционал, задачей которого является автоматическая реакция на изменения трафика [2].



Рисунок 1 – Архитектура традиционной сети и SDN

В архитектуре SDN уровень данных и уровень управления разделены, а логика управления (мозг) реализована в логически централизованном контроллере, поскольку трафик фактически контролируется интеллектуальным контроллером. На рисунке 2 показана архитектура SDN.

Уровни программно-конфигурируемых сетей (SDN) [3,4]**:**

1. Уровень управления в SDN – это контроллер, который отвечает за определение оптимального пути для потоков данных. Основное различие между традиционной архитектурой и архитектурой SDN заключается в том, что уровень управления централизован и может определять оптимальную маршрутизацию от конца до конца.

Контроллер – это специализированное программное обеспечение, установленное на персональный компьютер с предустановленной сетевой операционной системой или сервер. Контроллер предоставляет приложениям сетевые сервисы и программный интерфейс для управления сетевыми элементами и сетью.

Основными функциями контроллера являются:

* Обнаружение устройства конечного пользователя: обнаружение устройств конечного пользователя, таких как ноутбуки, настольные компьютеры, мобильные устройства и т. д.
* Обнаружение сетевых устройств: обнаружение сетевых устройств, включающих инфраструктуру сети, таких как коммутаторы, маршрутизаторы и точки беспроводного доступа.
* Управление топологией: поддержание информации о деталях взаимодействия сетевых устройств друг с другом и с конечным пользователем.
* Управление потоками: ведение базы данных потоков, которыми управляет контроллер, и выполнение необходимых настроек с устройствами, чтобы гарантировать синхронизацию записей потоков устройств с этой базой данных.
1. Уровень данных в SDN представляет из себя набор сетевых элементов (например, коммутаторы и маршрутизаторы) и отвечает за перенос пользовательских данных по сети в соответствии с путем, определенным уровнем управления.
2. Уровень сетевых приложений – набор SDN-приложений, взаимодействующих с SDN-контроллером через программный протокол (API) для сбора, анализа, развёртывания и управления сетевой инфраструктурой на уровне приложений.



Рисунок 2 – Архитектура SDN

Взаимодействие между уровнями обеспечивается северным и южным интерфейсами [5]:

* Северный интерфейс (Northbound interface).

Северный интерфейс – это API, который используется для обмена информацией между контроллером SDN и приложениями, совместимыми с SDN, работающими в сети. API-интерфейсы Northbound, по существу, берут сетевые требования из приложений SDN и согласовывают потребности с сетевым контроллером, который отвечает за предоставление приложениям оптимальных сетевых ресурсов и путей.

В противоположность южному интерфейсу северный интерфейс – это в основном программная система, в которой приложения с маршрутизацией создаются программно с помощью таких языков программирования, как Python или Java, что позволяет ускорить разработку, снизить инвестиционные затраты и упростить устранение неисправностей по сравнению с API южного направления.

* Южный интерфейс (Southbound interface).

Южный интерфейс – это связывающее соединение между средствами управления и сетевыми устройствами. Как только северный интерфейс сообщает контроллеру, какие возможности необходимы приложению SDN, контроллер должен сообщить аппаратному обеспечению коммутатора SDN, как обрабатывать поток данных. OpenFlow используется в качестве южного интерфейса для передачи этой информации между контроллером и аппаратным обеспечением коммутатора, совместимого с SDN.

# **2. SDN КОНТРОЛЛЕРЫ**

Контроллер SDN – это приложение, которое работает как мозг сети и действует как стратегическая контрольная точка в сети SDN.

Контроллеры SDN – это своего рода операционная система для сети, в которой любые коммуникации между приложениями и устройствами должны проходить через контроллер. Контроллер находится между сетевыми устройствами на одной стороне и прикладным уровнем на другой стороне, транслирует требования с уровня приложения и управляет управлением потоком на сетевые устройства (через южные API-интерфейсы) и предоставляет приложению SDN абстрактное представление сети и бизнес-логики (через северные API).

Контроллер SDN определяет потоки данных, которые происходят на уровне данных SDN. Каждый поток через сеть должен сначала получить разрешение от контроллера в соответствии с сетевой политикой. Если контроллер разрешает поток, он вычисляет маршрут для потока и добавляет запись для этого потока в каждый из коммутаторов на пути. Поскольку все сложные функции выполняются контроллером, коммутаторы просто управляют таблицами потоков, записи которых могут быть заполнены только контроллером. Для связи между контроллером и коммутаторами используется стандартизированный протокол и API.

Контроллер SDN служит своего рода операционной системой (ОС) для сети. Все коммуникации между приложениями и устройствами должны проходить через контроллер. Протокол OpenFlow соединяет программное обеспечение контроллера с сетевыми устройствами, чтобы серверное программное обеспечение могло указывать коммутаторам, куда отправлять пакеты. Контроллер использует протокол OpenFlow для настройки сетевых устройств и выбора наилучшего пути для трафика приложений. Поскольку план управления сетью реализован в программном обеспечении, а не во встроенном ПО аппаратных устройств, сетевой трафик может управляться более динамично и на гораздо более детальном уровне [6].

В архитектуре SDN-коммутатор выполняет следующие функции:

* Коммутатор инкапсулирует и перенаправляет первый пакет потока в контроллер SDN, чтобы контроллер мог решить, следует ли добавить поток в таблицу потока коммутатора.
* Коммутатор перенаправляет входящие пакеты из соответствующего порта на основе таблицы потоков. Таблица потоков может включать информацию о приоритете, продиктованную контроллером.
* Коммутатор может отбрасывать пакеты в определенном потоке, временно или постоянно, как это продиктовано контроллером. Выброс пакетов может использоваться для целей безопасности, сдерживания атак типа отказ в обслуживании (DoS) или требований к управлению трафиком.

## **2.1 Сравнительный анализ существующих контроллеров**

На сегодняшний день существует множество реализаций контроллеров. Ниже рассмотрены существующие контроллеры с открытым исходным кодом.

## **2.1.1 Контроллеры NOX/POX**

**NOX** – первый OpenFlow контроллер. NOX был разработан инженерами Nicira Networks параллельно с протоколом OpenFlow. Первоначально был написан с поддержкой двух языков: C++ и Python. В 2008 г. NOX был опубликован под лицензией GPL и с тех пор этот контроллер является базовым для многих научно-исследовательских групп, которые только приступают к изучению SDN. NOX ориентирован на дистрибутивы Linux (в частности Ubuntu 11.10 и 12.04, но также возможно использование на Debian и RHEL 6). Содержит сервисы для построения топологии сети и L2-L3 коммутации.

В процессе тестирования выяснилось, что поддержка двух языков сильно сказывается на производительности, поэтому часть, которая отвечает за Python, вытащили в отдельный проект, который позже назвали POX [7].

**POX** – контроллер POX разработчики называют «младшим братом» NOX. Если при разработке NOX основной целью была высокая производительность, то POX в первую очередь направлен на обучение и исследования. По своей сути POX – это платформа для быстрой разработки и прототипирования ПО управления сетью. Этот контроллер написан на Python, его легко запустить на Windows, Linux и Mac OS. К примеру, исследовательская группа в Стэнфорде использует POX для исследования ключевых проблем ПКС. POX находится в стадии активного развития: все удачные идеи постоянно перемещаются из лабораторных экспериментов в официальные релизы контроллера POX (по крайней мере так утверждают его разработчики из Стэнфорда).

POX поддерживает те же компоненты, графический интерфейс, средства визуализации, как и NOX.

## **2.1.2 Контроллер Beacon**

 **Beacon** – это достаточно быстрый, кросс-платформенный, модульный OpenFlow контроллер на Java. Beacon используется во многих научно-исследовательских проектах и тестовых внедрениях/развертываниях.

Beacon написан на Java и работает на многих платформах, начиная от высокопроизводительных многоядерных Linux-серверов до смартфонов на Android [7].

## **2.1.3 Контроллер OpenDaylight**

**OpenDaylight** – проект с открытым исходным кодом под эгидой Linux Foundation, направленный на содействие внедрению и развитию программно-определяемых сетей (SDN) путем создания общей отраслевой платформы.

OpenDaylight – это промышленное ПО на основе Java, управляемое консорциумом Linux Founsion включающее около 50 корпоративных представителей, таких как Brocade, Cisco, Citrix, Dell, Ericsson, HP, IBM, Juniper, Microsoft и Red Hat. Благодаря тому что OpenDaylight – открытая платформа, представители сообщества, конечные пользователи и даже клиенты могут участвовать в определении, анализе, разработке и тестировании его архитектуры и существующих модулей. Более того, они могут внести свой вклад, предлагая новые инициативы и представляя новые предложения техническому сообществу, уделяющему особое внимание OpenDaylight. Задача проекта OpenDaylight – создать совместное сообщество, способствующее успеху и принятию SDN.

## **2.1.4 Контроллер ONOS**

**Контроллер ONOS** (Java, OpenFlow 1.0 и 1.3) – это распределенная сетевая операционная система с открытым исходным кодом для ПКС операторов связи. Высокая производительность платформы управления ONOS достигается за счет разделения сети на сегменты, их распределения между экземплярами контроллера и использования многопоточной обработки сообщений. Для достижения высокой готовности платформа управления ONOS использует избыточные защищенные каналы связи между коммутаторами и контроллерами, глобальное распределение состояния сети между контроллерами платформы, распределение ролей контроллера для каждого коммутатора и процедуру изменения роли контроллера в случае отказа основного контроллера. Однако ONOS не поддерживает процедуру балансировки нагрузки между экземплярами контроллера.

## **2.1.5 Контроллер Floodlight**

**FloodLight** – это контроллер корпоративного уровня, разработан на основе котроллера Beacon. FloodLight написан на Java и имеет лицензию Apache. Floodlight – один из значительных вкладов Big Switch Networks (BNC) в сообщество разработчиков ПО с открытым исходным кодом. FloodLight также как и другие контроллеры на Java является модульным (т. е. перед запуском в специальный конфигурационный файл добавляются названия сервисов, которые будут использованы), что очень удобно для разработчиков. FloodLight поддерживает широкий спектр виртуальных и физических коммутаторов, способен поддерживать смешанные OpenFlow сети и сети традиционной архитектуры. Этот контроллер имеет статический Flow Pusher API, который способствует эффективному управлению сетью.

Ниже в таблице 1 представлен сравнительный анализ существующих контроллеров с открытым исходным кодом. Критериями при выборе контроллеров были: поддерживаемые версии протокола OpenFlow, производительность, готовый функционал, сложность установки, документация, серверные интерфейсы взаимодействия [1,3,6].

Таблица 1 – Сравнительный анализ существующих контроллеров с открытым исходным кодом

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Контроллер** | **ONOS** | **OpenDayLight** | **FloodLight** | **Beacon** | **POX** | **NOX** |
| **Язык программирования** | Java | Java | Java | Java | Python | C++ |
| **Графический интерфейс пользователя** | Web Based | Web Based | Web/Java Based | Web Based | Python | Python |
| **Документация** | Хорошая | Очень хорошая | Хорошая | Хорошая | Плохая | Плохая |
| **Поддерживаемые платформы** | Linux, Mac OS, Windows | Linux, Mac OS, Windows | Linux, Mac OS, Windows | Linux, Mac OS, Windows | Linux, Mac OS, Windows | Most supported on Linux |
| **Производительность** | Очень хорошая | Очень хорошая | Очень хорошая | Очень хорошая | Хорошая | Очень хорошая |
| **OpenFlow** | 1.0, 1.3 | 1.0, 1.3, 1.4 | 1.0, 1.3 | 1.0 | 1.0 | 1.0 |

Важно отметить, что ONOS и OpenDaylight являются наиболее популярными контроллерами. Эти два написанных на Java контроллера обеспечивают кроссплатформенность и предоставляют высокую модульность, используя контейнер OSGI, который позволяет загружать модули не перезагружая контроллер. Наследуя мощь Java в программировании графических пользовательских интерфейсов, они предоставляют хорошую возможность графического интерфейса. Будучи в партнерстве с известными сетевыми провайдерами и исследовательскими сообществами, они имеют не только хорошую расширенную документацию, но и стратегию развития с подробным планом релизов. Кроме того, их поддержка распределенной схемы делает их способными проводить развертывание в реальном масштабе SDN. Контроллер ONOS в основном предназначен для сетей операторов связи. Это дает им возможность предоставлять новые услуги SDN наряду с их первоначальными проприетарными услугами.

Архитектура ONOS предназначена для поддержки высокоскоростных и крупномасштабных сетей. Его отличительной особенностью является поддержка гибридных сетей.

OpenDaylight был в основном сосредоточен для использования в центрах обработки данных, но его последний выпуск (Lithium) показал возможность поддержки различных приложений. Были добавлены многие южные и северные интерфейсы, такие как COAP, PCEP, LACP, OpFlex, SNMP и т. д. Также были реализованы новые модули, такие как IoT data broker (IoTDM), унифицированный безопасный канал USC и т.д. Таким образом, это первый контроллер, который можно использовать для IoT.

# **ЗАКЛЮЧЕНИЕ**

В сложных распределенных масштабируемых сетях происходят быстрые изменения. Из этого следует проблема в управлении такими сетями. Средства управления такими сетями должны отвечать следующим критериям: безопасность, масштабирование, управляемость, гибкость.

Отличным решением являются программно-конфигурируемые сети (Software Defined Networking (SDN)). SDN – это подход к управлению сетью (уровень управления сетью отделён от устройств передачи данных), который обеспечивает динамическую, программно эффективную конфигурацию сети с целью повышения производительности и мониторинга сети. Ключевые принципы программно-определяемых сетей – разделение процессов передачи и управления данными, централизация управления сетью при помощи унифицированных программных средств, виртуализация физических сетевых ресурсов.

Контроллер – это специализированное программное обеспечение, установленное на персональный компьютер с предустановленной сетевой операционной системой или сервер. Контроллер предоставляет приложениям сетевые сервисы и программный интерфейс для управления сетевыми элементами и сетью. Основными функциями контроллера являются: обнаружение устройства конечного пользователя, обнаружение сетевых устройств, управление топологией, управление потоками.

Был проведен сравнительный анализ существующих SDN-контроллеров с открытым исходным кодом.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Семеновых А.А, Лапонина О.Р. Сравнительный анализ SDN-контроллеров // International Journal of Open Information Technologies. – 2018. – 7. – 50-56.
2. Hend Abdelagder Eissa, Kenz A. Bozed, Hadil younix. Software Defined Networking // International conference on Sciences and Techniques of Automatic control & computer engineering (STA). – October 2019.
3. Saleh Asadollahi, Dr. Bhargavi Goswami, Dr. Atul M Gonsai. Software Defined Network, Controller Comparison // International Journal of Innovative Research in Computer and Communication Engineering. – Vol.5. – April 2017.
4. Клепче В. Программно-определяемые сети сегодня // Connect WIT. – №1-2. – 2019.
5. Панеш А. Х. Содержание и перспективы технологий программно-конфигурируемых сетей и виртуализации сетевых функций // Вестник Адыгейского Государственного университета. – 2014. – 2. – 120-127.
6. Ola Salman, Imad H. Elhajj, Ayman Kayssi, Ali Chehab. SDN controllers: A comparative study // 18th Mediterranean Electrotechnical Conference (MELECON). – April 2016.
7. ПКС-контроллеры с открытым кодом [Электронный ресурс]. Режим доступа: <https://habr.com/ru/post/169403/>