

Лекция №5: Программно конфигурируемые сети (SDN)

План лекции:

- I Основные понятия SDN.
- II Отличие SDN от традиционных сетей связи.
- III Архитектура SDN.
- IV Контроллеры и сетевые операционные системы SDN.
- V Взаимодействие 5G и SDN.



I Основные понятия SDN

Программно-конфигурируемые сети (ПКС) – новый этап в развитии телекоммуникаций. В англоязычной интерпретации – **программно-определяемые сети SDN** (англ. *Software-Defined Networking*).

Разработкой стандартов SDN и исследованиями в этом направлении занимается множество отраслевых организаций и международных организаций.

Основные из них:

- **фонд открытых сетевых технологий ONF** (англ. *Open Network Foundation*);
- **рабочая группа по инженерным задачам Интернет IETF** (англ. *Internet Engineering Task Force*);
- **международный союз электросвязи, сектор телекоммуникаций ITU-T** (англ. *International Telecommunication Union Telecommunication Standardization Sector*).

Каждая из этих организаций даёт собственное определение SDN.

Определения SDN:

Определение ONF: SDN – динамичная, управляемая и адаптируемая сетевая архитектура, в которой разделены уровни управления сетью и передачи данных, что обеспечивает программное управление сетью и абстрагирование/изоляцию (уровня) сетевой инфраструктуры от (уровня) приложений и сетевых услуг (сервисов).

Определение IETF: SDN – подход к построению сетей, обеспечивающий

прямое управление ресурсами и сетями, а также их распределение за счёт добавления собственных средств обработки, администрирования и программного управления посредством открытых сетевых интерфейсов и абстракции (абстрагирования, изоляции) уровня сети.

Определение ITU: SDN – технология построения сетей, которая позволяет реализовать централизованный, программируемый уровень управления и изоляцию (абстракцию) уровня данных, при этом уровни управления и данных разделены, благодаря чему операторы сетей связи могут напрямую управлять своими виртуальными ресурсами и сетями.

Историческая справка

Концепция новой сетевой архитектуры, которая позднее получила название программно конфигурируемых сетей, появилась в 1993 г.

В 2007 г. преподавателями США Мартином Касадом, Ником Мак-Кеоном, Скоттом Шенкером разработан новый сетевой стандарт – OpenFlow, который позволяет осуществлять удалённое управление уровнем передачи данных – начало реализации SDN.

Историю SDN условно можно разбить на три этапа:

- середина 1990-х – начало 2000-х: АКТИВНЫЕ СЕТИ. В сети вводятся функции программирования.
- 2001 – 2007 гг.: РАЗДЕЛЕНИЕ ПЛОСКОСТЕЙ управления и передачи данных. Открытые интерфейсы между плоскостями (на основе OpenFlow).
- 2007 – 2010 гг.: РАЗРАБОТКА API OPENFLOW и сетевых операционных систем. В эти годы начинается распространение SDN.

Ключевые принципы программно-определяемых сетей:

- *разделение* процессов передачи и управления данными;
- *централизация* управления сетью при помощи унифицированных программных средств;
- *виртуализация* физических сетевых ресурсов.

II ОТЛИЧИЕ SDN ОТ ТРАДИЦИОННЫХ СЕТЕЙ СВЯЗИ

Основные тренды развития инфокоммуникационных технологий:

- стремительный рост объёмов трафика и изменение его структуры в сторону передачи видео и унифицированных коммуникаций (UC-C);
- необходимость поддержки мобильных пользователей и социальных сетей;

- высокопроизводительные кластеры для обработки Больших Данных (англ. *Big Data*);
- сложность современных сетей и как результат сложность их масштабирования;
- зависимость от вендоров;
- виртуализация для предоставления облачных сервисов (англ. *Cloud Bursting*).

2.1 Особенности традиционных сетей связи

Традиционные сети статичны и не соответствуют быстрой динамике развития современного IT бизнеса. Кроме того, сдерживает развитие инфокоммуникационных систем привязка к выбранному сетевому производителю, т.к. не гарантирует поддержку будущих приложений и сервисов.

Проблему развитию сетей создаёт и традиционный стек протоколов TCP/IP, поскольку является громоздкой и негибкой системой управления компьютерной сетью, она и «думает», и «делает»: сначала решает задачу построения маршрута, а потом сама же прокладывает этот маршрут.

Получается, что в современных сетях функции управления и передачи данных совмещены, что делает контроль и управление очень сложным. Такой подход накладывает серьёзные ограничения на работу с ресурсами сети.

До недавнего времени действующая архитектура сетей развивалась по методу «ласточкиного гнезда», т.е. по мере выявления проблем к стеку протоколов TCP/IP добавлялся новый, который эту проблему решал.

Однако, архитектура с каждым годом становится всё сложнее и, как результат, не выдерживает объёмов передаваемой информации, не решает всех проблем виртуализации и безопасности (работая на основе более 600 протоколов).

Традиционное сетевое устройство

Традиционное сетевое устройство состоит из трёх компонентов (рис. 5.1):

1) **Уровень управления** – это CLI, встроенный веб-сервер или API и протоколы управления. Задача этого уровня обеспечить управляемость устройством.

2) **Уровень управления** трафиком – это различные алгоритмы и функционал задач, которого является автоматическая реакция на изменения трафика, т. е. интеллект устройства.

3) **Передача трафика** – функционал, обеспечивающий физическую передачу данных, уровень микросхем и сетевых пакетов.

2.2 Особенности SDN

SDN – это новая, простая в управлении, гибкая и экономически

эффективная сетевая архитектура, обеспечивающая высокую пропускную способность и динамичность, что принципиально важно для современных приложений.

В ПКС *уровень управления* сетью отделён от *устройств передачи данных* и реализуется *программно*, что является одной из форм виртуализации сетевых ресурсов.



Рис. 5.1. Архитектура традиционного сетевого устройства

Весь интеллект (MANAGEMENT PLANE и CONTROL PLANE) переносится в отдельное центральное устройство – *контроллер SDN*.

В результате «новый» роутер или коммутатор обслуживает только поток данных (уровень передачи трафика DATAPLANE), становится более простым соответственно более дешёвым.

III АРХИТЕКТУРА SDN

Наиболее распространённой считается трёхуровневая архитектура, продвигаемая консорциумом ONF (рис. 5.2).

3.1 Уровень приложений

Уровень охватывает решения, направленные на расширение сетевых услуг. Эти решения в основном представляют собой программные приложения, которые напрямую взаимодействуют с контроллером через «северный» интерфейс NBI (англ. *North Bound Interface*), например, мониторинг, аналитика, бизнес-приложения и др.

Приложения на высоком уровне абстракции определяют и задают политики работы сети через централизованный SDN-контроллер, например, передают определенный трафик из точки А в точку Б по самому незагруженному пути.

Приложения могут быть запущены, в том числе, и на самом SDN-

контроллере, самые простые примеры таких приложений: коммутация, маршрутизация и т.п.

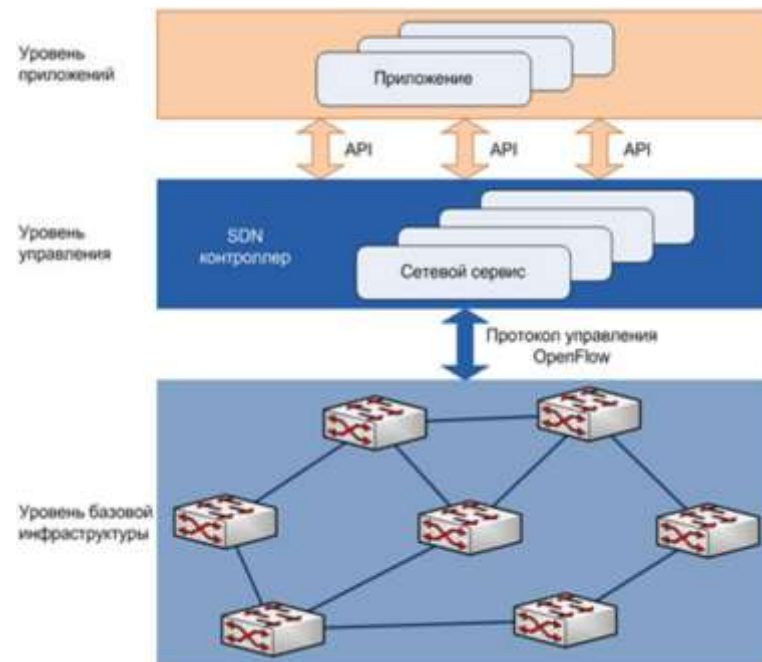


Рис. 5.2. Архитектура SDN

3.2 Уровень управления

SDN-контроллер на основании политик, получаемых от уровня приложений, вычисляет и задаёт сетевым устройствам правила действий по коммутации/маршрутизации трафика.

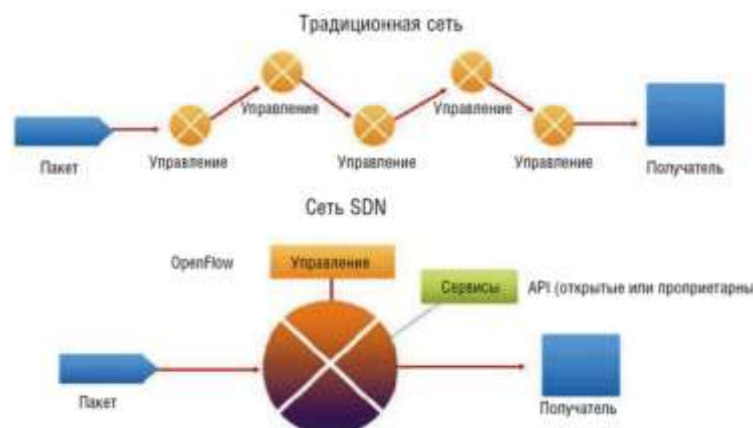


Рис. 5.3. Отличие плоскости управления в традиционной сети и SDN

3.3 Уровень инфраструктуры

Уровень реализуется через *физическое сетевое оборудование*, которое включает в себя *среду передачи* и *сетевые устройства* (OpenFlow-коммутаторы).

Взаимодействие уровня инфраструктуры с контроллером осуществляется через «южный» интерфейс, который представляет собой *протокол OpenFlow*.

Архитектура SDN является:

- ***программируемой***: управление сетью программируется напрямую, поскольку этот уровень отделен от функций передачи данных;
- ***адаптивной***: отделение функций контроля от функций передач данных позволяет администраторам динамически настраивать транспортные потоки по всей сети для удовлетворения меняющихся потребностей;
- ***централизованно управляемой***: интеллектуальный центр управления сетью логически централизован в программных SDN-контроллерах, которые дают общее представление о состоянии сети. В свою очередь для приложений и политик обработки контроллеры являются едиными логическими коммутаторами.
- ***программно конфигурируемой***: SDN позволяет сетевым администраторам конфигурировать, управлять, обеспечивать защиту и быстро оптимизировать сетевые ресурсы с помощью динамических, автоматизированных программ SDN, которые они могут писать самостоятельно;
- ***основанной на открытых стандартах*** и независимой от вендоров: при реализации на основе открытых стандартов, SDN значительно упрощает проектирование и эксплуатацию сети, поскольку управление сетью обеспечивается не устройствами и протоколами определенных производителей, а программными SDN контроллерами.

3.4 Три подхода к реализации архитектуры SDN

Три основных подхода к реализации сети:

- ***Open SDN***;
- ***SDN via API***;
- ***SDN via Overlay***.

Архитектура подхода Open SDN

Open SDN реализуется на базе протокола OpenFlow, отсюда и первое слово в названии. Это академический подход к SDN.

Основная концепция Open SDN – полное удаление плоскости контроля (англ. *Control Plane*) с сетевого оборудования. Плоскость контроля переносится на центральный SDN-контроллер (рис. 5.4).

Передача (англ. *forwarding*) трафика на сетевом оборудовании осуществляется на основании правил таблиц потоков (англ. *Flow Table*). Таблицы потоков загружаются с SDN-контроллера, а он, обладая полным знанием топологии сети и на основании заданных политик от SDN-приложений, формирует таблицы потоков для каждого сетевого элемента.

Сетевые элементы, согласно правилам в таблице потоков, осуществляют определенные действия над входящими пакетами – пересылают на нужный порт, сбрасывают, проверяют и т. п.

Взаимодействие контроллера SDN с сетевыми элементами обеспечивается через «южный» интерфейс SBI (англ. *South Bound Interface*) по стандартизированному открытому протоколу OpenFlow.

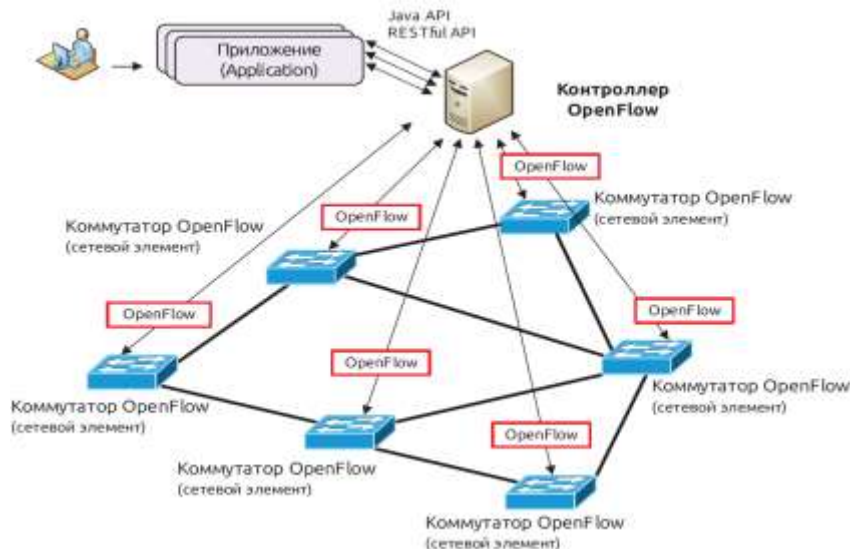


Рис. 5.4. Архитектура Open SDN

Протокол OpenFlow представляет собой открытый стандарт, поддерживаемый организацией ONF.

Сетевое оборудование для работы в Open SDN именуется OpenFlow-коммутаторами/маршрутизаторами. На рынке представлено достаточно большое количество OpenFlow коммутаторов, как программных (например, Open vSwitch), так и аппаратных.

Отличительной особенностью подхода Open SDN на базе OpenFlow является возможность прямого доступа и изменения плоскости передачи (англ. *Forwarding Plane*) на сетевых устройствах.

SDN via API

Основное отличие SDN via API от Open SDN заключается в сохранении плоскости контроля на сетевых элементах. В данном подходе сетевой элемент – это классический маршрутизатор или коммутатор, в котором присутствуют и уровень контроля, и уровень передачи (рис. 5.5).

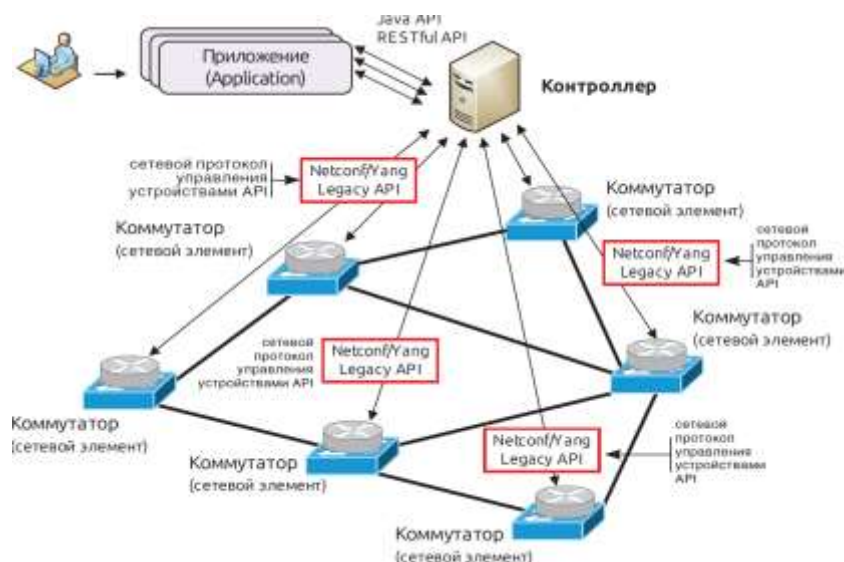


Рис. 5.5. Архитектура SDN via API

«Программирование» сетевой топологии реализуется не через прямое влияние на плоскость передачи, как в Open SDN, а следующими двумя способами:

- через управление конфигурацией оборудования – подход Management Plane;
- через влияние на таблицы маршрутизации/коммутации сетевого элемента – подход Control plane.

SDN-контроллер взаимодействует с сетевым элементом либо через существующие протоколы (CLI, SNMP, PCEP и т. д.), либо через новые или усовершенствованные (Netconf/Yang, I2RS).

«Северный» интерфейс сохраняет своё назначение.

Примером реализации контроллера SDN является контроллер OpenDaylight.

OpenDaylight – это открытый проект, созданный консорциумом ведущих игроков телекоммуникационного рынка (Cisco, Juniper, Ericson, Microsoft, VMware, NEC и т. д.).

Основное преимущество данного подхода – использование существующего оборудования и возможность реализации концепции SDN на существующих традиционных сетях.

К минусам данного подхода можно отнести

- отсутствие абстракции;
- необходимость конфигурировать каждый элемент сети;
- необходимость синхронизации между плоскостью контроля (Control Plane) контроллера и распределёнными плоскостями контроля каждого сетевого элемента.

С одной стороны, SDN via API служит для защиты сферы своего влияния крупными производителями сетевого оборудования, с другой – представляет собой движение к централизации и программному конфигурированию сети.

Данное решение многими рассматривается как эволюционный переходный процесс от традиционных сетей к Open SDN.

SDN via Overlay

SDN via Overlay – это одно из самых часто реализуемых решений на базе SDN. Решение SDN via Overlay, если более точно – **SDN via hypervisor-based Overlays**, нашло своё применение в дата-центрах, использующих на своей серверной инфраструктуре технологии виртуализации VM (англ. *Virtual Machine*).

Суть подхода заключается в следующем (рис. 5.6 – прошу прощения за ошибки в подписях на рисунке, т.к. рисунок взят из статьи).

Физическая underlay-сеть реализуется на базе классических коммутаторов/маршрутизаторов, поверх физической сети строятся overlay-сети.

Физическая сеть обеспечивает только L2/L3 (модель OSI) связность между гипервизорами (физическими серверами).

В качестве аналогии overlay- и underlay-сетей можно привести пример работы IPsec-туннелей поверх WAN-сети. IPsec-туннели – это виртуальная overlay-сеть, наложенная на физическую underlay WAN-сеть.

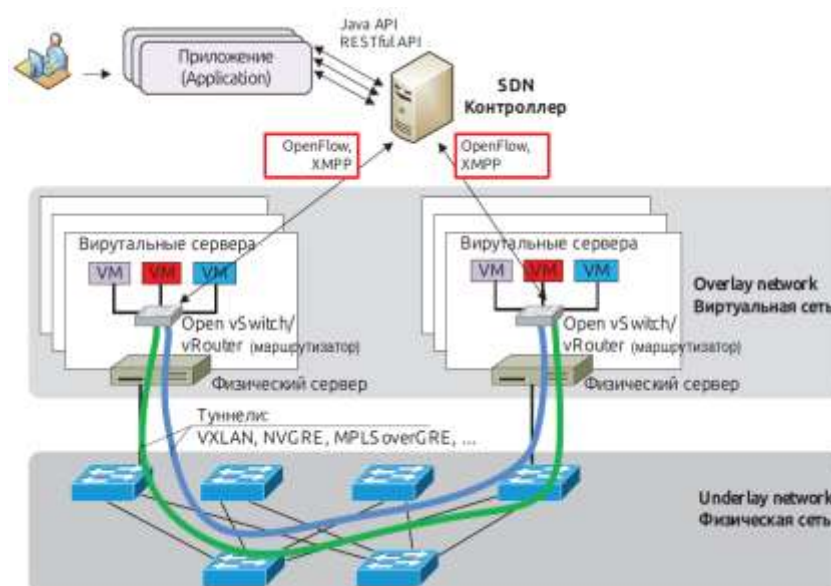


Рис. 5.6. Архитектура SDN via Overlay

Overlay-сеть в архитектуре SDN представляет собой виртуальную топологию из виртуальных же коммутаторов/ маршрутизаторов, соединённых между собой виртуальными каналами «точка-точка» (туннелями). В качестве туннельного механизма в основном используется MAC-in-IP-туннелирование.

Множественные оверлейные сети могут существовать независимо друг от друга и одновременно работать в одной физической сети.

В качестве виртуальных коммутаторов/маршрутизаторов используются такие решения, как Open vSwitch, vRouter и т.п. Виртуальные коммутаторы/маршрутизаторы именуются **Virtual Tunnel Endpoint (VTEP)**.

«Южный» интерфейс взаимодействия между контроллером и виртуальными коммутаторами может быть реализован на таких протоколах, как:

- OpenFlow;
- Extensible Messaging and Presence Protocol (XMPP);
- проприетарные протоколы, например, OpenFlex от Cisco.

Подход SDN over Overlay позволяет быстро, динамично, по запросу создавать произвольные overlay-топологии без внесения изменений в сетевое оборудование. При этом наложенные сети полностью прозрачны для сетевого оборудования.

К недостаткам данного подхода относятся:

- существование underlay-сети вне управления контроллером;
- ручное конфигурирование сетевых устройств;
- неэффективное использование физической инфраструктуры.

IV КОНТРОЛЛЕРЫ И СЕТЕВАЯ ОПЕРАЦИОННАЯ СИСТЕМА SDN

Уровень управления реализуется на SDN-контроллере через *сетевую операционную систему* NOS (англ. *Network Operation System*).

4.1 Понятие сетевой операционной системы

NOS – сетевая операционная система обеспечивает приложениям доступ к управлению сетью и отслеживает конфигурацию сетевых устройств (рис. 5.7).



Рис. 5.7. Место NOS в архитектуре SDN

Решение задач управления сетью выносится на уровень приложений, реализованных на основе API сетевой операционной системы. Приложения управляют разными аспектами функционирования сети, включая построение топологии, принятие маршрутизирующих решений, балансировку нагрузки.

Сетевая ОС обеспечивает приложениям доступ к управлению сетью и постоянно отслеживает конфигурацию средств сети. В отличие от

традиционного толкования термина ОС, под NOS понимается программная система, обеспечивающая мониторинг, доступ и управление ресурсами сети.

NOS обеспечивает программный интерфейс для приложений управления сетью и реализует механизмы управления таблицами коммутаторов: добавление, удаление, модификацию правил и сбор разнообразной статистики.

Таким образом, NOS организует низкоуровневое управление ресурсами сети, а приложения – высокоуровневое, не зависящее от базовой конфигурации сети. Например, приложения имеют дело с доменными именами, именами хостов и абонентов, а контроллер имеет дело с IP-адресами хостов и серверов и MAC-адресами устройств.

Резюмируя, можно дать следующие определения.

Контроллер – это выделенный сервер, на котором работает специальное программное обеспечение, состоящее из сетевой операционной системы и сетевых управляющих приложений.

Сетевая операционная система (NOS) представляет собой фреймворк, который взаимодействует с коммутирующим оборудованием и предоставляет интерфейс для сетевых приложений для контроля и управления сетью целиком.

Сетевая операционная система не управляет сетью сама по себе, она предоставляет интерфейс прикладного программирования для сетевых приложений, которые уже в свою очередь реализуют функциональность управления сетью.

4.2 Классификация контроллеров SDN

Рис. 5.8 содержит классификацию контроллеров SDN.

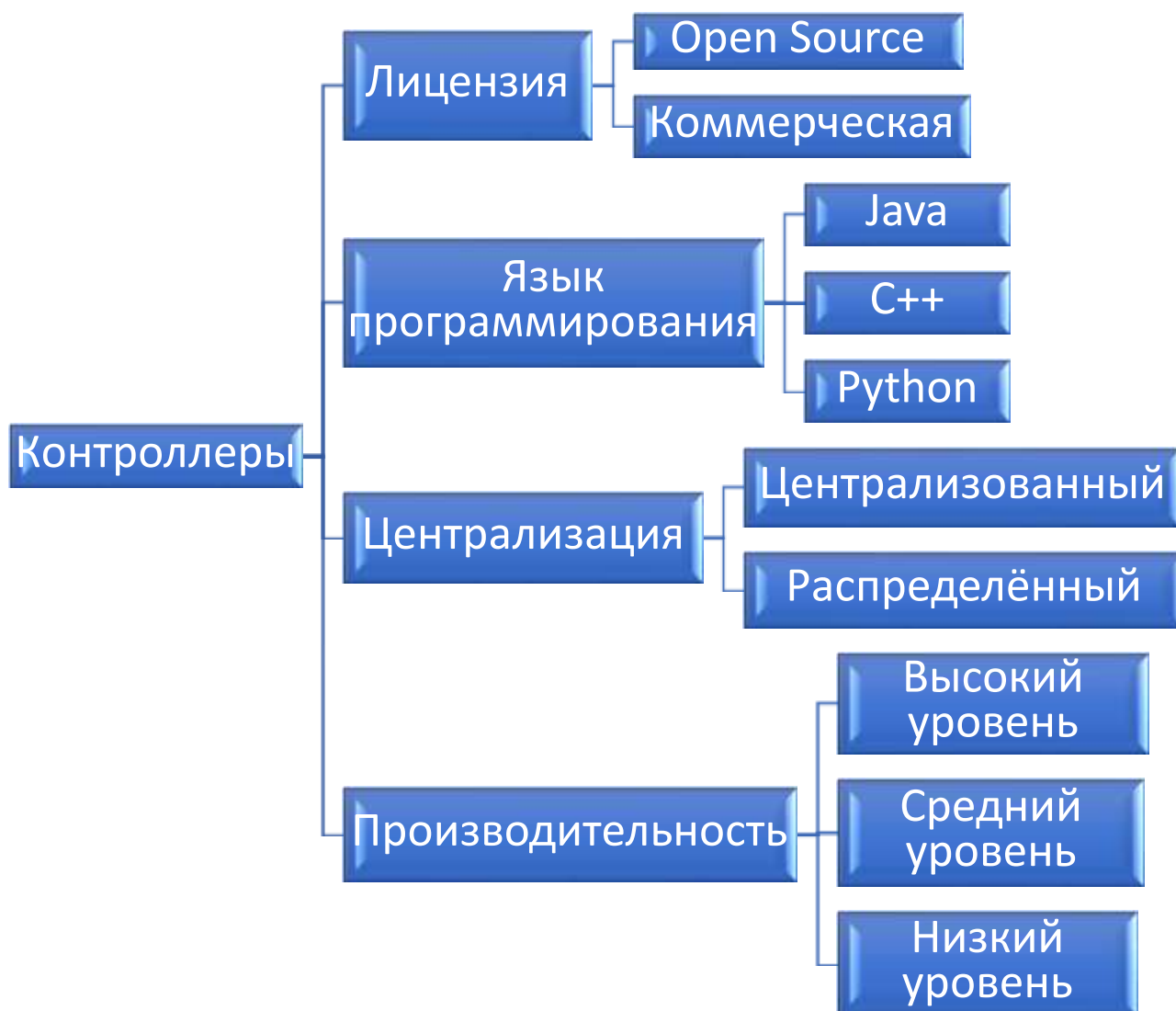


Рис. 5.8. Классификация контроллеров SDN

4.3 Список контроллеров SDN

Classic NOX	NOX	Iris
POX	ONOS	ONIX
Beacon	Rosemary	HyperFlow
Trema	Libfluid_msg	HP VAN SDN
Maestro	BigSwitch	DISCO
FloodLight	Ryu NOS	PANE
FlowER	Meridian	Ravel
Mul	ProgrammableFlow	SMaRtLight
SDN Unified Controller)	OpenDaylight	yanc

4.4 NOS Open Source

Рассмотрим SDN-решения с открытым исходным кодом (англ. *Open Source*).

Диаграммы количества внедрённых в эксплуатацию некоммерческих NOS приведены на рис. 5.8.

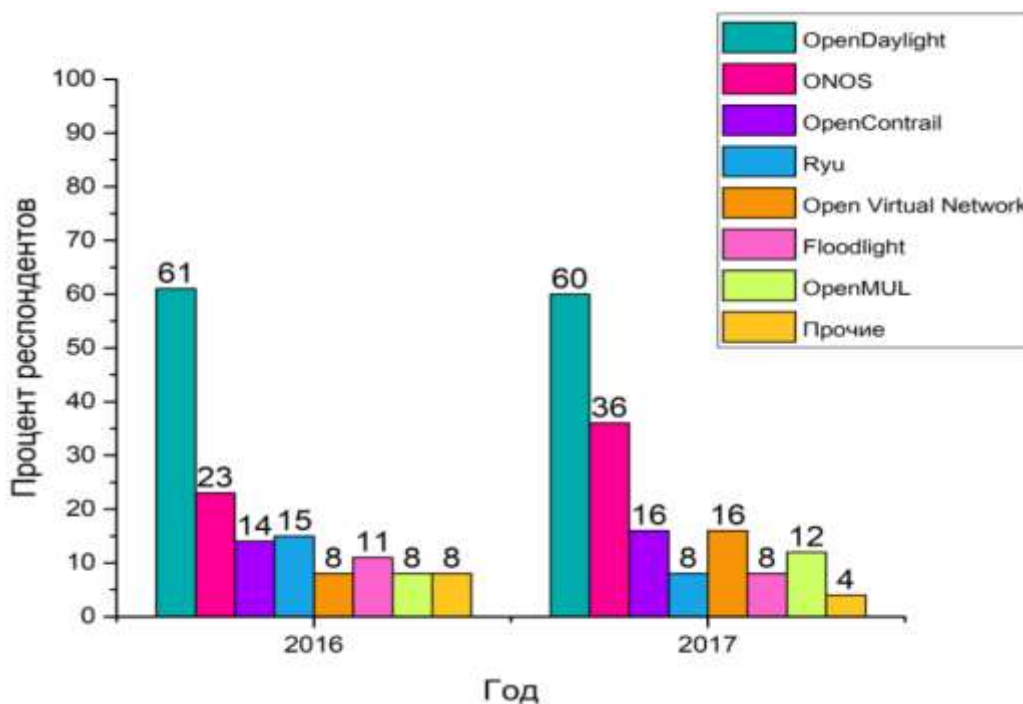


Рис. 5.8. Популярные сетевые операционные системы с открытым исходным кодом, внедрённые в эксплуатацию

OpenDaylight

Тип лицензии: Eclipse Public License (EPL).

Кем разрабатывается: Linux Foundation при участии компаний IBM, Juniper Networks, Cisco, Red Hat, VMware, Citrix, Ericsson, Microsoft и NEC.

Где применяется: Orange, China Mobile, AT&T, T-Mobile, Telefonica, China Telecom, Deutsche Telekom.

Особенности: кроссплатформенное программное обеспечение, способное функционировать в среде любой операционной системы или на аппаратной платформе, поддерживающей виртуальную машину Java (JVM).

OpenDaylight существует в трёх редакциях:

- *Base Edition*;
- *Virtualization Edition*;
- *Service Provider Edition*.

Базовая редакция (англ. *Base Edition*) предоставляет необходимый минимальный костяк и, сама по себе, может быть интересна в академических целях или в качестве прототипа для создания собственных решений SDN и OpenFlow.

Редакция для виртуализации (англ. *Virtualization Edition*) предназначена для дата-центров и кроме состава базовой версии включает дополнения для создания и управления виртуальными сетями, а также набор приложений для управления безопасностью и параметрами сети.

Редакция для сервис-провайдеров (англ. *Service Provider Edition*) нацелена на использование в существующих сетях провайдеров и операторов связи, желающих перейти на использование SDN и NFV.

На рис. 5.9 представлена Архитектура OpenDaylight версии Base Edition.



Рис. 5.9. Архитектура OpenDaylight Base Edition

Архитектура СОС является модульной и включает в себя следующие компоненты:

- **Clustering Manager** – модуль, управляющий общим кэшем экземпляров контроллеров в кластере;
- **Container Manager** – модуль, управляющий сетевыми срезами (англ. Network slices);
- **Switch Manager** – модуль обработки информации об устройствах на южном интерфейсе;
- **Statistic Manager** – модуль сборки статистической информации;
- **Topology Manager** – модуль, отвечающий за построение топологии сети;
- **Host Tracker** – модуль, отслеживающий состояние подключённых конечных узлов;
- **Forwarding Rules Manager** – модуль установки потоков на

коммутационных узлах южного интерфейса;

- **ARP Handler** – модуль обработки ARP-сообщений;
- **Forwarding Manager** – модуль маршрутизации, вычисляющий маршрут передачи данных.

Эти модули динамически связываются в *Service Abstraction Layer* (SAL) – связующее программное обеспечение, определяющее способы выполнения запросов приложений, поступающих с северного интерфейса, и их преобразование в сообщения протоколов южного интерфейса, отвечающего за взаимодействие с сетевым оборудованием. Другими словами: SAL транслирует запросы конкретного приложения в команды, понятные сетевым узлам.

Южный интерфейс (англ. *Southbound Interface*), отвечающий за взаимодействие с сетевой инфраструктурой, поддерживает множество протоколов посредством плагинов.

Северный интерфейс взаимодействия с приложениями представлен в виде REST API (англ. *REpresentational State Transfer*).

Состав Base Edition:

- модульный, масштабируемый и поддерживающий разные протоколы SDN-контроллер, соответствующий спецификации OSGi (англ. Open Services Gateway Initiative);
- плагин для интеграции поддержки протокола OpenFlow через абстрактный SAL-интерфейс SDN-контроллера (англ. *Service Abstraction Layer*);
- библиотека с реализацией протокола OpenFlow 1.3;
- компонент с поддержкой протокола для управления и конфигурации Open vSwitch (англ. *Open vSwitch Database*) и другими серверами OVSDB;
- YANG Tools – управляющие утилиты Netconf и Yang, написанные на языке Java.

Состав Virtualization Edition:

- сервис привязки метаданных (англ. *Affinity Metadata Service*), предоставляющий API для определения взаимосвязи типов нагрузки и уровней обслуживания;
- фреймворк Defense4All для выявления и блокирования DDoS-атак;
- система Open DOVE для организации сетевой виртуализации, включающая управляющий слой и уровень обмена данными на базе виртуального коммутатора Open vSwitch;
- приложение для обеспечения работы многопользовательской (англ. *Multi-tenant*) виртуализированной сети с использованием OpenFlow;

Состав Service Provider Edition:

- сервис привязки метаданных (англ. *Affinity Metadata Service*);
- система управления трафиком с использованием моделей BGP-LS (модель топологии с использованием протокола BGP) и PCER (модель программирования путей);

- фреймворк Defense4All для выявления и блокирования DDoS-атак;
- плагин LISP Flow Mapping с реализацией протокола LISP (англ. *Locator/identifier Separation Protocol*) и сервиса маппинга;
- плагин SNMP4SDN, предоставляющий поддержку протокола SNMP и API для управления типовыми коммутаторами Ethernet.

ONOS

Тип лицензии: Apache license 2.0.

Кем разрабатывается: Open Networking Lab (On.Lab) – некоммерческая организация, действующая в сфере разработки ПО для программно-определяемых сетей (SDN).

Где применяется: Huawei, AT&T, NTT Communications.

Особенности: распределённый SDN-контроллер (в виде множества экземпляров), который чаще всего используется в комплексных решениях SDN/NFV. Он обеспечивает плоскость управления для программно-конфигурируемых сетей SDN, разделяет их на гибкие сегменты (Slice) и обеспечивает работу программных модулей для соответствующего администрирования глобальной сети, состоящей из логически независимых сегментов.

Архитектура контроллера ONOS представлена на рис. 5.10.

Контроллеров (экземпляров) ONOS в данной архитектуре может быть много, для разных сегментов сети, для разных сетевых слоёв и для разных приложений.

Состав ядра ONOS:

- модуль **Абстрагирование конфигурации** на уровне распределённой сети отвечает за общий вид сетевой топологии, которая отображается на уровне программной конфигурации сети.

- модуль **управления сетевыми потоками** предоставляет информацию для таблиц OpenFlow (англ. *Flow Table*), которые загружаются в сетевые элементы (маршрутизаторы, коммутаторы, файрволлы и пр.) нижележащей IP-сети для управления потоками данных, проходящих через них.

- уровень **южных протоколов** работает как провайдер протоколов управления программно-конфигурируемой IP-сети. Это может быть как протокол OpenFlow, так и другие протоколы, например, NETCONF.

- **приложениями** в этой архитектуре могут быть: протоколы, драйверы, библиотеки модулей и подпрограмм, модели данных (например, YANG), приложения направления трафика (англ. *traffic steering*) и другие сетевые приложения.

- **фреймворк для намерений** (англ. *Intent Framework*) – подсистема контроллера ONOS, позволяющая приложениям определять их «пожелания» по управлению сетью в форме политик. Ядро (Core) ONOS воспринимает спецификации намерений и переводит их в установки (скрипты) намерений

путём компиляции. Эти установки управляют операциями по организации сетевой среды.



Рис. 5.10. Архитектура ONOS

В фреймворк можно вводить дополнительные шаблоны намерений, и также дополнительные компиляторы и установщики, причём их можно устанавливать без прерывания работы. Это позволяет постоянно улучшать и модифицировать фреймворк намерений под нужды конкретной цифровой системы.

На самом низшем уровне, намерения могут быть описаны следующим образом:

- **сетевые ресурсы**: набор моделей объектов, таких как подключения (линки), которые связывают части сети, которые затрагиваются намерением.
- **ограничения**: определённые коэффициенты, или веса (англ. *Weights*), заданные для сетевых ресурсов, такие как полоса пропускания, длина световой волны в оптоволокне и тип подключения (линка).
- **критерии**: поля заголовка пакета, которые описывают трафик сетевого слоя (срез трафика). Параметр **TrafficSelector** в модели намерений содержит набор объектов, которые выполняют роль интерфейса критерия (англ. *Criterion*).
- **инструкции**: действия, прилагаемые к срезу трафика, такие как модификация поля заголовка пакета, или направление пакета на определённый выходной порт. Параметр **TrafficTreatment** содержит набор инструкций, которые выполняются через интерфейс **Instruction**.

Узким местом платформы можно назвать *безопасность*. У ONOS есть ряд незакрытых уязвимостей: например, подверженность DoS-атакам и возможность установки приложений без аутентификации.

RUNOS

Тип лицензии: некоммерческая версия и коммерческая.

Кем разрабатывается: RunSDN (Россия).

Где применяется: некоммерческая (демо) версия прошла апробацию в России, США, Европе, Индии и Бразилии. Коммерческая (полная) – ПАО Ростелеком, АО "НИИ "Масштаб".

Особенности: является самым быстрым из всех существующих в мире ПКС контроллеров:

- за 1 секунду обрабатывает 30 млн. потоков;
- за 45 мкс. устанавливает новое соединение;
- 150 мс и меньше на гарантированный отклик на запрос коммутатора;
- поддерживает до 1 тыс. коммутаторов.

RuNOS предназначен для использования в сетях операторов связи, сетевой инфраструктуре центров обработки данных, в корпоративных сетях с высокими требованиями к гибкости и масштабированию. Язык разработки – C++.

Реализованные средства мониторинга позволяют отображать в WEB-GUI загрузженность портов и следить за управляющим трафиком.

В графическом интерфейсе визуализирована топология сети и доступная подробная информация об отдельных её элементах.

Отказоустойчивость обеспечивается режимом Active/Standby – в случае «падения» основного экземпляра контроллера управление передаётся на резервный.

Для отладки используется интерфейс командной строки.

Подтверждена совместимость RuNOS с устройствами производителей Arista, Juniper, Extreme Networks, NEC, IBM, Huawei.

Для телекоммуникационных компаний:

– снижаются эксплуатационные затраты за счёт реализации концепции управления сетью как единым объектом.

Для разработчиков сетевых сервисов:

– предоставляется высокоуровневая модель программирования программно-конфигурируемой сети, в которой скрыты низкоуровневые детали протокола OpenFlow и ограничения коммутационного оборудования.

Для распределённых дата-центров:

– снижение стоимости аренды высокоскоростных каналов за счёт использования решения "Пропускная способность по требованию".

Для предприятий и организаций:

– быстрый и недорогой переход к современной организации корпоративной сети на базе технологии ПКС и как следствие, сокращение затрат на поддержку и развитие корпоративной сети.

Архитектура RuNOS представлена на рис. 5.11.

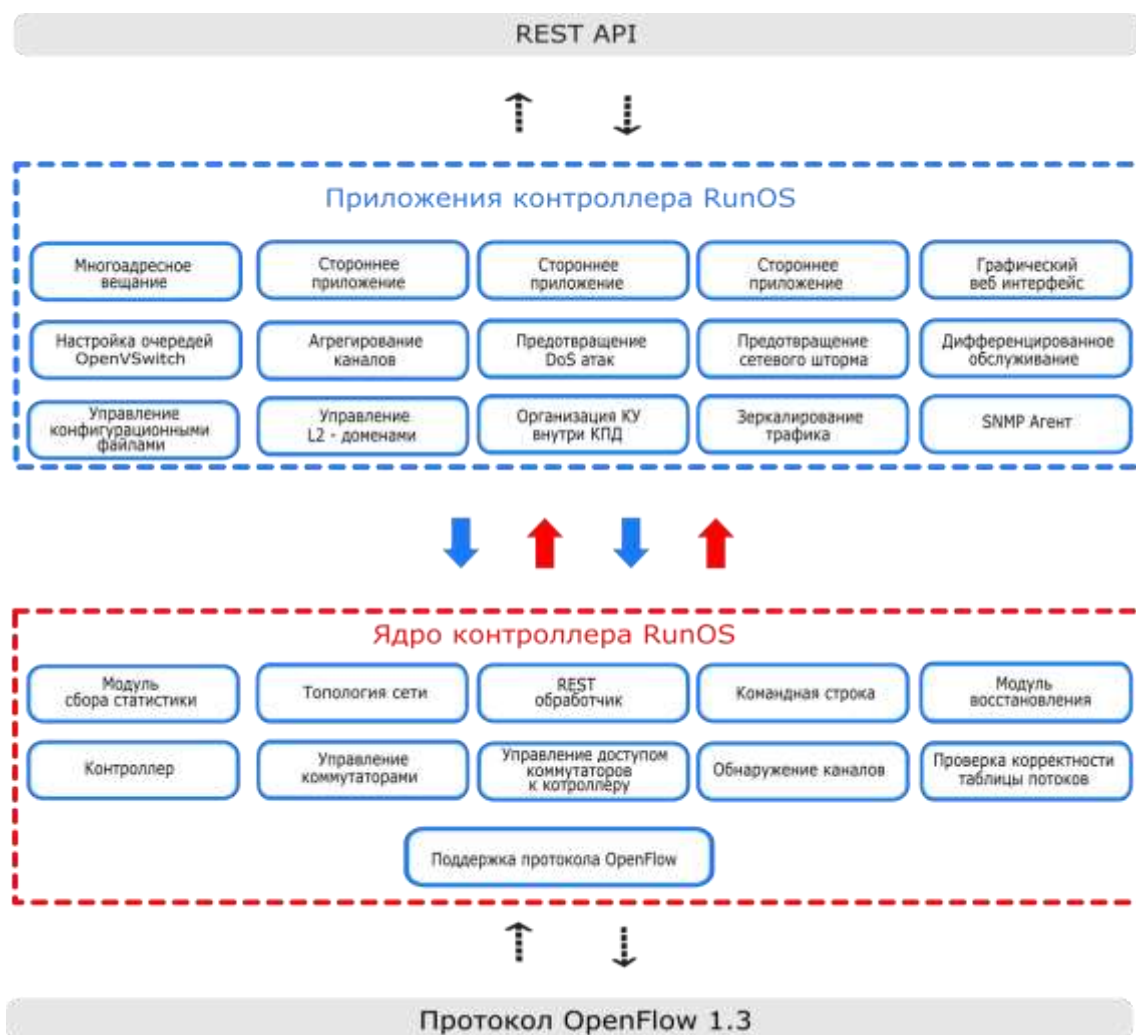


Рис. 5.11. Архитектура RuNOS

Пользовательские приложения реализуют политики управления сетью:

- многоадресное вещание;
- зеркалирование трафика;
- организацию контура управления (КУ) внутри контура передачи данных (КЖД);
- агрегирование каналов;
- создание L2-доменов;
- управление очередями.

В ядре реализованы функциональности, которые используют приложения:

- разведка топологии;
- управление из командной строки;
- обеспечение высокой доступности;

- обеспечение работы по протоколу OpenFlow/NETCONF;
- предоставление информации о коммутационных устройствах;
- сбор статистики с коммутационных устройств;
- обработка и передача сообщений через REST интерфейс.

Южный интерфейс обеспечивают протоколы OpenFlow версии 1.3 и NETCONF.

Bandwidth on Demand или "Пропускная способность по требованию" – принципиально новый подход к организации передачи данных, который позволяет заменить аренду канала с гарантированной полосой пропускания, рассчитанной на максимальную интенсивность трафика, предоставлением L3-канала с пропускной способностью, отвечающей текущим потребностям клиента, без потери качества обслуживания.

За счёт этого обеспечиваются:

- ***масштабируемость***. Сервис автоматически обрабатывает изменение числа клиентов, для которых одновременно поддерживаются L3-каналы в подконтрольной топологии.
- ***адаптивность***. Оперативное изменение используемого основного маршрута при изменении топологии или загрузки физических линий связи.
- ***минимальная задержка на выделение ресурсов***. От начала предварительной настройки сетевых устройств до момента использования L3-канала непосредственно пользователем проходит всего несколько минут.
- ***отказоустойчивость***. Реализованные механизмы детектирования отказа, возможность автоматического переключения и резервирование позволяют, в случае отказа основного маршрута, переключиться на запасной за доли секунды.
- ***балансировка трафика***. Распределение потоков данных по нескольким маршрутам.

NOX

Тип лицензии: General Public License (GPL).

Кем разрабатывается: Nicira Networks.

Где применяется: разработчики и исследователи SDN.

Особенности: первый контроллер SDN, который разрабатывался параллельно протоколом OpenFlow, реализован на языке программирования C++ с открытым кодом и поддержкой многопоточности, представляет собой основу для изучения SDN и для более продвинутых проектов.

POX

Тип лицензии: General Public License (GPL).

Кем разрабатывается: Стэнфордский университет.

Где применяется: сфера образования, исследования.

Особенности: «младший брат» контроллера NOX, который реализован на языке Python, применяется для исследований уязвимостей и других проблем SDN, а также для обучения студентов и сотрудников компаний основам SDN.

Ryu

Тип лицензии: Apache License 2.0.

Кем разрабатывается: Nippon Telegraph and Telephone (NTT).

Где применяется: малые сети и исследовательские проекты.

Особенности: Ryu предоставляет программные компоненты с чётко определенными API, которые облегчают разработчикам создание новых приложений для управления и контроля сети. Ryu поддерживает различные протоколы для управления сетевыми устройствами, такие как OpenFlow, Netconf, OF-config и т. д.

FloodLight

Тип лицензии: Apache License 2.0.

Кем разрабатывается: Big Switch Networks.

Где применяется: SDN сети любого масштаба.

Особенности: контроллер разработан на основе Beacon, поддерживает широкий спектр виртуальных и физических коммутаторов, способен поддерживать смешанные OpenFlow сети и сети традиционной архитектуры.

На рис 5.12 можно видеть архитектуру контроллера FloodLight.

4.5 Сравнение контроллеров SDN

Сравнительная характеристика наиболее распространённых контроллеров представлена в таблице 5.1.

Таблица 5.1. Сравнительная характеристика контроллеров SDN

Контроллер	Лицензия	Язык прог.	Централизация	Производительность	Южный интерфейс + OpenFlow	Северный интерфейс
NOX	GPLv3	C++	центр.	средняя	–	Ad-hoc API
POX	GPLv3	Python	центр.	низкая	–	Ad-hoc API
OpenDaylight	EPL	Java	распр.	высокая	NETCONF/YANG, OVSD, PCEP, BGP/LS, LISP, SNMP	REST, RESTCONF
FloodLight	Apache	Java	центр., м/п	средняя	NETCONF	REST API Flow Pusher API
Ryu	Apache	Python	центр., м/п	низкая	NETCONF, OFCONFIG	Ad-hoc API

Контроллер	Лицензия	Язык прог.	Централизация	Производительность	Южный интерфейс + OpenFlow	Северный интерфейс
Beacon	GPLv2 + FOSS	Java	центр., м/п	высокая	–	Ad-hoc API
ONOS	Apache	Java	распр.	высокая	NETCONF	RESTful
Onix	коммер.	Python C++	распр.	средняя	OVSDDB	API
FlowER	Open	Erlang	центр.	низкая	–	API
Mul	Open	C++	центр.	средняя	NETCONF OVSDDB	RESTful
Maestro	GNU LGPL	Java	центр., м/п	высокая	–	API
RuNOS	коммер. + беспл.	C++	распр.	высокая	NETCONF	REST

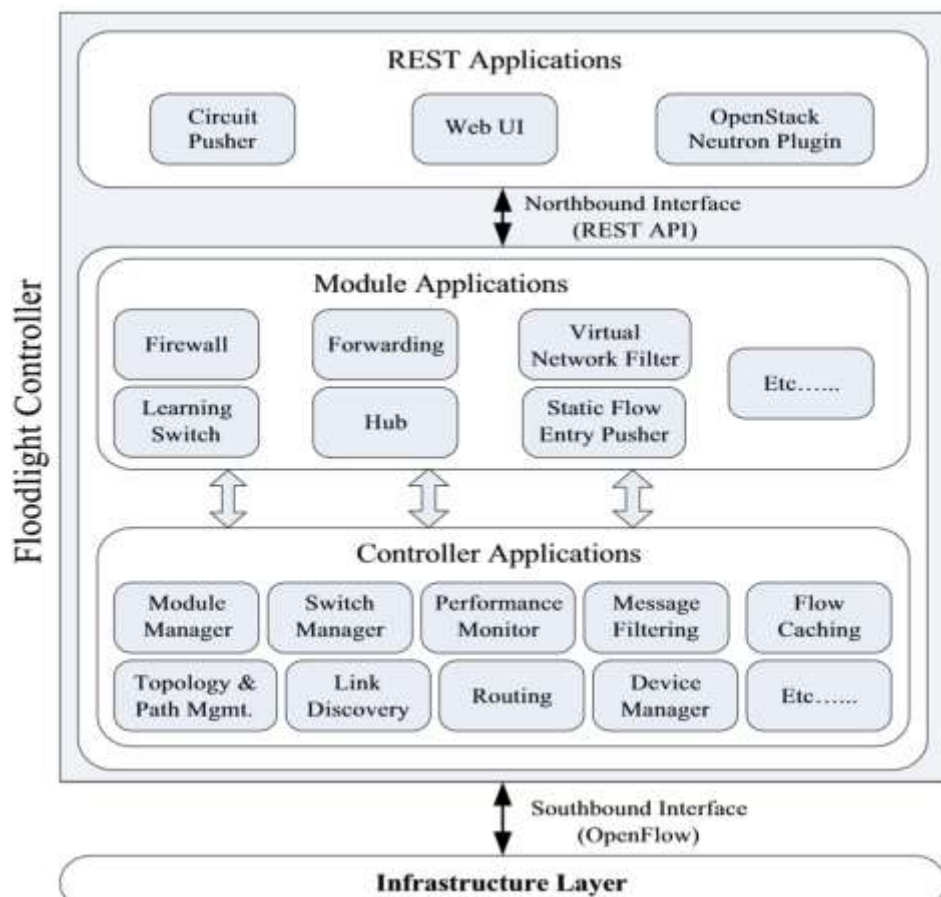


Рис. 5.12. Архитектура FloodLight

Пример распределённой платформы управления

Рис. 5.13 иллюстрирует работу распределённого контроллера в режиме Master/Slave.

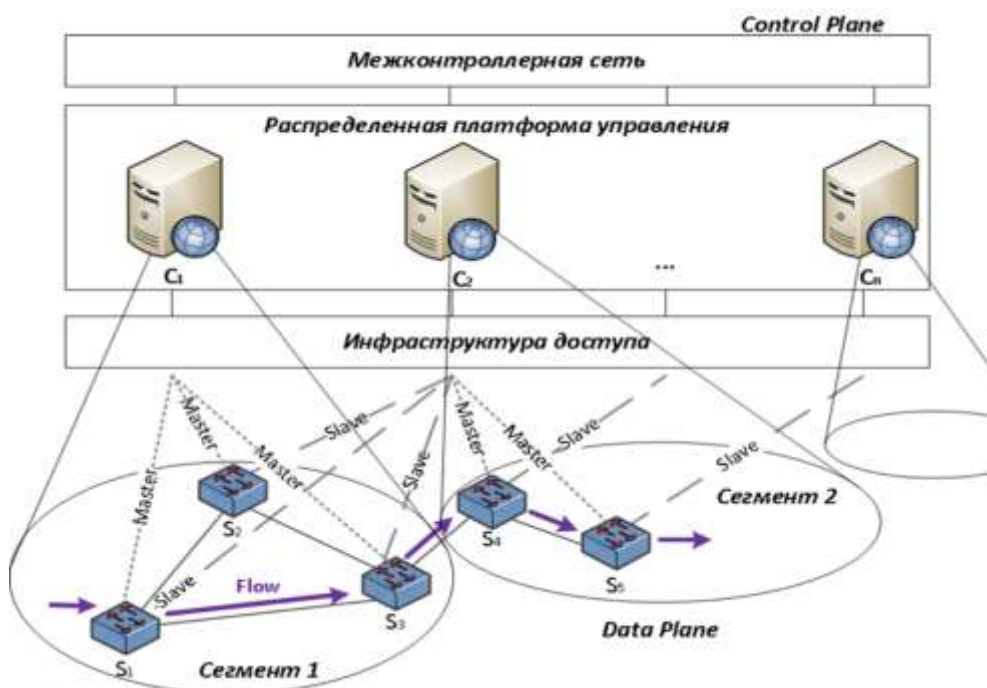


Рис. 5.13. Работа распределённого контроллера

V ПРИМЕНЕНИЕ SDN В СЕТЯХ 5G

5G – сети с программируемыми параметрами, виртуализацией сетевых функций, разделением функциональной архитектуры и физической базовой инфраструктуры с взаимодействием через API (Application Programming Interface).

5.1 Концепция IMT-2020

Требования 3GPP к 5G:

- пропускная способность 10 (до 20) Гбит/с;
- задержка 1 мс;
- ёмкость 1 млн. устройств на 1 кв. км.

Сети поколений 1G – 4G базируются на аппаратных решениях, а 5G – на программных, в том числе – на технологиях SDN и NFV (англ. *Network Functions Virtualization*).

Рис. 5.14 содержит сравнительную характеристику параметров стандартов 4G и 5G.



Рис. 5.14. Характеристики стандартов IMT-Advanced и IMT-2020

По сравнению с 4G сети 5G имеют следующие преимущества:

- усовершенствование eMMB (мобильного широкополосного доступа);
- гарантированная скорость и надёжность;
- высокоскоростные межмашинные коммуникации.

Переход от аппаратных технологий к программным

На рис. 5.15 представлены преимущества программных технологий перед аппаратными.

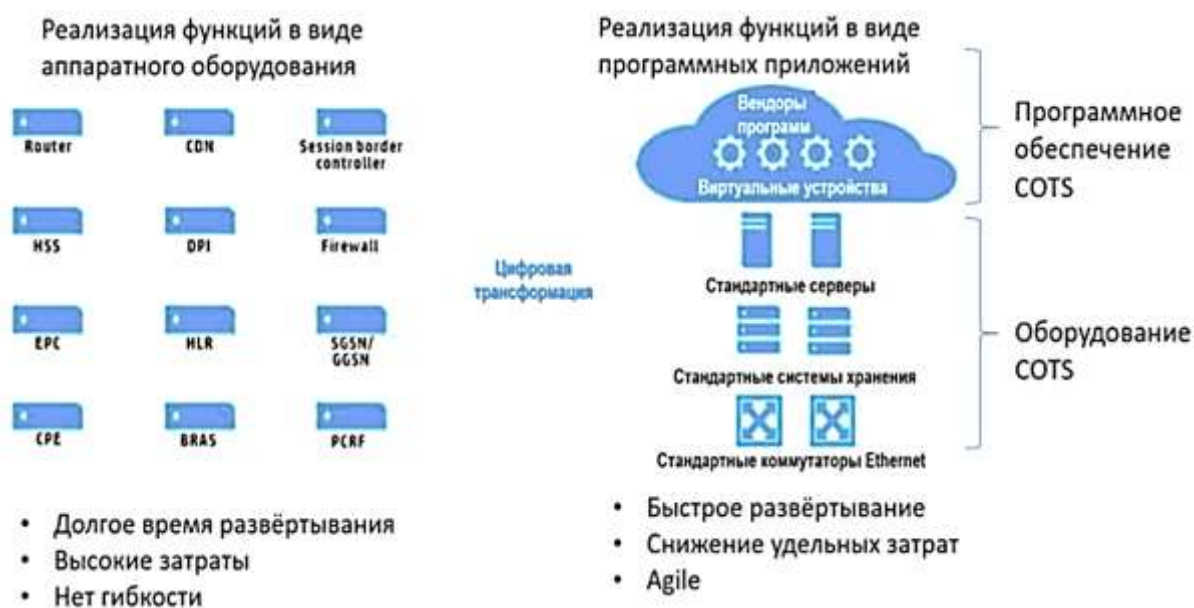


Рис. 5.15. Выгода от перехода к программным технологиям

5.2 Общая архитектура 5G

Рис. 5.16. содержит обобщённую архитектуру 5G сети.

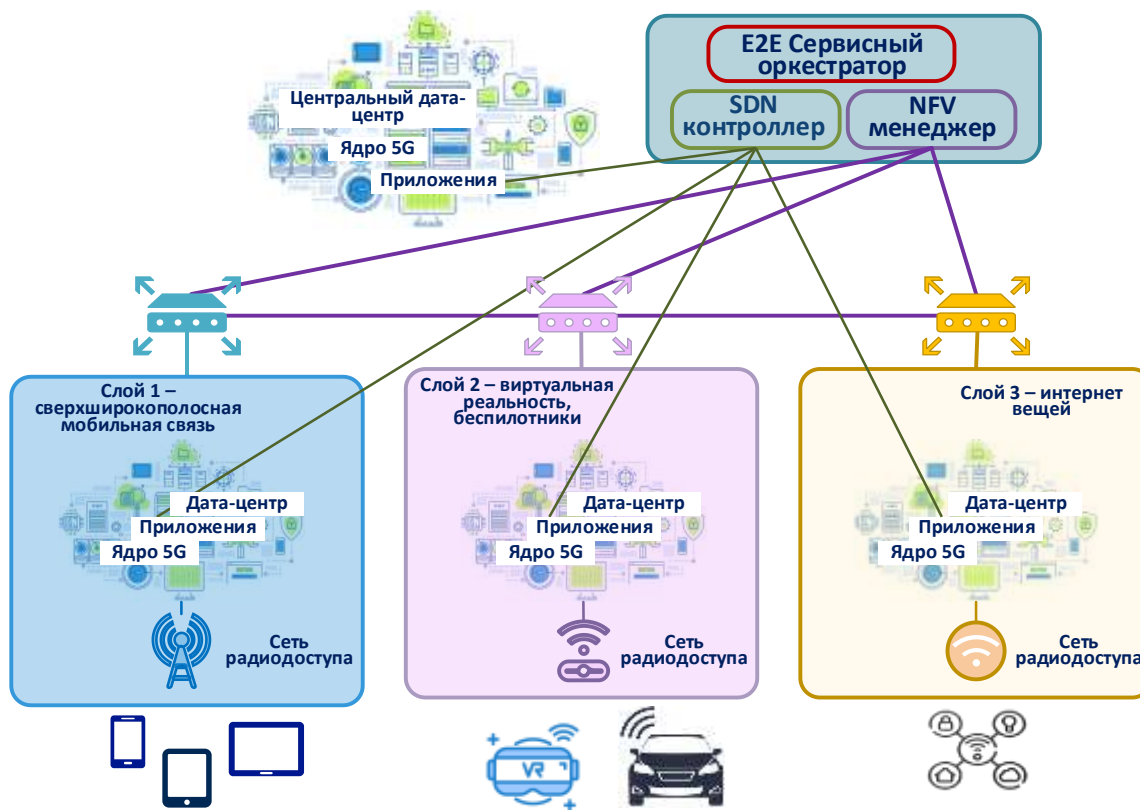


Рис. 5.16. Архитектура 5G

Применение SDN в архитектуре 5G позволяет существенно упростить реализацию транспортного уровня сети, использовать унифицированный не зависящий от поставщика интерфейс между плоскостями управления и передачи данных, а также эффективно адаптировать сеть под передачу больших объёмов гетерогенного трафика.

NFV является дополнением к SDN и позволяет перенести сетевые функции в виртуальное пространство на серверах дата-центров DC (англ. *Data Center*), реализованных на базе стандартного коммерческого оборудования COTS (англ. *Commercial Off The Shelf*). Оборудование COTS представляет собой комбинацию трёх видов стандартных устройств: серверов; коммутаторов и систем хранения данных.

Таким образом, традиционное оборудование сетей мобильной связи в системах 5G заменяется программными продуктами, которые запускаются на серверах и виртуальных машинах VM (англ. *Virtual Machines*) в дата-центрах.

Кроме того, понятие сетевой архитектуры 5G сетей тесно связано с концепцией “сетевой нарезки” (англ. *Network Slicing*). Данная концепция предполагает разделение инфраструктуры сети на логические слои (англ. *slice*), предназначенные для различных сервисов и различных технологий радиодоступа RAT (англ. *Radio Access Technology*). Слои могут быть по отдельности оптимизированы под различные требования к скорости передачи

данных и задержке для конкретных приложений. Поскольку управление такой сложной структурой в единой сети осуществить затруднительно, каждый слой подчиняется отдельной управляющей системе BSS/OSS (англ. Business Support System/ Operation Support System). Технология Network Slicing позволяет обеспечивать гибкость инфраструктуры, безопасность и масштабируемость сетей 5G.

Требования стандарта IMT-2020

- осуществление на одной платформе различных сервисов и приложений;
- сетевая сегментация (слайсинг);
- быстрое внедрение новых сервисов и приложений, их диспетчеризация в сетевых сегментах;
- динамическое распределение сетевого трафика;
- программируемость услуг, мониторинг физических и виртуальных ресурсов;
- интеллектуализация управления сетевыми ресурсами;
- гарантированное QoS;
- полностью централизованное и автоматизированное управление сетью.

5.3 Программируемость IMT-2020

Программируемость подразумевает:

- использование программных средств вместо аппаратных на всех уровнях сетевой модели;
- программная автоматизация конфигурирования и управления сетью.

SDN привносит в 5G методы динамической конфигурации инфраструктурных ресурсов сети для их изменения и лёгкой адаптации к меняющимся условиям и требованиям. Это позволяет обеспечить гибкость, простоту управления и масштабируемость сети.

Недостатки:

- не гарантируется отсутствие багов в программном обеспечении;
- уязвимости в API и ПО ослабляют безопасность SDN и, следовательно, 5G сети;
- закрытое ПО производителя может менять функциональность SDN и 5G.

5.4 Виртуализация IMT-2020

Виртуализация нужна для более эффективного использования (распределения) сетевых ресурсов.

SDN позволяет абстрагироваться от аппаратной платформы и, следовательно, более гибко (легко) работать с физическими ресурсами сети. Например, если физический сервер вышел из строя, его нужно заменить на

такой же физический сервер (купить), пройдя заново все этапы его настройки и конфигурирования. Виртуальный сервер проще заменить его копией на той же или другой аппаратной платформе. Кроме того, можно создать множество копий виртуального сервера и распределить нагрузку между ними.

Виртуализация позволяет:

- эффективно использовать сетевые ресурсы и осуществлять балансировку нагрузки между ними;
- изолировать потоки данных пользователей приложений;
- создавать виртуальные сети поверх физических и использовать для них различные сетевые политики.

SDN позволяет изолировать виртуальные сети друг от друга и, таким образом, обеспечивать их безопасность и производительность. Виртуализация SDN также позволяет осуществлять сегментацию (слайсинг) сети и управление каждым сегментом по отдельности.

Технологии виртуализации SDN реализуются в 5G совместно с технологиями NFV.

Недостатки:

- увеличение количества одновременно работающих виртуальных сетей потребует увеличения вычислительных мощностей физических ресурсов;
- зависимость виртуальных ресурсов от отказоустойчивости физических ресурсов;
- сложность эмуляции всех сетевых устройств.

5.5 Слайсинг IMT-2020

Сегментация сети (по определению 3GPP) позволяет оператору построить множество виртуальных сетей с разными целями и вариантами применения на основе единой физической инфраструктуры. Каждая из этих виртуальных сетей применяет свои принципы адресации, методы маршрутизации, обеспечение QoS, администрирование и пр.

Сетевой сегмент – это логическая сеть автоматически выделенных абоненту по его запросу ресурсов (физических и виртуальных).

Сетевой срез (по определению ITU-T) – логически изолированный сетевой сегмент = виртуальный ресурс + программируемое управление + плоскость данных.

На базе SDN и NFV сетевой слайсинг 5G позволяет обеспечить многопрофильный многопользовательский сервисный набор мобильных сетей с высокой пропускной способностью.

Для каждого сервиса организуется индивидуальная копия мобильной сети (на базе одного оператора связи) со своим набором требований и критериев QoS.

Оркестрация – это процесс отбора необходимых сетевых ресурсов по настраиваемым критериям оптимизации и текущим потребностям абонентов.

(Оркестрация – объединение разрозненных объектов в единое целое).

Недостатки:

- сложность слайсинга;
- сложность оркестрации;
- недостатки, присущие виртуализации.

5.6 Интеллектуализация управления ИМТ-2020

Интеллектуальность сети – способность автоматически вырабатывать и принимать решения по результатам анализа большого количества данных о её состоянии и состоянии её компонент.

Контроллер SDN имеет глобальное сетевое представление, позволяющее автоматически собирать трафик и проводить его интеллектуальный анализ и применять его результаты в 5G.

В то же время контроллер SDN (и южный интерфейс) считается узким местом сети, требует применения механизмов отказоустойчивости и надёжности.

Применение искусственного интеллекта усовершенствует управление сетью.

Недостатки: сложность интеллектуализации.

5.7 Диспетчеризация ИМТ-2020

Диспетчеризация в SDN осуществляется следующим образом:

- 1) контроллер получает потоки из плоскости данных;
- 2) контроллер организует маршрутизацию потоков;
- 3) контроллер проверяет поток и заполняет таблицы потоков коммутаторов на маршруте потока.

Коммутаторы продвигают пакеты в соответствии с таблицами потоков.

Контроллер взаимодействует с плоскостью передачи через протокол OpenFlow и осуществляет полное управление коммутаторами (конфигурирование, маршрутизация, безопасность, перераспределение трафика и пр.).

Недостатки: полная зависимость от контроллера (при разрыве соединения коммутаторы становятся неуправляемыми).

5.8 Динамическое распределение трафика ИМТ-2020

Динамическое распределение трафика связано с диспетчеризацией и интеллектуализацией, т.к. контроллер SDN полностью управляет пересылкой. Он может забрать пакеты у одного коммутатора и передать другому.

По результатам интеллектуального анализа трафика происходит перераспределение нагрузки таким образом, чтобы обеспечить наиболее эффективное её обслуживание.

Управление трафиком происходит на уровне приложений с учётом требований безопасности, надёжности и с помощью API.

Недостатки:

- централизованное управление;
- сложность алгоритмов перераспределения нагрузки;
- сложность интеллектуального анализа (невозможность проверить достоверность его результатов).

5.9 Централизованное управление сетью IMT-2020

В SDN централизованное управление плоскостью передачи (коммутаторы SDN) осуществляется контроллером SDN, который в свою очередь находится под управлением приложений (плоскость управления).

На контроллере установлена операционная система, которая собирает сведения о топологии сети и на этой основе организует работу сетевых служб.

Недостатки: все присущие централизации управления.

5.10 Качество обслуживания (QoS) IMT-2020

В SDN контроллер с операционной системой управляет коммутаторами, собирает статистику их работы и детализации трафика, осуществляет *интеллектуальный анализ* собранной статистики.

Статистика и анализ служат основой прогнозирования показателей *качества обслуживания*.

Показатели QoS:

- задержка;
- джиттер задержки;
- коэффициент загрузки;
- коэффициент простоя;
- BER;
- вероятность потерь и пр.

В 5G контроллер SDN на основе статистической обработки собранных данных, интеллектуального анализа и методов машинного обучения динамически управляет QoS.

Возможности SDN и NFV позволяют обнаруживать дефекты, реагировать на них адекватно и исправлять их наилучшим образом.

Достоинства: самоорганизация, искусственный интеллект, гарантированное QoS.

Недостатки:

- сложность интеллектуального анализа;
- точность интеллектуального анализа;
- сомнительная достоверность принятия решений.

VI NETWORK FUNCTIONS VIRTUALIZATION, NFV

Основная идея NFV заключается в том, чтобы отделить сетевые функции от аппаратного обеспечения, превратив их в более гибкие и динамичные. Так, если в традиционной модели для каждой сетевой функции необходима дополнительная единица оборудования, «железный» компонент, то в модели NFV достаточно одного физического фундамента, а расширение сетевых функций осуществляется в облаке. Архитектура NFV представлена на рис. 5.17.

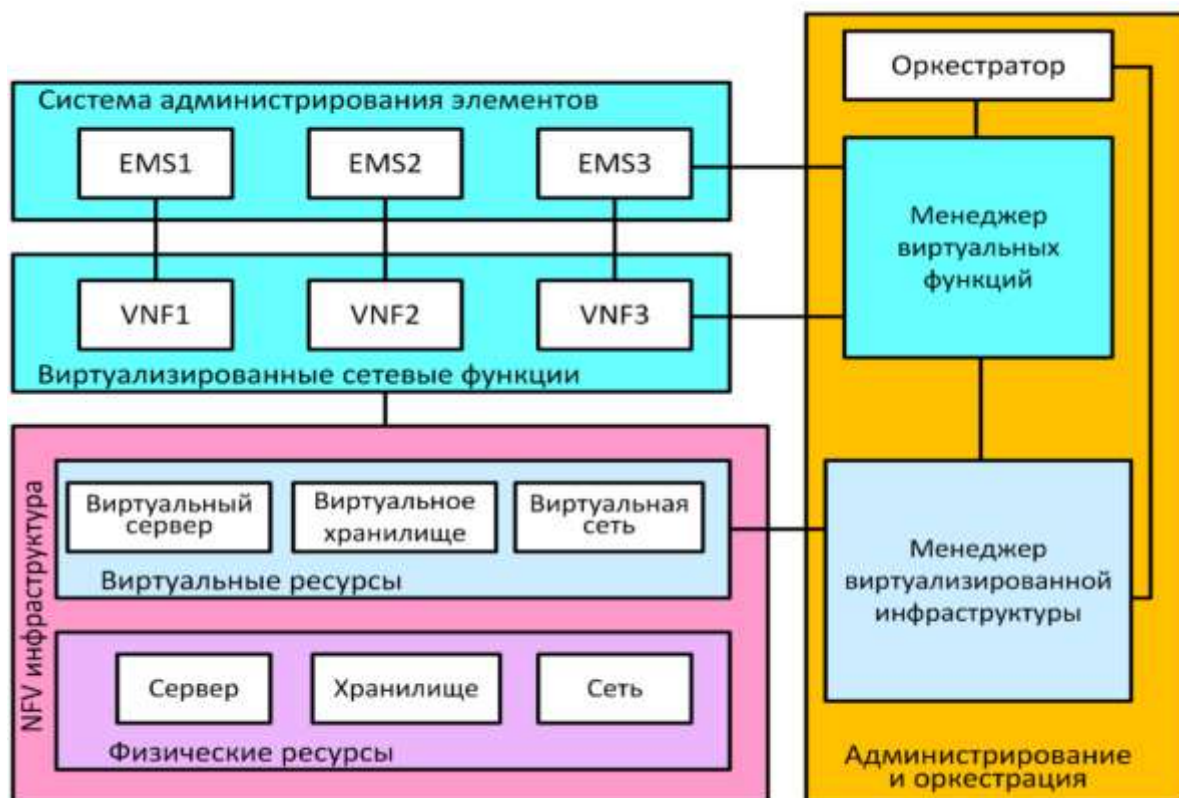


Рис. 5.17. Архитектура NFV

Система администрирования элементов EMS (англ. *Element Management System*) управляет работой виртуализированных сетевых функций VNF. Причём сама EMS также может представлять собой VNF, управляемую другой EMS.

Инфраструктура виртуализации сетевых функций NFVI (англ. *NFV Infrastructure*) – это среда, на которую опираются в своей работе VNF. Она включает ресурсы физического оборудования и виртуальные ресурсы. Специальная программная платформа гипервизор (англ. *Hypervisor*) производит отделение программных средств от аппаратных, то есть даёт возможность запускать программы независимо от выбранного оборудования, таким образом, преобразуя физические компоненты сети в виртуальные, необходимые для реализации VNF.

Система менеджмента и оркестрации MANO (англ. *Management & Orchestration*) представляет собой важнейшую часть технологии NFV. MANO

состоит из трёх элементов:

- **менеджер VNFM** (англ. *VNF Manager*) управляет жизненным циклом одной или нескольких функций VNF (запуск, остановка, обслуживание и пр.);
- **менеджер виртуализированной инфраструктуры VIM** (англ. *Virtualized Infrastructure Manager*) осуществляет администрирование ресурсов NFVI внутри одного домена инфраструктуры оператора. Он также отвечает за сбор результатов измерений производительности и событий.
- **оркестратор** (англ. *Orchestrator*) через менеджеров VNFM и VIM осуществляет администрирование и обслуживание как VNF, так и NFVI.

VII ПРЕИМУЩЕСТВА И НЕДОСТАТКИ SDN

Эффекты от внедрения SDN

- автоматизированное управление сетью. Упрощение эксплуатации;
- автоматизированное внедрение нового сетевого сервиса и быстрота развёртывания;
- программируемость. Возможность внедрения новой идеи в сети (сетевого сервиса);
- независимость от производителя. Использование открытых стандартов;
- гибкая модернизация и масштабирование;
- гибкость сети. Эффективное использование аппаратных и канальных ресурсов.

Преимущества SDN

Благодаря снятию с коммутаторов нагрузки по обработке тракта управления, SDN позволяет этим устройствам направить все свои ресурсы на ускорение перемещения трафика, что существенно повышает производительность. При этом за счёт виртуализации управления сетью снижаются расходы на их построение и сопровождение. По результатам тестов, проведённых на базе крупнейших провайдеров США, использование SDN позволяет на 20–30% увеличить утилизацию ресурсов ЦОД и в несколько раз снизить эксплуатационные расходы.

Программные средства SDN позволяют администраторам добавлять новую функциональность к уже имеющейся сетевой архитектуре. При этом новые функции будут работать на многих платформах — их не придётся реализовывать заново во встроенном программном обеспечении коммутаторов каждого поставщика.

На централизованном контроллере SDN системный администратор может наблюдать всю сеть в едином представлении, за счёт чего повышаются удобство управления, безопасность и упрощается выполнение ряда других задач. Действительно, поскольку администратор видит все потоки трафика, то

ему легче замечать вторжения, назначать приоритеты различным типам трафика и разрабатывать правила реагирования сети при заторах и проблемах с оборудованием.

Теоретически неограниченные возможности сетей SDN к расширению позволяют строить реальные облака, масштабируемые в зависимости от решаемых задач. При этом сеть обладает требуемой «интеллектуальностью», необходимой, в частности, для оркестровки работы обширных групп коммутаторов.

Недостатки SDN

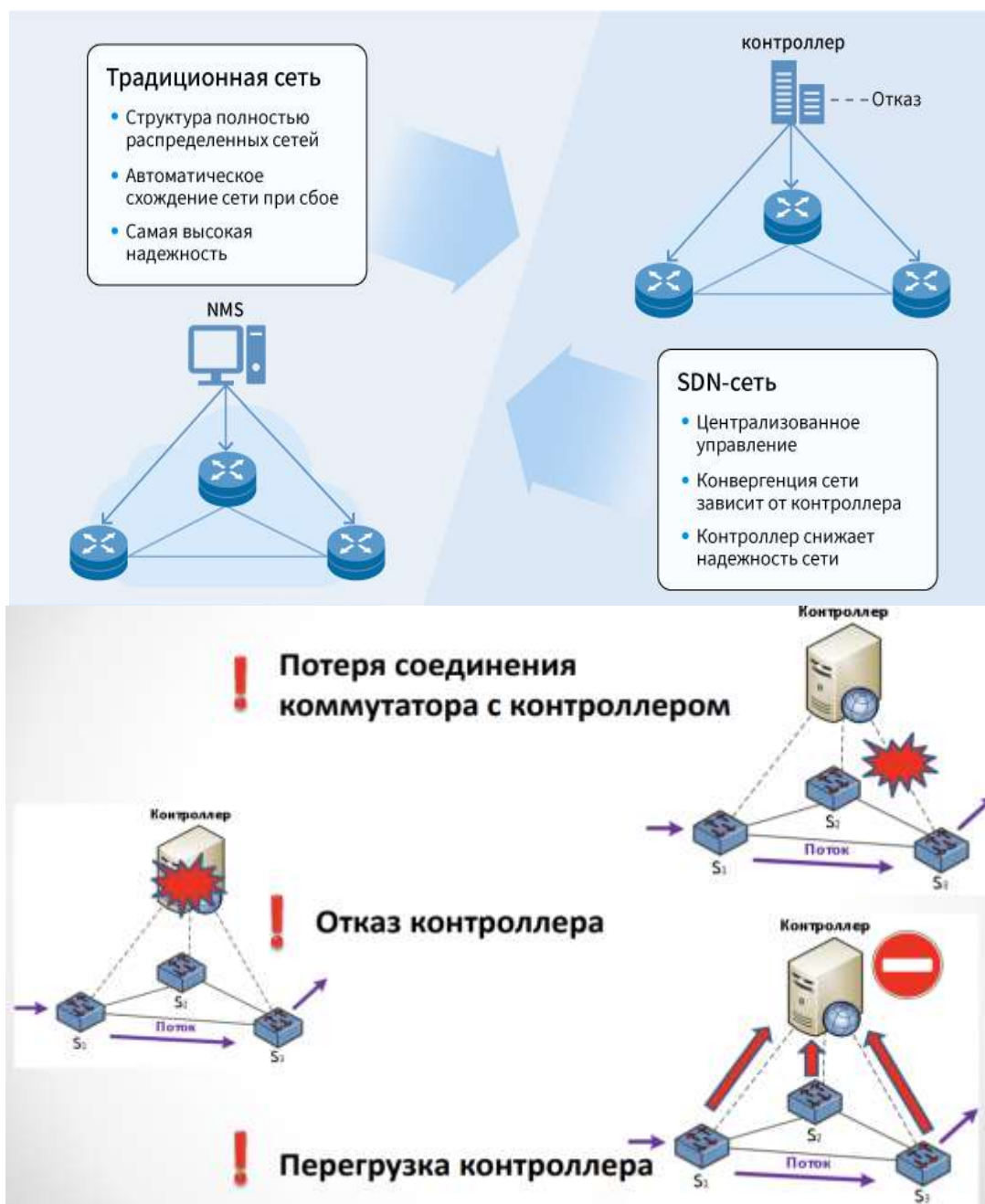


Рис. 5.18. Слабые места сетей SDN