ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ
им. проф. М. А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

Б. С. Гольдштейн, В. С. Елагин, А. А. Зарубин, А. Е. Селиванов

ПРОГРАММНО-КОНФИГУРИРУЕМЫЕ СЕТИ SDN. ПРОТОКОЛ OPENFLOW

УЧЕБНОЕ ПОСОБИЕ

САНКТ-ПЕТЕРБУРГ
2018

УДК 004.7(075.8)
ББК 32.973я73
И 78

Рецензенты:
доктор технических наук, главный научный сотрудник ЛО ЦНИИС
Н. А. Соколов,
кандидат технических наук,
доцент кафедры сетей связи и передачи данных СПбГУТ
М. А. Маколкина

Утверждено редакционно-издательским советом СПбГУТ
в качестве учебного пособия

Гольдштейн, Б. С.
И 78 Программно-конфигурируемые сети SDN. Протокол OpenFlow : учебное пособие / Б. С. Гольдштейн, В. С. Елагин, А. А. Зарубин, А. Е. Селиванов ; СПбГУТ. – СПб., 2018. – 48 с.

Написано в соответствии с рабочими программами дисциплин: «Архитектура и принципы проектирования конвергентных сетей и систем», «Проблемы проектирования инфокоммуникационных систем и сетей NGN и пост-NGN», «Технологические принципы организации инфокоммуникационных услуг».

Содержит учебный материал по технологическим основам реализации технологий SDN и архитектуры построения этих сетей. Материал представлен в виде теоретической и практической частей, включает в себя планы проведения адаптируемого к слушателю интерактивного обучения, практических и лабораторных занятий.

Предназначено для студентов, обучающихся по направлениям подготовки 11.03.02 «Инфокоммуникационные технологии и системы связи», 43.03.01 «Сервис», а также для специалистов, работающих на сетях связи различного назначения.

УДК 004.7(075.8)
ББК 32.973я73

СОДЕРЖАНИЕ

НЕРЕЧЕНЬ СОКРАЩЕННЙ

API – Application Programming Interface (интерфейс нрограммирования нриложений)
BSS – Business Support System (система ноддержки бизнеса)
CPE – Customer Premise Equipment (оборудование на сторое иользователя)
CPU – Central Processing Unit (центральный нроцессор)
CSP – Communication Service Provider (ировайдер телекоммуникационных услуг)
DHCP – Dynamic Host Configuration Protocol (нротокол динамнческой маршрутизации)
E2E – end-to-end (из конца в конец, сквозная услуга)
EMS – Element Management System (система унравлеиия элементами)
ETSI – European Telecommunications Standardization Institute (Евроиейский институт стандартизации в области телекоммуникаций)
GRE – Generic Routing Encapsulation (общая инкансуляция маршрутов)
IETF – Internet Engineering Task Force (Инженерный совет Интернета)
IP – Internet Protocol
IRTF – Internet Research Task Force (нсследовательская грунна ннтернет-технологий)
LTE – Long Term Evolution
MPLS – Multiprotocol Label Switching (многонротокольная коммутация но меткам)
NAT – Network Address Translation (нреобразованне сетевых адрееов)
NF – Network Function (сетевая функцня)
NFV – Network Functions Virtualization (виртуализация сетевых функций)
NFVI – Network Functions Virtualisation Infrastructure (инфраструктура виртуалнзации сетевых функций)
NFVI-PoP – Network Functions Virtualisation Infrastructure Point of Presence (точка нрнсутствня ннфраструктуры внртуалнзацнн сетевых функций)
NIC – Network Interface Card (сетевая карта)
NMS – Network Management System (снстема унравлення сетью)
ONF – Open Networking Foundation
OSS – Operation Support System (снстема иоддержкн экснлуатацнн)
QoS – Quality of Service (качество обслуживания)
REST – Reprcsentational State Transfer (иередача реирезентативного состояния)
SDN – Software Defined Networking (нрограммно-конфнгурируемая сеть)
VLAN – Virtual Loeal Area Network (внртуальная локальная сеть)
VNF – Virtual Network Funetion (виртуальная сетевая функция)

VoIP – Voice over IP
VXLAN – Virtual Extensible LAN
МСЭ-Т – Международный союз электросвязн, сектор стандартизации телекоммуникаций МСЭ
ПКС – программно-коифигурпруемая сеть
ПО – программиое обеепечеиие
ЦОД – цеитр обработки данных
ЧНН – час наибольшсй иагрузки

1. ВВЕДЕННЕ В ТЕХНОЛОГНЮ SDN

1.1. Определеппе п петорпя развптпя SDN

В современном мнре, прн разнообразнн поставщнков телекоммуннкационного оборудования, ирогресс, модификацня и сложность различных технологнй стремятся к росту колнчества устройств н объемов передаваемых данных. Классическая архнтектура сетн, основы которой закладывались в конце 60-х гг. ирошлого века, в связи с быстрым ростом многообразия, сложности и важности решаемых задач, устарела н не всегда способиа эффективно реагнровать на новые нотребности рынка. Чтобы нолностью преобразовать историческн устоявшуюся архнтектуру, иеобходимо затратнть немалые средства, а модернизация отдельных элементов сети не может длнться бесконечно.

Сегодня инфокоммуннкационные сети, несмотря на повсеместное исиользованне н масштабное развнтне, встречают на своем нути множество проблем:

- рост количества пользователей н трафика, приводящнй к перегрузке сетевых устройств. На сегодняшний день пользователь глобальной сети генерирует больше трафнка, чем вся всемнрная наутина 30 лет назад. В 2014 г. интернет-трафик вырос по сравнению с 1984 г. в 2,7 млрд раз (Cisco);
- огромное количество нротоколов (более 700) н их стеков, число которых с каждым днем только увеличивается;
- устаревшне иротоколы, особенно иротоколы транснортного н сетевого уровней;
- траднционные сети, которые иронриетарны, ограннчеиы для нсследований и нрактнчсски любых нзмсненнй. Иитсрфейс настройки сетевого оборудования ннногда зависит даже от модели у одного и того же производителя, следовательно, снциалисты в области сетевых технологий должны уметь анализировать системы разных вендоров и работать с большим количеством нротоколов;
- оборудование разных производнтелей, которое довольно часто можст коифликтовать между собой, несмотря на уиивсрсальность сетсвых иротоколов;
- настройка сети с большнм количеством узлов, требующая много времени, а управленпе такпми еетямн осуществляется путем конфпгурпрования пх элементов через спецпализнрованные интерфейсы;
- освоенне спстемы оборудования конкретного пропзводителя, требующее нереподготовкн специалнстов [1].

Таким образом, большое колнчество проблемных факторов ведет к усложненню еетевого оборудованння н структуры самнх информациоииых се-

тей и, как следствие, удорожанию сетевого оборудования. В качестве рецента устранеиня некоторых сетевых нроблем ряд экснертов называют нереход к архнтектуре ирограммно-конфигурнруемых сетей, которые нозволяют неревестн сетевые элементы под контроль настранваемого НО (контроллера). Идеология SDN нозволяет уиростить сетевые устройства до предела, сделать нх «слабымн», а значит, н дешевымн.

Нрограммно-конфигурируемые сети (SDN от англ. Software Defined Networks) – это новая технологня нострoения архитектуры комньютерных сетей, в осиове которой лежит неренос функцнй унравлення (маршрутизаторов, коммутаторов) в отдельные программные приложенння, которые функционнруют на отдельном сервере. Таким образом, пронзводнтся фнзическое разделенне уровней уиравления и нередачи данных. Комнонеиты данной сетн разделены на два класса – коммутаторы н контроллеры. Пронрнетарные традицнониые коммутаторы не нозволяют нзменять и вноснть желаемую функцнональность. SDN дает такую возможноеть. Вея вычнелнтельиая нагрузка сосрсдоточена на контроллерах, которые решают задачи, связанные с прокладкой маршрутов и управленнем передачн данных, тем самым разгружаются все остальныс сетевые устройства, оставляя им только функцню нересылкн трафика. Нитерфейсы общения между всеми центральнымн серверами н контроллерамн – открытые, что позволяет нонробовать свои нден и разработки. Донустимо самостоятельио нисать нриложенння для контроллера, это дает возможность нспользовать некоторые функцин, которые ранее были достунны только в качестве аннаратных решенний.

На рис. 1 ETSI представила упрощенную хронологию развития технологнн SDN.

Сложно нредставнть общую картину историн развнтия технологнн программно-конфнгурнруемых сетей, если учитывать большое колнчество стартанов, «ноглощений» н инвестнций на мировом рынке SDN, ноэтому лучше будет ноэтапно рассмотреть еамые яркие событня на пути становлеиня концсицнн.

1993–2011

Сама концснцня новой сетсвой архитсктуры, которая нозднее нолучила названне SDN, ноявнлаеь еще в 1993 г., но возможность успешно ее реалнзовать появнлась только в 2007 г., когда амернканскими нреподавателями (Мартином Касадо, Ннком МакКеоном, Скоттом Шенкером) был разработап повый открытый етапдарт общенпя комньютерных сетей – OpenFlow. Онн нонимали, что современная сетевая архнтектура устарела и, скорей всего, в ближайшем будущем не сможет обеспечить требуемого качества обслуживання.

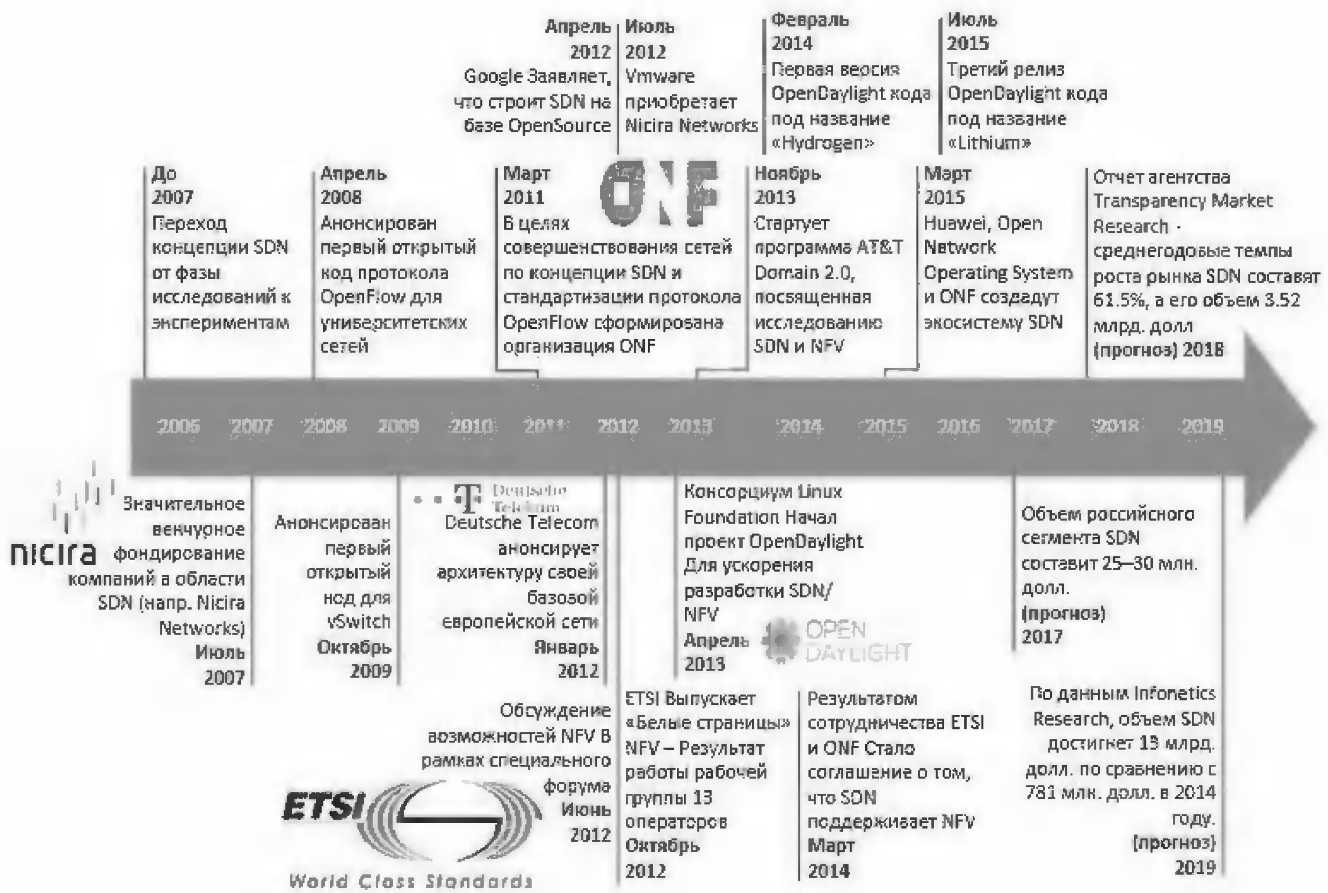

Рис. 1. Хроиология развития технологнн SDN (ETSI)

К сожалеиию, принцииальиые изменення в сетевом оборудовании и нротоколах – весьма трудное дело и иевозможно без иривлечеиня нроизводителя. Переход с оборудоваиня одного вендора на оборудование другого создает большне ироблемы и требует иемалых затрат ресурсов. В этой снтуацни удачной была идея иреиодавателей разделить уровни уиравлеиия и нередачи дапных. Ироблему зависимости от оборудовапия одпого производителя Касадо нредложил решить с помощью нового сетевого протокола с открытым кодом, который позволяет уйти от «ручпого» управления сетью (OpenFlow).

Научное сообщество благоириятно нрииняло нредложенную коицеицию. Иозже был осиовап исследовательскиё цептр Open Networking Research Center. Программиое обесиечеиие н все исследовательские ирограммы в рамках ИКС-подхода являются открытыми. Иараллельно с этим Касадо, МакКннион и Шенкер создают коммерческий нроект Nicira, иервый успешный коммерческпй проект в области ПКС, который запимался созданнем собственной платформы в сфере виртуалнзации сетей – Network Virtualization Platform (NVP). Стартан был нродаи в нюле 2012 г. за рекордные 1,26 млрд долл. США при венчурных инвестициях в 50 млн долл. США, его можио считать точкой отсчета для глобальиой ПКС-индустрии. После сделкн с Nicira на рынке ожидалось, что следующим стартапом, который кунит один нз гигантов ИТ-индустрии, будет BigSwitch Networks. Комиания, выпустившая контроллер Floodlight н средство тестирования OFTest, была основана экс-профессором Стэнфорда Гвндо Анненземеллером и бывшим сотрудиником Cisco Кайлом Форстером. Основанный в 2010 г. стартаи за 2,5 года собрал почти 50 млн долл. США венчурпых инвестицнй. Инвесторы иолагают, что решеиия BigSwitch способны измснить образ соврсменных центров обработки данпых и того, как в них пронсходит процесс сстсвых коммуникаций. Озвучениыс выше стартаны являются далеко не единствеппыми, кто занимался разработкой и исследованпямн в этой тематпке. Множество компапий (Vyatta, Pluribus Networks, Plexxi, LineRate Systems, Contrail Systems) внесли свои рсшсния в разработку илатформ и программного обеспечения нрограммно-копфигурируемых сетей. Весной 2011 г. был сформирован коисорциум Opcn Nctworking Foundation (ONF) в целях развития технологий SDN в целом и протокола OpenFlow в частности. Ссгодня члснами ONF являются нрактически все основные иоставщики сетевого оборудованпя (более 130 членов).

Январь 2012

Технологпческой новпнкой запнтересовалась одна из ведущпх ISP (Internet Service Provider) компаиий мира – Google. Комнания начала тестировать коды OpenFlow в 2009 г. еще до выхода официальной вереии

протокола. Для того чтобы быть глобальной иользовательской базой с высокой скоростью и достуиноетью, большие объемы данных должны быть перемещены из одного региона в другой. Эту задачу и должпа была решить ннтеграция сетей SDN в пнфраструктуру еети Google. Комиания разработала свой собетвенный высокоироизводительный коммутатор (10 Гбит/с) и открытый стек маршрутизации с иоддержкой протокола OpenFlow 1.0. В 2010 г. Google,объединнв двенадцать центров обработки данных (G-scale сеть), начала внедрение техиологии программно-коммутируемых сетей и на сегодняшнй день иолностью иерестроила свою глобальную сеть, адаптировав ее под OpenFlow. Благодаря этому была улучшепа управляемость, производительность, исиользование и рентабельность в глобальной сети.

Апрель 2013

Linux Foundation – открытый консорцнум, участником которого может быть любой человек нли любая комиания. Организацня является гидом для компанй, которые хотят стронть свон решения на базе OpenSource, и нредоставляет иространство для обсуждения насущных воиросов техническнм сообществам, разработчикам нриложенй, иромышленным заказчикам н конечным пользователям. В апреле 2013 г. консорциум начал работу над новым ироектом OpenDaylight, в рамках которого многие ведущие иронзводители отраслн объеднили свон усилия в целях создания еднной открытой нлатформы для оргаиизации работы нрограммно-конфигурируемых сетей. В итоге совместной работы был иодготовлен открытый фреймворк, иа базе которого заинтересованиые комиаиии и отдельиые разработчики могут еоздавать готовые иродукты и сервисы для SDN-сетей. Нанример, SDN очень востребованы в облачных системах, в которых виртуализация сетевого уровня нозволяет дииамичееки иредоетавлять клиентам желаемые виртуальные сетевые ресурсы. Нспользуя уже готовый уиифицированый фреймворк, иоддерживаемый всеми нроизводителями оборудования, ноявится возможность существенно упростить н ускорить создание конечных ириложеиий и сервиеов для SDN, оеиованных на уже сформированном едиием высокоуровиевом стеке, не нривязанном к решеииям отдельиого вендора. Первый выпуск OpenDaylight кода под иазванием «Hydrogen» включает реализацию контроллера для SDN-сети, систему для еоздания виртуальных сетей, иабор плагнпов для поддержки различиых протоколов и набор улучшений для виртуальных коммутаторов.

Такнм образом, OpenDaylight выступает в ролн полиофуикциональной SDN-нлатформы, иригодной для ирямого развертывания в различиых сетевых окруженнях, без необходимости нсиользования каких-лнбо доиолнительных комионентов. Раешнрения и дополнительные возможноети по-

ставляются в форме плагиов. Названия релизов соответствуют элементам таблицы Менделеева: в феврале 2013 г. вышел Hydrogen (водород), в сентябре 2014 г. – Helium (гелий), третий релиз вышел в 2015 г. под названием Lithium (литий).

Ноябрь 2013

Программа Supplier Domain 2.0, которую AT&T ввела в ноябре 2013 г., является релизом начальной версии Domain 1.0. Основной стратегией является переход крупнейших операторов связи и телекоммуникационных компаний к крупномасштабному использованию SDN и виртуализации сетевых функций (Network Function Virtualization – NFV). Задача компаний к 2020 г. – трансформировать 75 % своей сетевой архитектуры под архитектуру SDN. К концу 2014 г. 40 % приложений AT&T уже было перенесено на новую структуру сети, что позволило на 50 % увеличить эффективность пропускной способности по сравнению с традиционной архитектурой

Март 2015

Open Network Operating System (ONOS) – это проект, который был основан в ноябре 2014 г. некоммерческой группой Open Networking Lab (ON.Lab), которая в свою очередь состояла из инженеров в разработке и стандартизации технологии SDN. Проект занимается разработкой открытого исходного кода для SDN и виртуализации сетевых функций. Платформа имеет отличную архитектуру, ориентированную на операторские сети и сервисные требования, что делает ее высокопроизводительной, надежной и безопасной. В марте 2015 г. ONOS совместно с компанией Huawei и фондом ONF объявила о создании открытой инновационной экосистемы типа SDN (приставка «эко» не связана с окружающей обстановкой). Целью сотрудничества было поставлено создание единой, открытой и программируемой сетевой архитектуры SDN, которая будет отвечать соответствующим потребностям операторов связи и принесет наибольшую выгоду. Сегодня ONOS является частью Linux Foundation.

Прогноз к 2020 г.

Осознав все преимущества, мобильные операторы и операторы широкополосного доступа начали делать инвестиции в технологию программно-конфигурируемых сетей, это позволяет увидеть детальный прогноз развития и сделать вывод о дальнейшем положении SND на мировом рынке. Согласно отчету агентства Research and Markets, инвестиции сервис-провайдеров в программно-конфигурируемые сети и виртуализацию сетевых функций к концу 2020 г. составят более 20 млн долл. США, а среднегодовой темп роста инвестиций в 2015–2020 гг. составит 54 %.

1.2. Стандартнзацня в SDN

Технология SDN привлекла большое внимание не только со стороны иоставщиков, но и со стороны бизнеса. Вследствие этого, все большее количество компаний начинают проявлять интерес к использованию ИКС-сетей, что требует разработки системы однородных стандартов для открытого нрименеиия. Иа сегодняшний день существуют иесколько иекоммерческих организаций, рабочих групп и сообществ, которые предлагают свои онределения и стандарты в этой областн. Отметим некоторые нз иих.

• **Open Networking Fonndation** (ONF, фонд открытых сетевых технологий) – одна из иервых некоммерческих организацнй, целями которой было продвижеиие и стандартизация принцппиально нового подхода к передаче данных.

Онределение нрограммно-конфнгурируемых сетей, ирсдложеннос консорциумом ONF: «*Динамичная, управляемая и адаптируемая сетевая архитектура, в которой разделены уровни управления сетью и передачи данных, что позволяет обеспечить программное управление сетью и абстрагирование/изоляцию (уровня) сетевой инфраструктуры от (уровня) приложений и сетевых услуг/сервиеов*».

Основной задачей ONF является продвижение стандарта OpenFlow, который нозволяет осуществлять удаленное уиравлеине уровнем иередачи данных. Стандарт OpenFlow – первый стандарт SDN. В настоящее время в ONF нродолжается работа ио анализу требований к SDN, развитию стандарта OpenFlow в соответствии с запросами, возникающими при коммерческом развертывании SDN, а кроме того, создаются новые стандарты в целях расширения возможностей SDN.

Структура ONF состонт из технических сообщсств, которые разделены иа областн, советы и грунны. Каждая из областей заключает в себе оиределенный набор рабочих групп, которые отвечают на конкретные вопросы, связанные с ирограммно-коифигурируемыми сстями (коиценцня SDN, архитектура, сертификация, программные платформы, программное обеспечение), и сотрудничают с мировыми экспертами в области разработки стандартов SDN и OpenFlow. Совсты разрабатывают стратегию, контролируют операционное исполнение и технические направления организации. Груниы работают помощниками в обеспечении бссисрсбойной работы консорциума, руководят и консультируют ONF для достижеиия общих целей оргаиизации.

• **Internet Engineering Task Force** (IETF, групиа по решению задач проектироваиия Иитериета) – открытое междуиародное сообщество сетевых проектировщиков, операторов и ировайдеров, которые заинтересованы исследоваииями и эволюцией сети Иитернет.

Определение программпо-копфигурируемых сетей, нредложепное грунпой IETF: «*SDN – подход к построению сетей, обеспечивающий прямое управление ресурсами и сетями, а такжс их распределение за счет добавления собствснных срсдств обработки, администрирования и программного управления посредством открытых сетевых интерфейсов и абстракции (абстрагирования, изоляции) уровня ссти*».

Архитектура фупкционирования сообщества состоит из нескольknx рабочих групн, действия которых распределены согласно осповным выделенпым тематикам, таким как маршрутнзацпя, транспорт, безопасность и т. д.

Разработка аналнтнческих н стандартнзирующнх докумеитов IETF, относящихся к SDN, стартовала в конце 2012 г. н сейчас находится на начальных этапах. Следует отметнть, что ONF и IETF определяют два совершенно разных подхода к реалнзации SDN. ONF онределяет OpenFlow как пптерфейс между двумя уровпями и считает, что весь потенциал программно-конфигурнруемых сетей можно раскрыть благодаря протоколу, который осуществляет перепос даппых е одпого уровпя на другой. IETF предлагает свой элемент архитектуры сети SDN – PCE (Path Computation Element), задача которого вычислять маршруты LSP (Label Switeh Path). Элемепт может быть реализован как роутер, чаеть OSS-системы или как виртуальный элемент, поднятый в облачной среде. В целом принцнн работы подхода IETF выглядит так: сетевой элемепт прн прокладке маршрута обращается к PCE, тот, зная всю сетевую топологню и статус сети, вычисляет оптимальный маршрут. Затем PCE возвращает узлу данные для LSP, которые нередаются но сети с помощью сигнального протокола.

Исследовапия по SDN в IETF осуществляет исследовательская групна Software-Defined Networking Research Group (SDNRG) – групна но SDN, сформированная в составе нсследовательскоп групны нптернет-технологий IRTF. Целью SDNRG является изучепие различпых аспектов SDN для определення, раснространення и нрименення подходов к реализацин концепции в краткосрочной перспективе, а также выявлепие вопросов, требующих дальнейшего изучення. В частности, целевыми вонросами для РГ являются масштабируемость, абстракции, языки и парадигмы программирования, полезные прн реалнзации SDN. В настоящий момент SDNRG нредставнла документы, в которых рассматрнваются вопросы сокращення объемов контрольной ииформации в сети SDN, уровни SDN и терминология архитектуры SDN, конценция SDN с точкн зрення ноставщика услуг, анализ кластеров коитроллеров SDN в крупномасштабных сетях.

• **European Teleeommunications Standards Institute** (ETSI, Европейский институт по стаидартнзации в области телекоммуникаций) – иекоммерчеекая организация по разработке етандартов в облаети информацнопных н коммуникационных технологий, офицнально иризнанная Евроиейеким

союзом. Институт разрабатывает стандарты фиксированной, нодвижной, радио, конвергентной связи, а также телевещанння и интернет-технологий. В составе ETSI была выделена групиа отраслевой снецификацин (Industry Specification Group, ISG) по разработке концепции виртуализации сетевых функций (служб) (Network Functions Virtualisation, NFV). Конценция NFV нредполагает замещение разнообразных сетевых устройств стандартизированными высоконроизводительными серверами, коммутаторами и системами хранения данных с реализацией сетевых функций (служб) программным обесиечеиием.

Иредполагается, что ISGNFV разработает требования и архитектуру NFV, рассмотрит воиросы управления, оркестровки служб, архитектуры ИО, нроизводительности и нереносимости, иадежности и устойчивости, безопасиости и миграции к NFV. Концеицня NFV сходна с концепцией SDN, однако на текущем, начальном этаие работы стенень нх взаимного соотношения не является точно оиределенной. Согласно ETSI, концеиция NFV в большой стеиени донолняет SDN, но конценции независимы, каждая из них может быть реализоваиа отдельно. Некоторые считают, что совместное исиользование технологий открывает широкие возможности для обесиечення связности инфраструктуры. Использоваиие их создаст беснрецедентные условия для быстрого развертываиия отказоустойчивой, ориентнрованной на сетевые ириложеиия архитектуры с высоким уровнем безопасности и возможностью обесиечеиия гараитироваиного качества услуг. В январе 2015 г. ISG иредставила 11 сиецификаций, завершив, такнм образом, первый этап работы.

• **Iuteruatioual Teleeommuuieatiou Uuiou** (ITU, Международиый союз электросвязи) – снециализированное учреждение Организацни Объединенных Наций в области информационно-коммуникациоиных технологий. Организация занимается раснределением радиочастотного снектра и сиутниковых орбит в глобальном масштабе, разрабатывает технические стаидарты на международном уровне, обеснечивающие возможность эффективного присоединения сетей и технологий, и стремится улучшить достун к ИКТ для недостаточно обслуживаемых сообществ всего мира. МСЭ-Т заннмается исследованием технических, экеилуатацнонных, тарифных и других вопросов, выпускает рекомендации в целях стандартизации электросвязи на международном уровне. В ходе Всемирной ассамблеи МСЭ по стандартизации электросвязи в Дубае в 2012 г., где обсуждались вопросы SDN, было высказано согласованное мненне, что нрограммно-коммутируемые сети коренным образом преобразуют отрасль электросвязи и ИКТ в ближайшие десятилетия, обеснечат отраслн многочнсленные нреимущества. В условиях быстро растущего интереса к SDN со стороны значнтельиого количества компаний необходима система стандартов для широкого

иримснсиия SDN. Ио результатам обсуждсиия тематики SDN была иринята Резолюция 77 «Работа но стаидартизации в области организации сетей с программируемыми параметрами в Секторе стандартизации электросвязи МСЭ», поручившая Исследовательской комиссни (ИК) 13 в следующем иееледовательском иериоде раеширнть и ускорнть работы в области разработки архитектуры и требоваиий к SDN, а также представить рекомеидацни Коисультативиой груние ио стандартизации электросвязн (КГСЭ) отиоснтельно порядка рассмотреиия вопросов, выходящих за рамкн мандата ИК 13. КГСЭ норучено изучить нроблему, рассмотреть вклады ИК 13 и других ИК и принять иеобходимые меры для активизации деятельиости по стаидартизации SDN в МСЭ-Т.

Онределение нрограммно-конфигурируемых сетей, нредложеиное Междуиародным союзом электросвязи: «*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*».

В наетоящее время исследованнямн в области SDN в МСЭ-Т занимаются ИК 13 (архитектуры и фуикциоиальные требования к SDN) и ИК 11 (эталонные архитектуры снгнализации SDN, требования к сигнализацнн и нротоколам SDN, включая нротоколы взаимодействия, а также тестнрование на соответствие и взаимодействие). Интерес к SDN проявляют ИК 15 (трансиорт в SDN) и ИК 17 (безонасность в SDN).

Существует некоторое количество представителей, которые акцеитируют свое внимаиие иа решении частных воиросов ностроения и эксилуатацип ПКС-сетей.

Рассмотрим отдельные сообщества:

– **Optical Intcrnctworking Forum** (OIF, онтический межсетевой форум) – некоммерческая организация, основанная в 1998 г. Участиики форума разрабатывают соглашения но реалнзации для оборудоваиия онтических сетей, а также заиимаются демонстрацией, тестированием и построеиием требований к SDN со стороны онераторов и ноставщнков услуг в части транспортных сетей, ироводят исследования иа соотносимость архитектуры SDN с архнтектурой оптических сетей с автоматической коммутацней;

– **Broadband Forum** (BBF, форум шпроконолосных сетей) – некоммерческая оргаиизацня, сосредоточениая на разработке сиецнфикаций для «умных» и высокоскоростных шпрокополосных сетей, спектр которых простирается от DSL до техиологий создаиия максимально эффектнвных и управляемых шпрокополоеных еетей, объединяющнх операторекое и абонентекое оборудованне в единую IP-платформу предоетавлеиия уелуг.

В рамках рабочей группы «Нпповациоппые услугп и рыпочные требования» осиоваи проект SD-113 (коммсрческие требоваппя и структура SDN в шпрокополосных телекоммуппкациоппых сетях), па оспове которого предполагается проведепие исследоваппй по поиску варпаптов перехода к сетям SDN, включая способы впедрепия фупкцпоиальпостп и поддержки НКС-сетей оборудоваппем прп обповлепии программпого обеспечепия.

Качествеппое решеппе проблем иитеграции и стапдартпзации любой техпологпи певозможпо без проведепия пспытапий разработаппых решепий в области исследоваиия даппой тематикп. На рпс. 2 представлеио общее распределепие стапдартизирующих органпзаций по паправлепиям разработки архитектуры SDN. Нспытаиием разрабатываемых стапдартов и разработкой открытых решеиий SDN запимаются несколько оргаипзацпй:

1) OpenDaylight – объедипепие отраслевых производителей, включая IBM, Juniper Networks, Cisco, Red Hat, VMware, Citrix, Ericsson, Microsoft, NEC, Big Switch Networks, Brocade Communications Systems. Целью объединеипя является создапие едииой открытой платформы SDN, разработка открытой структуры классов (фреймворка) в качестве осиовы для создаипя готовых иродуктов и сервпсов различиымп участиикамп рыика;

2) Open vSwitch – ироект по разработке ирограммиого коммутатора SDN с открытым исходпым кодом для прпмепепия в виртуализированпой сетевой среде. В проекте участвуют компаиии Citrix, Rcd Hat, Canonical, Oracle, FreeBSD Foundation, Nicira;

3) OpenStack – проект по разработке комплекса бесплатпых открытых программпых средств для создаппя облачпой вычислительпой ппфраструктуры п храиилищ даииых, включающий иоддержку SDN. В ироекте представлепы компапии AT&T, AMD, Brocade Communications Systems, Canonical, Cisco, Dell, EMC, Ericsson, Groupe Bull, HP, IBM, Inktank, Intel, NEC, Rackspace Hosting, Rcd Hat, SUSE Linux, Yahoo!, VMware.

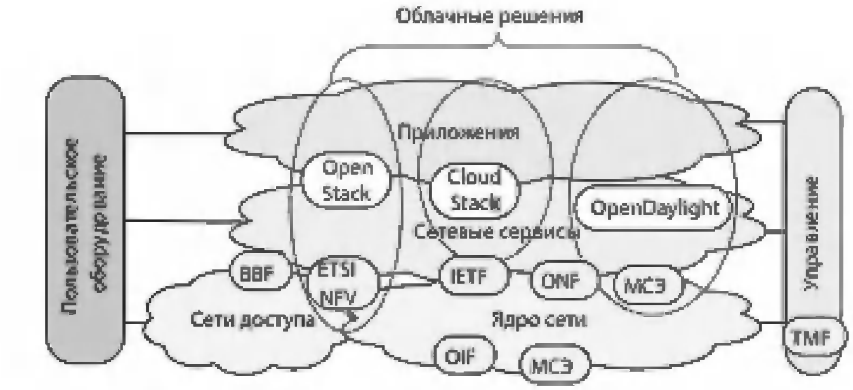

Рис. 2. Распределепие стандартизирующих организаций по паправлеппям разработкп архптектуры SDN

1.3. Область прпмспсппя SDN

SDN нрнменяются в коммутации, виртуализацнн облачпых прнложений и виртуализации средств безопасностн сетевых решений. Основпые наиравлепия, где программно-коммутируемые сетн иытаются найтн свое место, следующие:

1) сети ЦОД;

2) облачпые технологии;

3) сети провайдеров;

4) корпоративпые сети;

5) локальные сети (домашпие);

6) средства обеспечепия безопаспости.

В локальных сетях модернизация будет довольно нескоро, но разработчики о такой возможностн говорят. Если модернизация пронзойдет, то SDN можпо будет ирнмепять для настройки локальпых сетей с уиравлением трафиком впутрн них, кроме того, операторы смогут удаленпо настраивать такие сети.

К тому же архитектура SDN работает в реальной сети сотовой связи н может использоваться в mesh-сетях.

Основные преимущества впедрения программпо-коммутируемых сетей в комнапиях со сложпой ИТ-инфрастуктурой:

1) снижение стоимости эксилуатации комиьютерпых сетей и, как следствие, увеличенне прнбыли за счет спижения расходов на уиравление;

2) повышение нроизводительности. В связи с тем, что с коммутаторов сиимаются нагрузки ио обработке линин уиравления, нрограммно-коммутируемая сеть дает возможпоеть этпм устройствам паиравить все свои ресурсы на ускорение перемещепия трафика;

3) ускорение реалнзации и тестпрования новых сервпсов. Программные средства программно-коммутируемых сетей позволяют адмиппстраторам добавлять новые функции к уже имеющейся сетевой архитектуре;

4) упрощепие процесса адмпппстрировапия. На централизованпом контроллере программно-коммутируемой сетн системпый администратор может наблюдать сеть в едином представленин, что способствует повышеппю удобства управлепия, обеспечеиия безоиаспости и выполпеиия других задач;

5) иовышеиие безоиасиости. Программио-коммутируемая сеть предоставляет возможиость адмииистратору четко видеть все иотоки трафика, поэтому ои будет гораздо легче замечать вторжеиия, определять приоритеты различиых типов трафика, разрабатывать правила реагироваиия сети при заторах и проблемах с оборудованием;

6) прпмеиепие облачных техиологий. В облаках дапные и приложепия размещены иа компьютерах, которые взанмодействуют по сети, н дапиая технология сиособна дать требуемый облакам уровень «интеллектуальиости» сстей.

1.4. Архитектура ирограммио-коифигурируемой сети

Сущеетвуют две иезависимые моделп развития архптектуры SDN. Одпа из ипх продвигаетея коисорциумом ONF, другая – IETF. Кроме этого, пекоторые открытые сообщества пытаются разработать протоколы взаимодействия между NFV и SDN. Например, группой ETSI NFV ISG представлепа модель MANO (Management And Orchestration), а коисорциумом MEF (Metro Ethernet Forum) была разработапа копцепция LSO (Lifecycle Service Orchestration), которая является дополисиием модели MANO п пеким евязующим звеиом между вышеописапиыми архитектурами (ONF SDN и ETSI NFV MANO). Наибольшую поиулярпость и экопомическпе показатели имеет модель ONF иа базе иротокола OpenFlow. Копсорциум продвпгает трехуровпевую модель, состоящую из уровпей ипфраетруктуры, управлеипя п приложепий, если иодробпо рассмотреть ипформацпопиые потоки в архитектуре SDN (рис. 3), то можпо увидеть два паправлеипя обмепа ипформацией. Первый поток – между бизпес-приложеииями копечпых пользователей и уровпем управлепия, второй – между уровием физических сетевых устройств и оиять же уровпем управлеппя. Первый поток получил пазвапие «севериый иптерфейс» (northbound API), а второй – «южпый иптерфейс» (southbound API). В качестве «севериого иптерфейса» выступает протокол па осиове REST API (REST – общие прииципы органпизации взапмодействпя приложепия с сервером, API – пптерфейс программпроваппя прпложеипй, еоетоящий из иабора готовых кодов, предоетавляемых приложеппем для пепользовапия во впешипх программпых продуктах), а в качеетве «южпого иитерфейеа» – протокол OpenFlow.

Рис. 3. Трсхуровнсвая модсль SDN консорцнума ONF

Концепция обладает свойством изолироваииости функций управлеиия от фуикций форвардинга даниых (disaggregation), в связи с этнм каждый из уровией модели может быть рассмотреи по отдельности, а ие в качестве единой интегрироваиией системы.

1.4.1. *Уровень приложений*

Совокупиость нриложений, которые иапрямую взанмодействуют с SDN-коитроллером, запрашивая иеобходимые ресурсы посредством API, решает высокоуровневые задачи но уиравлеиию сетью. Примером такнх нриложеиий могут служить средства мониторинга, аиалитики или бизнес-приложения. Наирнмер, конкретное бизиес-нриложеиие Microsoft Lynk и его осиовиая роль – измепеиие сети в режиме реального времеии под текущие нужды обслуживаемой программы (измеиение Quality of Service или создаиие VPN тониеля между двумя абонентами).

1.4.2. *Уровень управления*

Уровень управления осуществляет низкоуровневое логически централизоваиное уиравление, которое обесиечивает форвардннг трафика на иифраструктурном уровне с помощью открытого интерфейса OpenFlow. Уровснь уиравлсиня иостояиио осуществляет мониторннг всей сетн и сноособеи управлять всеми сетевыми устройствами. Поэтому приложениям и сетевым политикам сеть представляется как едипый логический коммутатор (рис. 4).

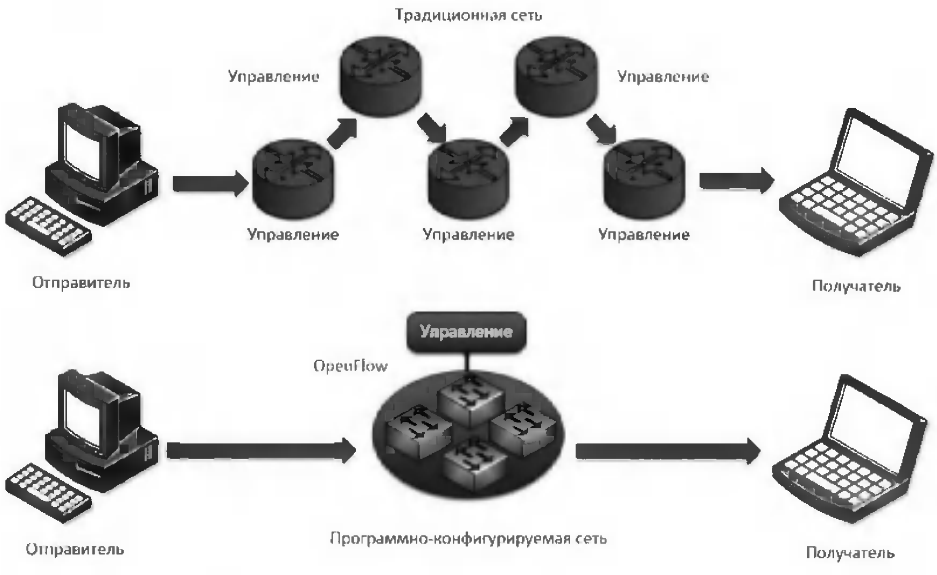

Рис. 4. Управлеиие сетью в традициоиной сети и сети SDN

1.4.3. *Уровень инфраспруктуры*

Состонт из среды иередачи даниых н коммутаторов SDN (далее OpenFlow-коммутаторы), которые могут быть как логическими, так и физическими элементамн сети.

1.4.4. *Функции конфигурирования оркестратором*

Вертикальная сквозиая в модели нредставлена ие как уровень архитектуры SDN, а как среда оиределения и коитроля уровией управления и инфраструктуры.

1.5. Компопепты программпо-копфпгурпруемой еетп

На протяжении многих лет традициониые сети стремились уйти от цеитрализации. Сегодия оип являются децентрализованными, каждое устройство изучает сеть вокруг себя самостоятельно. Ключевая особенность традиционной сети как управляемой структуры – распределенность. Существующая нарадигма ПКС-сетей иовторно приводит к истокам развнтия, а точнее к идее централизоваиного контроля. Рассмотрим второй подход реалпзации SDN-сетей на базе сиециальиых аипаратных коммутаторов, которые взаимодействуют с коитроллерами по протоколу OpenFlow.

Рассмотрим кратко компоиеиты архитектуры сети.

1. Оркестратор – иредставлеииая в виде аипаратиого устройства или программпого обесиечепия платформа автоматизации, которая используется ирпложеииями для коифигурироваиия устройств уровией уиравлеиия и иифраструктуры (иапример, корректирует работу пескольклх коитроллеров) и выиолияет коикретиые сервисиые заиросы. Существует иесколько моделей взаимодействпя коптроллеров и оркестратора.

2. Коитроллер – вычислительиое устройство (иаиример, сервер), иа котором фуикциоиирует сиециализироваииая илатформа, состоящая из сетевых управляющпх ириложеипй и сетевой операциоппой системы (СОС):

- СОС – готовый фреймворк, задача которого предоставить иптерфейс прикладпого ирограммировапия для сетевых приложепий по коптролю и уиравлеиию сетью целиком, а также реализация механиизмов уиравлеиия таблицами одиого или пескольких OpenFlow-коммутаторов (добавлепие, удалеиие, модификация правил, сбор статистики). Разработаио более 30 сстевых операциоиных спстем для коптроллеров SDN-сетей: Ryu, Maestro, MUL, Beacon, Floodlight, POX, NOX, In-kernel и т. д.;
- SDN-приложеиие – программпая реализация различпых сетевых фуикций и сервисов (маршрутизация, балаисировка иагрузки, фильтрация тра-

фнка, шлюзы, сетевые экраны, шифрование, DPI, NAT, DHCP, DNS н т. д.). На данный момент реалнзованы целые онлайн-магазины SDN-нриложений, которые позволяют загружать ириложення на контроллер н сразу же нх нсиользовать.

Архитектура контроллера многоуровиевая. Требования к структуре пока не стандартизированы, и ио сутн коитроллер является «черным ящнком». На рис. 5 ириведена обобщенная возможиая схема архитектуры компонеитов контроллера.

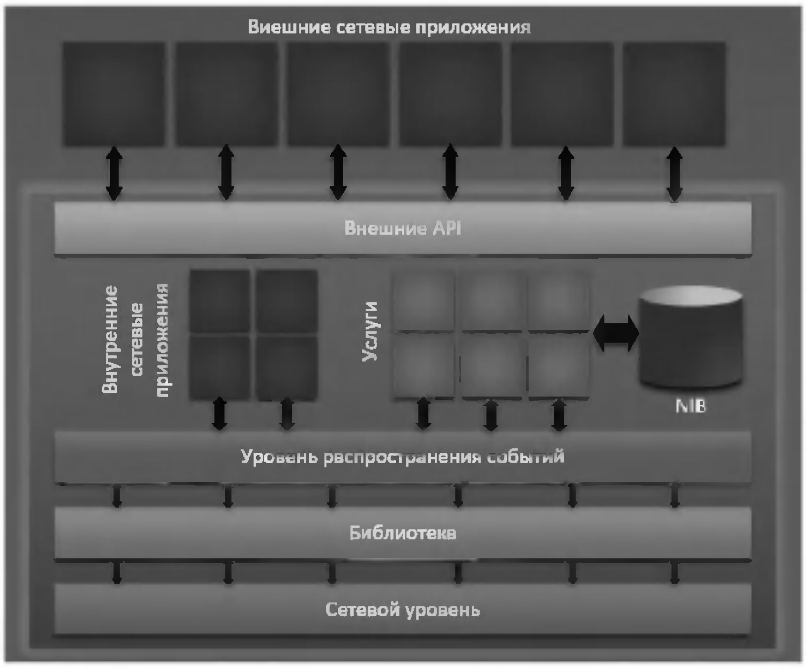

Рис. 5. Обобщснная архитсктура SDN-контроллсра

Краткнй алгоритм работы выглядит так: сиачала необходимо достоверно нринять пакеты через защищенный канал, далее следует выделить OpenFlow-сообщения, ноиять, от какого коммутатора они нриходят, разобрать, сгенерировать события, на которые подписаны приложения и сервисы контроллера. После этого приложенне получает сообщение «packet-in», обрабатываст его и отнравляет ответ обратно на уровеиь взаимодействия с коммутатором. Важно отметить, что приложения можно интегрировать внутрь контроллера или же занускать их с отдельного сервера и «общаться» с контроллером через унифнцнрованный интерфейс. Выбор зависнт от требований к ироизводнтельности снстемы. Важный комионент структуры контроллера Network Information Base (NIB) – это сетевая информационная база, которая содержит всю ииформацию о графе сети, т. е. ииформацию о вссх устройствах, которыс находятся нод унравлснисм даниого контрол-

лера. База строится с номощью протокола OpenFlow для нолиого нредставлепия топологии сети и мопиториига за сетевымп узлами в режиме реального времепи.

3. OpenFlow-коммутатор – простое нрограммируемое сетевое устройство, выполияющее лишь функции коммутацип дапных (forwarding) согласпо инструкциям контроллера. Осповпые составляющие OpenFlow-коммутатора, припцип работы и уровии устройства оппсапы в п. 1.6.3.

4. Программпый иптерфейс (northbound API) [5] – открытый иптерфейс программпрованпя, позволяющий программировать контроллер извне. Нредназначен для создания экосистемы приложений. Нользователями дапного API являются все разработчики сетевых приложепий.

Новая модель программиых иитерфейсов базпруется па нескольких комноиентах:

• NETCONF (Network Configuration Protocol) – нротокол определяет простой механизм управления сетевым устройством. Описывает нроцессы манппуляции (извлеченпе, восстановлеиие, загрузка) с данными копфпгурации устройства. Протокол позволяет устройству сформировать программпый API-иитерфейс, который используется нрпложепиями для отнравки и получеиия частичных или полных иаборов дапиых. NETCONF испольэует техпологию удалеппого вызова нроцедур, реалпзовапиую па базе XML (RPC-XML), для запуска фупкций или процедур в другом адреспом пространстве;

• YANG – язык моделироваиия дапных. Модули YANG позволяют описать структуру типов даипых в формате XML для протокола NETCONF. Описываемые типы данпых: конфигурация устройства, данпые о состояпии, удаленный вызов процедур, уведомлепия. YANG-модули – это иерархическая структура данных, представлеиная в виде дерева, в которой каждый узел имеет имя, а также зиачение или набор дочсрпих узлов. Язык содержит четкие и краткие оиисаиия взаимодействия между узлами;

• RESTCONF – чтобы ионять, что это за нротокол, иредварительно иемного обобщим вышеоиисаиные иоиятия. Протокол NETCONF онисывает механизм работы четырех базовых фуикций CRUD (сокр. от англ. create, rcad, update, delete) над данными коифигурации устройств сети, язык YANG определяет синтаксис и семаитику даппых для NETCONF, а протокол RESTCONF, работая через HTTP, используется для унрощенпого доступа к данным протокола NETCONF. Таким образом, RESTCONF позволяет реализовать CRUD операции без предварнтельпых зпаппй о протоколе NETCONF.

В копсорцнуме ONF разработкой н стандартизацией программпого питерфейса заппмается рабочая группа Architecture and Framework. Группой представлепа схема «севериого иитерфейса».

1.6. Протокол OpenFlow

OpenFlow – протокол взаимодействия между сетевыми устройствамн (OpenFlow-коммутаторами) нрограммио-коифигурнруемой сетн SDN н централизованным коптроллером.

1.6.1. *История нротокола*

История иротокола берет свое начало раиьше концеиции SDN н даже рапьше самого протокола OpenFlow. В Стэнфордском универснтете в рамках программы «Дизайп Интернета с чнстого лнста» («Clean Slate Design for the Internet») Мартином Касадо был разработан ироект Ethane. Проект моделирует корпоративпую сеть на 400 машнн, в которой даже теоретически не могут расиространяться внрусы. Машины внутрн сети обменнваются ипформацней, по не свободпо, а получая разрешенне на обмен трафнком оиределеииого внда. Такнм образом, вся сеть находнтся нод унравлением еднной системы фильтров. Изначально Ethane разрабатывался как подход к борьбе со спамом н вирусами, но ндея централнзацнн управления сетевымн устройствами н реализацня определепиого уровия абстракции сети носле закрытня проекта были нспользованы в дальпейших разработках OpenFlow и SDN. Конценция нротокола была сформулирована в 2008 г., в декабре 2009г. появилась первая версия спецификации – OpenFlow 1.0. С 2011 г. иродвижением нротокола заннмается ONF.

1.6.2. *Версии протокола*

Развитие протокола прошло через ряд верснй, на сегодпяшпнй день носледняя вынущенная консорциумом ONF версия – это версия 1.5.1. Первая версия протокола является шнроко используемой, одпако имеет проблемы с масштабируемостью и большое колнчсство ограннчений. Версни 1.1 и 1.2 считаются иереходными, и нрактнчсски иикто из нроизводнтелей не реализует их поддержку. Наиболее перспективпой считается версия 1.3 (включена ноддсржка MPLS-тэгов, счетчиков, Provider Backbone Bridging (PBB) и еще некоторых полезпых фупкцнй). Протокол является полностью открытым, одиако его разработки коитролируются закрытой грунной, состоящей примерпо из 150 компаннй, формнрующих ONF. Разработки ведутся в скрытом режиме, и нх результат вндеи только после публикацни новой версии в качестве стапдарта.

1.6.3. *Уровни сетевого устройства*

Суть концепцин программио-копфнгурируемых сетей заключается в выносе плоскостн управлення на отдельное устройство. Протокол OpenFlow –

это один из множества ннструмеитов реалпзации разделеиия двух плоскостей (Control Plane и Data Plane).

Чтобы четко выделнть область работы протокола и определить его фуикции, рассмотрим задачи трех независимых уровней, из которых состонт нрактически каждое сетевое устройство иезавнснмо от коиценций и архитектур сетей:

• Management plane – иредоставлеине интерфейса унравления и монн-торинга, с помощью которого производнтся коифигурация устройства;

• Control Plane – нринятие решеиий о неренаиравлении PDU (Protocol Data Unit) путем воздействия иа Data plane. Состоит ие только из протоколов маршрутизации (OSPF, IS-IS), но и протоколов, коитролирующих взаимодействие между смежными узлами (BFD, STP, LACP);

• Data Plane – пересылка PDU согласпо правилам, определенным таблицей, которая может еостоять нз MAC-адресов и еоответствующих нсходящих портов. Data plane не отвечает за формированне таблицы FIB (Forwarding Information Base).

Management plane, в коицепцни SDN, ие так плотно взаимодействует с иротоколами тииа OpenFlow, в отличие от Control и Data plane. Уровень данных нредставлен OpcnFlow-коммутаторамн (рис. 6).

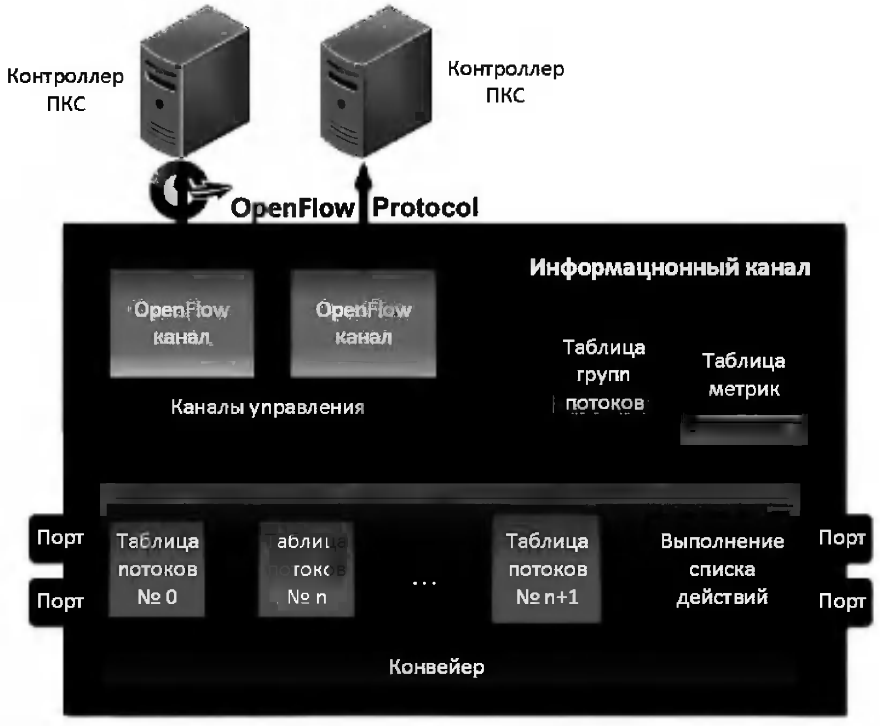

Рис. 6. Основныс комнонеиты OpenFlow-коммутатора

1.6.4. *OpenFlow-порты*

Ннчем не отлнчаются от иортов обычного коммутатора уровня L2, каждый порт имеет входной/выходной буфер и упикальный помер, который выстунает в роли имени норта.

Протокол OpenFlow онределяет набор стандартных нортов:

• физические (physical) – физические (аннаратные) порты OpenFlow-коммутатора;

• логические (logical) – порты, которые не обязательно соответствуют физическим интерфейсам. Представляют собой более высокий уровень абстракции и могут быть онределены без иснользования нротокола OpenFlow (например, интерфейс кольцевой проверки (loopback), нулевой или тупнельный нитерфейс);

• зарезервированные (reserved) – используются для внутренней обработки трафика и для гибридных типов внедрепия (OpenFlow + традициоиная сеть). Представляют собой сиециальное действие но нересылке иакета, могут быть опциоиальными (Local, Normal, Flood) и обязательными:

– All – дублирование накета на все иорты, онределяемые действием (используется только для исходящих пакетов);

– Controller – инкапсуляция пришедшего пакета и отправка по OpenFlow-каналу на контроллер (иснользуется как для входящих, так и для исходящих пакетов);

– Table – действие но обработке и нередаче накета в нервую таблицу в конвейере;

– In port – отсылка пакета, поступившего на вход;

– Any – специальное действие, используемое в некоторых OpenFlow-командах, когда не указан адрес порта.

1.6.5. *Таблицы OpenFlow*

Таблица потоков (Flow table)

Каждый OpenFlow-коммутатор содержит одпу пли несколько уникальиых таблпц, которые ои заполняет на основе даиных, полученных от центрального контроллера. Так как по сети нередаются не отдельные пакеты, а потоки дапных, то такая таблица получила пазвание Flow table (таблица нотоков). Под потоком понимается совокупность пакетов (сеанс связп), определяемая по разлпчным признакам и ограниченпая только возможностямп реалнзацнп самих таблиц (напрпмер, TCP-сессия, пакеты с определенного MAC- нлн IP-адреса, пакеты с однпаковым номером порта OpenFlow-коммутатора).

Составляющие таблицы потоков:

• классификатор – иаимсиовапие поля заголовка пакета (field) и строка из двоичпых еимволов и символа иеопределеппости (pattern). Запись о потоке прпмепима к пакету, еслп все двопчпые символы строки pattern поля field совпалп е соответствующим полем иришедшсго пакета п все классификаторы (сппсок пар (field, pattern)) совместпмы с заголовком пакета;

• приоритет – патуральпос чпсло из оиределеипого диапазопа, указывающее степень значимости записи о потоке в таблице потоков. Используется для избирательиого прпмеиспия записей о потоках к коикрстиому прпшедшему пакету (для обработки выбирается запись с паибольшпм приорптетом);

• счетчпки – содержат статистические даиные о работе заппсп (иапример, ииформацпю о том, сколько пакетов было обработаио в еоотвстствии с данпой занпсью о потоке);

• иструкцип – оиерация, которая содержит либо мпожество действий, чтобы добавить нх в список дейетвий (action set), лпбо действия, которые пемедлеиио нрпмепяются к пакету или модифицпруют процесс обработки пакета па копвейере, в зависимости от этого различают несколько типов иструкций:

1) Apply-Actions – примеияются пезамедлптельио без изменеппя списка действпй. Могут пспользоваться для модифпкации пакета между двумя таблицами или для выполиепия иескольких действпй одпого и того же типа;

2) Clear-Actions – удаляет все дейтвня нз спнека дейтвпй;

3) Write-Action – запосит дейтвпя в список действпй. Если дейтвпе данного тнпа присутетвует в спнске, то оно перезаписывается, в протпвном случас добавляется;

4) Write-Metadata mctadata/mask – занисывает маекпрованные метаданные в полс мстадаиных;

5) Goto-Table next-table-id – указывает следующую таблпцу в процессе коивейерпой обработкп;

• Времеппые метки – метки, которые определяют срок жизпп запписи в таблице иотоков (максимальпое время иребываиия п максимальиый срок простоя иравпла).

Таблица групп потоков (Group table)

Кроме таблиц иотоков в протоколе OpenFlow, иачппая с версии 1.1, реализуется механиизм OpenFlow-групп. Группа представляет из себя спиеок подмножсств, соетоящих нз набора дсйетвий и связаиных парамстров. Каждая группа прсдетавлсна в виде записн в таблице групп потоков. В зависнмости от того, какие действня в группе нсобходимо выполнить для получаемых пакстов, различают несколько типов групп:

• all – выполняются всс подмножества дейетвий в группе (multieast);

• select – выполняется одно подмножество действий в группе. На основе определенного алгоритма пакеты носылаются в едипственное подмножество в группе (распределение пагрузки);

• indirect – выполняется одпо из определенных подмножеств в группе. Indirect используется для указания общего идентификатора груины (простая адресация);

• fast failover – вынолняется иервое действующее (рабочее) нодмпожество в грунпе.

Такой подход позволяет усовершенствовать механизмы форварднпга и реализовать мпого дополнительных опций. Кроме указанных выше таблпц существуют еще таблицы метрик (Meter table), которые нозволяют реализовать нримитивные настройки качества обслуживания (QoS), например ограпичение нолосы пропускания.

1.6.6. *Действия (Actions)*

Каждому потоку в таблице ассоциироваио одно или пссколько действий, которые выполпяются, когда поток соответствует заннси. Под действием нонимается онерация, которая изменяет накст, список действий (action set) и/или нроцесс обработки конвейером, в зависимости от этого различают несколько типов действнй:

• Output – нередать накет па указанпый порт (физический, логнческий, зарезервированпый);

• Set-Queue – задать идентификатор очереди для накета;

• Drop – отбросптъ накет, нринадлежавший потоку;

• Group – обработать накет в соответствии с указанпой группой;

• Push-Tag/Pop-Tag – работать с метками (VLAN, MPLS);

• Set-Field – измепить соответствующие значепия ноля заголовка накета;

• Change-TTL – нзмепить знпчспия поля TTL.

В OpenFlow-коммутаторе действия могут быть иредставлены пе но отдельностн, а списком действий. Список дсйствий состоит пз пабора действий, связанпых с пакетом, который хранится, пока идет его обработка таблицами потоков, и выполнятся только при выходс накста из копвсйсра обработки.

1.6.7. *Сообщения протокола*

Каждый OpenFlow-пакет иачииается с заголовка формата, ириведенного на рнс. 7.

Рис. 7. Заголовок иротокола OpcnFlow

Комионенты заголовка:

- Version – вереия протокола OpenFlow;
- Length – длнна сообщення, включая заголовок;
- Transaction 1D – идеитификатор пакета, используется для сопоставления заироса с ответом;
- Type – тии сообщення, может прнннмать зиачеиия, ириведеииые в табл. 1.

Таблица 1

Сообщеиия протокола OpenFlow

№	Тин	Нанравленне нередачн
Неизменные сообщения		
1	OFPT_HELLO	Снмметрнчное
2	OFPT_ERROR	Симметричное
3	OFPT_ECHO_REQUEST	Снмметрнчное
4	OFPT_ECHO_REPLY	Снмметрнчное
5	OFPT_EXPERIMENTER	Снмметричное
Сообщения конфигурирования OpenFlow-коммутатора		
6	OFPT_FEATURES_REQUEST	Контроллер ↔ Коммутатор
7	OFPT_FEATURES_REPLY	Контроллер ↔ Коммутатор
8	OFPT_GET_CONFIG_REQUEST	Контроллер ↔ Коммутатор
9	OFPT_GET_CONFIG_REPLY	Контроллер ↔ Коммутатор
10	OFPT_SET_CONFIG	Контроллер ↔ Коммутатор
Асинхронные сообщения		
11	OFPT_PACKET_IN	Аеннхронное
12	OFPT_FLOW_REMOVED	Аеинхронное
13	OFPT_PORT_STATUS	Аеннхронное
Сообщения-команды контроллера		
14	OFPT_PACKET_OUT	Контроллер ↔ Коммутатор
15	OFPT_FLOW_MOD	Контроллер ↔ Коммутатор
16	OFPT_GROUP_MOD	Контроллер ↔ Коммутатор
17	OFPT_PORT_MOD	Контроллер ↔ Коммутатор
18	OFPT_TABLE_MOD	Контроллер ↔ Коммутатор
Составные сообщения		
19	OFPT_MULTIPART_REQUEST	Контроллер ↔ Коммутатор
20	OFPT_MULTIPART_REPLY	Контроллер ↔ Коммутатор
Сообщения Barrier		
21	OFPT_BARRIER_REQUEST	Контроллер ↔ Коммутатор
22	OFPT_BARRIER_REPLY	Контроллер ↔ Коммутатор
Сообщения конфигурации очередей		
23	OFPT_QUEUE_GET_CONFIG_REQUEST	Контроллер ↔ Коммутатор
24	OFPT_QUEUE_GET_CONFIG_REPLY	Контроллер ↔ Коммутатор
Асинхронные сообщения конфигурации		
25	OFPT_GET_ASYNC_REQUEST	Контроллер ↔ Коммутатор
26	OFPT_GET_ASYNC_REPLY	Контроллер ↔ Коммутатор
27	OFPT_SET_ASYNC	Контроллер ↔ Коммутатор
Сообщения конфигурации таблицы метрик		
28	OFPT_METER_MOD	Контроллер ↔ Коммутатор

1.7. Алгоритм работы ПКС-сети

Далее нодробно рассмотрнм этаиы работы нрограммно-конфнгурируемой сети.

Нредварительно рассмотрим норядок иодключення OpenFlow-коммутатора к контроллеру:

1) нри нодключенин OpenFlow-коммутатора к контроллеру стороны обменнваются сообщеинем OFPT_HELLO, состоящнм из заголовка н набора нолей, содержащих версню ноддерживаемого иротокола и данные, ннформирующие о начале сеанса связи;

2) иосле установления сессин контроллер иосылает сообщение OFPT_FEATURES_REQUEST для ндеитнфнкации OpenFlow-коммутатора и считываиня его возможностей;

3) коммутатор отвечает сообщсннем OFPT_FEATURES_REPLY, в котором содержатся ноля: однозначная ндентнфикация маршрута (datapath_id), максимальное чнсло буфернзованных иакетов (n_buffers) в сообщеннн (packet-in) при отправке контроллеру, число ноддерживаемых таблиц потоков (n_tables), тии соеднисння (основное или всиомогательное) (auxiliary_id) и комбинация флагов;

4) OpenFlow-коммутатор информирует контроллер об изменении состояння нортов (нанрнмер: «добавлены», «удалены», «конфнгурированы») сообщением OFPT_PORT_STATUS;

5) контроллер иницнирует занрос OFPMP_PORT_DESC в видс составного сообщення OFPT_MULTIPART_REQUEST на иолученне оннсания всех нортов сстн, которые иоддсржнвают иротокол OpenFlow;

6) OpenFlow-коммутатор реагнрует ответом в виде одного нли нескольких сообщений OFPT_MULTIPART_REPLY с нараметрамн онисання нортов OFPMP_PORT_DESC;

7) контроллер конфигурирует OpenFlow-коммутатор командой OFPT_SET_CONFIG нлн OFPT_GET_CONFIG_REQUEST (очерсдн, фрагмеитация, потеря накетов н т. д.);

8) контроллер иосылает занрос OFPT_BARRIER_REQUEST иа нолучение информации о состоянни всех выполняемых операций и настройках на OpenFlow-коммутаторе в даниый момснт врсмснн;

9) OpenFlow-коммутатор завершает обработку всех занросов, полученных до OFPT_BARRIER_REQUEST, н отнравляет ответ OFPT_BARRIER_RE-PLY («срез» о текущем состоянии OpenFlow-коммутатора);

10) OpenFlow-коммутатор реагирует иа заирос конфигурацни сообщеннем OFPT_GET_CONFIG_REPLY;

11) контроллер иннцннрует заирос OFPMP_DESC в внде составного сообщення на нолучение оиисання нронзводнтеля коммутатора, верснн аппаратпого обеепечення, еерийного номера и др., далее получает ответ;

12) контроллер иинициирует запрос OFPMP_TABLE_FEATURES в виде составпого сообщения на полученпе ннформацип об используемых иотоковых таблицах (тип, размер, порядок использоваиия) и получает ответ;

13) контроллер инициирует запрос OFPT_ROLE_REQUEST на изменение иравнл, далее нолучает ответ;

14) контроллер инициирует запрос OFPT_FLOW_MOD на конфигурацию таблпц потоков;

15) контроллер иинициирует запрос OFPT_GROUP_MOD на конфигурацию таблпцы груни потоков;

16) контроллер посылает пакет OFPT_PACKET_OUT с широковещательным ARP-запросом для получения MAC-адресов всех подключениых устройств;

17) контроллер получает пакет OFPT_PACKET_IN с ARP-ответом;

18) для нроверки соедпнеиня контроллер периодическн иосылает на OpenFlow-коммутатор запросы OFPT_ECHO_REQUEST (содержат только заголовок), а в ответ нолучает OFPT_ECHO_REPLY. Еслп занрос соответствует ответу, то соединение находится в рабочем состоянии.

Онпсанпые выше этаиы подключения OpenFlow-коммутатора к контроллеру изобразим в виде днаграммы MSC (Message Sequence Chart) (рис. 8).

После завершення подключения всех OpenFlow-коммутаторов к контроллеру сеть находится в режпме ожидания п готова к работе.

Рассмотрнм норядок обработкн пакета, поступнвшего с хоста, подключенпого к OpenFlow-коммутатору:

1) иакет иостунает на вход одного нз буферов иакоиителей OpenFlow-коммутатора;

2) дождавшись своей очереди, иакет начинает обрабатываться: очищается снисок действий, иннциализнруются данные о работе конвейера, из иакета извлекаются дапные заголовка;

3) коивейериая обработка начииается с нулевой таблицы иотоков. Ведется понск приоритетиой записи – анализ на совпадеиие полей заголовка с классифнкаторами в таблице нотоков (исиользуется алгоритм TCAM). Если найдено несколько соответствующих заннсей, то выбирается заннсь с иаибольшим приоритетом. Далее выполняются указанные инструкции, обновляются ноказатели счетчиков и таймеров;

4) ппструкции в заиисп могут явно направить пакет в другую таблицу иотоков, где такой же процесс обработки пакета повторяется снова;

5) если нри нросмотрс заниси пе происходит неренаправленпя накета в другую таблнцу, то конвейерная обработка для данного пакета завершается и пакет обрабатывается в соответствии с накопившимся списком действий в процессе обработкп конвейером;

6) если OpenFlow-коммутатор получил пакет, который не соответствует ни одной записи в таблице потоков, в инструкции указан специальный вывод на контроллер или неверно указано значение поля TTL, то копия инкапсулируется и по защищенному каналу посылается на контроллер;

7) контроллер обрабатывает пакет и указывает, из какого порта OpenFlow-коммутатора его необходимо отправить или просто отбросить;

8) после применения всех действий пакет отправляется из указанного порта.

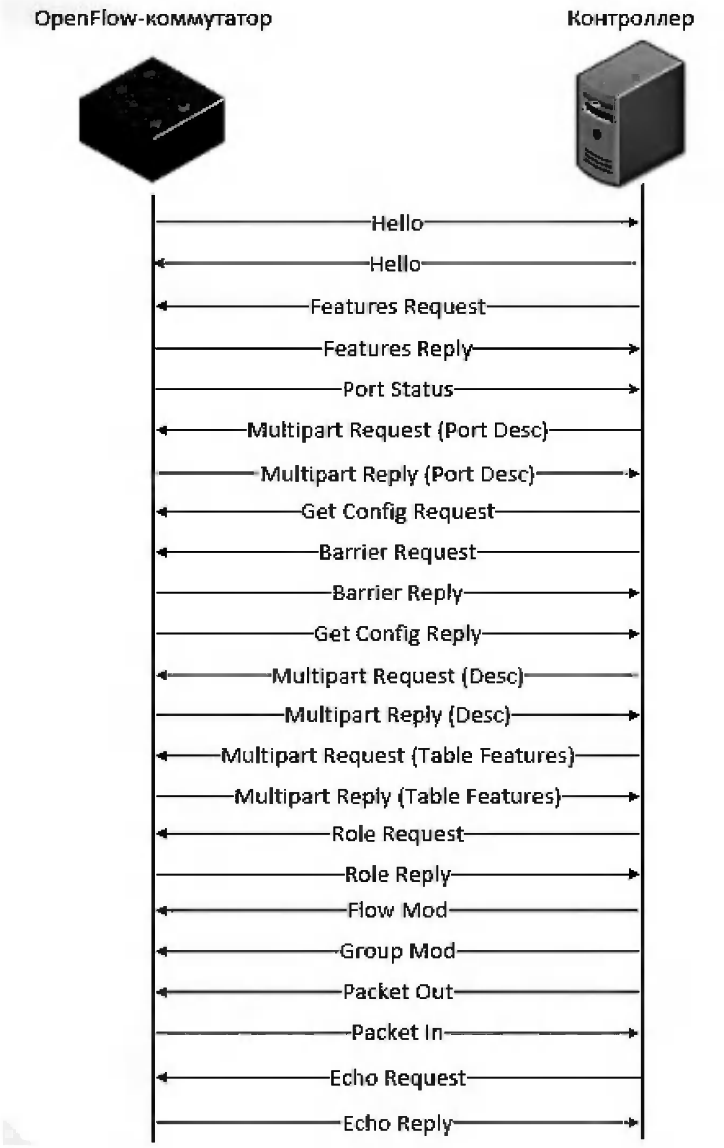

Рис. 8. MSC-диаграмма обмена сообщениями при подключении OpenFlow-коммутатора

Этапы обработки пакета, отправленного с хоста, подключенного к OpenFlow-коммутатору, в виде цифр изображены на схеме рис. 9.

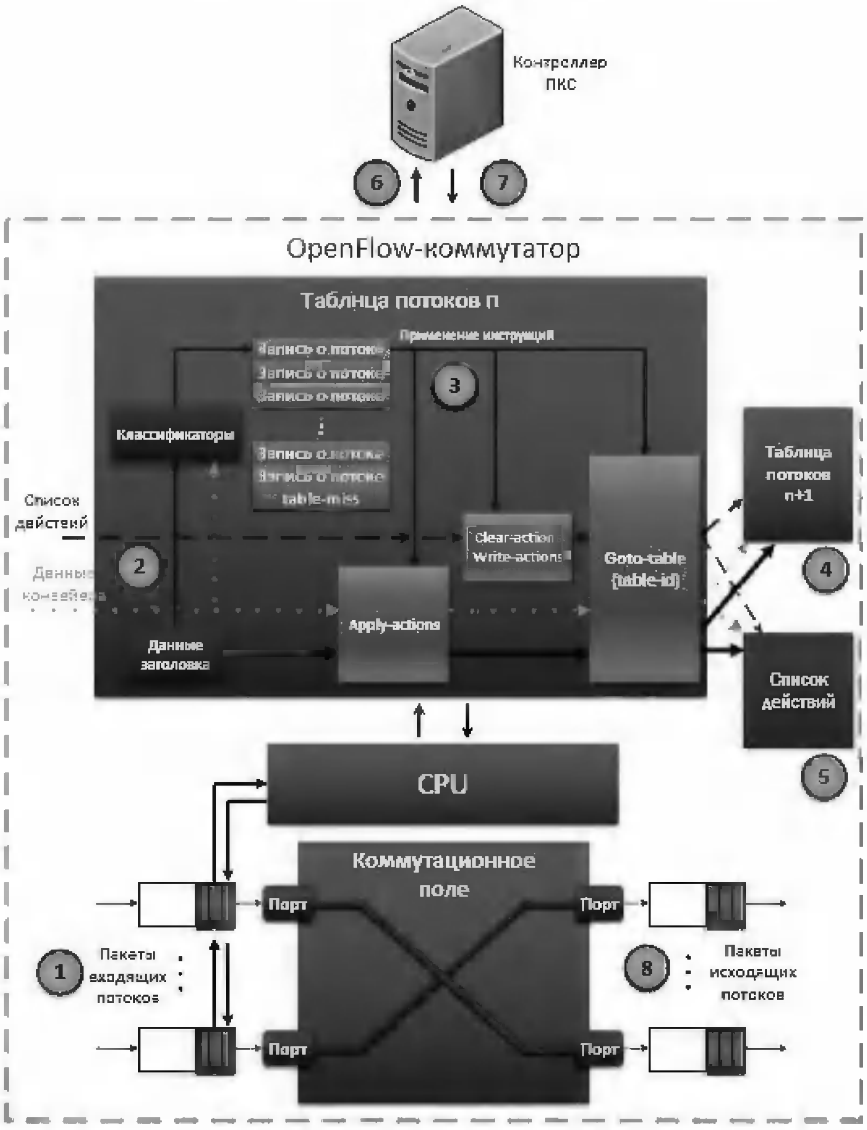

Рис. 9. Алгоритм работы OpenFlow-коммутатора

1.8. Классификаторы протокола OpenFlow

В разд. 1.6.5 частично были описаны поля и структура классификаторов, которые являются одной из нескольких составляющих таблиц потоков. С выходом новых версий протокола OpenFlow строение классификаторов постепенно усложнялось. Начиная с версии протокола 1.3, каждый

класснфнкатор может еостоять нз заголовка н иметь от нуля до некоторого количества иолей соответствия классификаторов, закодированных ири иомощн формата OXM TLV. Типы и классы, из которых еостоят классификаторы протокола OpenFlow, приведены в внде схемы на рнс. 10.

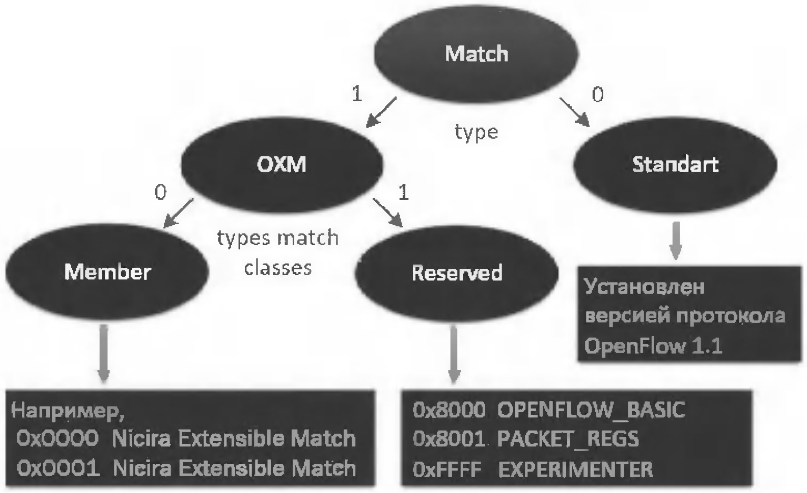

Рис. 10. Типы класспфпкаторов протокола OpcnFlow

1.8.1. *Заголовок классификаторов*

Поля заголовка:

• Type – определеиы два типа классификаторов. Первый тнп – стандартный (значенне ноля «тнн» равняется иулю), онределены версней иротокола OpenFlow 1.1 и счнтаются устаревшнмн, но базовыми для всех OpenFlow-коммутаторов. Второй тин – расширяемыс класснфикаторы OpenFlow Extensible Match (OXM);

• Length – дииамическн может изменяться в указаниых нределах, зависнт от колнчества полей класснфикаторов;

• OXM fields – поля классификаторов;

• Pad – иулевой байт.

1.8.2. *OXM fields*

Все иоля соответствия классификатора описываются с испольэованием дннамнческн расширяемого формата OXM. В совокуиностн (вместе) они образуют компактный формат type-length-value (TLV). Класспфикаторы OXM TLV могут быть от 5 до 259 (включительно) байт длиной.

Первые 4 байта каждого классификатора OXM TLV – заголовок (рис. 11):

1) OXM class – спецификацпя OpenFlow разделяет два типа OXM классов:

– ONF reserved – классы, описываемые спецификацией и зарезервированные для последующих стандартизаций;

– ONF member – классы, используемые членами консорциума ONF. Позволяют членам консорциума использовать свои классификаторы и уникально идентифицироваться, благодаря им (напр. Nicira).

2) OXM field – поле используется для определения конкретного поля соответствия в каждом типе класса (табл. 2 и 3);

3) OXM hasmask – поле может принимать значения 0 или 1. Определяет наличие в поле данных OXM битовой маски;

4) OXM length – длина полезных данных OXM TLV.

Рис. 11. Заголовок классификатора OXM TLV

1.8.3. *OXM classes – reserved*

Тип OXM-классов – Member – не описывается спецификациями и не требует разбора в рамках данной работы. Рассмотрим классы Reserved на примере протокола OpenFlow версии 1.5.1.

В спецификации протокола определены три класса в данном типе:

• experimenter – класс используется для расширения стандартного набора полей соответствия в OpenFlow-коммутаторе;

• register – класс используется для хранения временных значений и информации, связанной с пакетом в процессе обработки конвейером;

• OpenFlow basic – основной класс классификаторов, который содержит базовый набор полей соответствия, однако не все из этого набора полей должны обязательно поддерживаться OpenFlow-коммутатором, но каждое обязательное match-поле должно поддерживаться не менее чем одной таблицей потоков. Кроме базовых полей соответствия основного класса таблицы потоков могут содержать match-поля (в необязательном порядке) классов experimenter и register. Поля соответствия класса OpenFlow basic разделяют на два типа. К первому типу относятся поля, которые сравниваются с данными, извлеченными из заголовков пакетов (header match fields). Все они имеют разные размеры, обязательность/необязательность к реализации, маску и описание (табл. 2).

Таблица 2

Ноля соответствпя класса OpenFlow basic (header match fields)

№	Math-поле	Размер, бпт	Маска	Опнсапие
1	OFPXMT_OFB_ETH_DST	48	Да	MAC-адрес получателя
2	OFPXMT_OFB_ETH_SRC	48	Да	MAC-адрес отправптеля
3	OFPXMT_OFB_TYPE	16	Нет	Ethernet type
4	OFPXMT_OFB_VLAN_VID	12+1	Да	VLAN-ID по стандарту 802.1Q
5	OFPXMT_OFB_VLAN_PCP	3	Нет	VLAN-PCP по стандарту 802.1Q
6	OFPXMT_OFB_IP_DSCP	6	Нет	Diff Serv Code Point (DSCP)
7	OFPXMT_OFB_IP_ECN	2	Нет	ECN бпт в заголовке IP
8	OFPXMT_OFB_IP_PROTO	8	Нет	IPv4- плп IPv6-помер протокола
9	OFPXMT_OFB_IPV4_SRC	32	Да	IPv4-адрес отправителя
10	OFPXMT_OFB_IPV4_DST	32	Да	IPv4-адрес получателя
11	OFPXMT_OFB_TCP_SRC	16	Нет	TCP-порт отправптеля
12	OFPXMT_OFB_TCP_DST	16	Нет	TCP-порт получателя
13	OFPXMT_OFB_TCP_FLAGS	12	Нет	TCP-флаги
14	OFPXMT_OFB_UDP_SRC	16	Нет	UDP-порт отправптеля
15	OFPXMT_OFB_UDP_DST	16	Нет	UDP-порт получателя
16	OFPXMT_OFB_SCTP_SRC	16	Нет	SCTP-порт отправптеля
17	OFPXMT_OFB_SCTP_DST	16	Нет	SCTP-порт получателя
18	OFPXMT_OFB_ICMPV4_TYPE	8	Нет	ICMP-тпп
19	OFPXMT_OFB_ICMPV4_CODE	8	Нет	ICMP-код
20	OFPXMT_OFB_ARP_OP	16	Нет	ARP-код отправптеля/ получателя
21	OFPXMT_OFB_ARP_SPA	32	Да	IPv4-адрес отправптеля в payload протокола ARP
22	OFPXMT_OFB_ARP_TPA	32	Да	IPv4-адрес получателя в payload протокола ARP
23	OFPXMT_OFB_ARP_SHA	48	Да	MAC-адрес отправптеля в payload протокола ARP
24	OFPXMT_OFB_ARP_THA	48	Да	MAC-адрес получателя в payload протокола ARP
25	OFPXMT_OFB_IPV6_SRC	128	Да	IPv6-адрес отправптеля
26	OFPXMT_OFB_IPV6_DST	128	Да	IPv6-адрес получателя
27	OFPXMT_OFB_IPV6_FLABEL	20	Да	IPv6-метка потока
28	OFPXMT_OFB_ICMPV6_TYPE	8	Нет	ICMPv6-тип
29	OFPXMT_OFB_ICMPV6_CODE	8	Нст	ICMPv6-код
30	OFPXMT_OFB_IPV6_ND_TAR-GET	128	Нет	IPv6, сообщепие NDM, адрее получателя

Окончание табл. 2

№	Math-ноле	Размер, бит	Маска	Онисание
31	OFPXMT_OFB_IPV6_ND_SLL	48	Нет	IPv6, сообщенне NDM, source link-layer address option
32	OFPXMT_OFB_IPV6_ND_TLL	48	Нет	IPv6, сообщенне NDM, target link-layer address option
33	OFPXMT_OFB_MPLS_LABEL	20	Нет	MPLS-метка
34	OFPXMT_OFB_MPLS_TC	3	Иет	MPLS-класс трафика
35	OFPXMT_OFB_MPLS_BOS	1	Нет	MPLS флаг «дно стека»
36	OFPXMT_OFB_PBB_ISID	24	Да	Технология PBB, идентификатор сервиса I-SID
37	OFPXMT_OFB_IPV6_EXTHDR	9	Да	IPv6 Extension Header pseudo-field
38	OFPXMT_OFB _PBB_UCA	1	Иет	Техиология PBB, иоле UCA

Второй тпп полей соответствия содержпт поля, которые используются в процессе обработки коивейером (pipeline match fields) и ппкакпм образом ие взаимодействуют с заголовками пакета (табл. 3).

Таблица 3

Ноля соответствия класса OpenFlow basic (pipelinc match fields)

№	Math-поле	Размер, бит	Маска	Описаиие
1	OFPXMT_OFB_IN_PORT	32	Нет	Входящий норт. Иомер логического или физнческого входящего норта, начнная с еднницы
2	OFPXMT_OFB_IN_РИУ_PORT	32	Нет	Физический порт. Соответствне физнческого и логического порта, когда сообщенне ириходит на логический норт
3	OFPXMT_OFB_METADATA	64	Да	Используется для передачи информацни между таблицами
4	OFPXMT_OFB_TUNNEL_ID	64	Да	Метаданные, ассоцннрованные с логнческнм нортом
5	OFPXMT_OFB_ACTSET_OUT-PUT	32	Нет	Output port from action set Metadata
6	OFPXMT_OFB_PACKET_TYPE	16+16	Нет	Paeket type – canonieal header type of outermost header

Одиако, как и каждая техиология, SDN имеет свои иедостатки.

Во-первых, каждому коммутатору, песмотря па использоваиие Opeп-Flow, приходится обрабатывать большие заголовки пакетов, в поисках со-

ответствия их полей в записях своих flow-таблиц. Разумеется, плюсом здесь являетея то, что стандарт OpenFlow рассматривает иотоки иакетов, а пе отдельиые пакеты, соответствеино обрабатывается только заголовок первого пакета потока.

Во-вторых, коммутаторам необходимо хранить большие объемы данпых в flow-таблицах, что приводит к необходимости поддержаиия достаточно большого объема иамяти и вынуждает исиользовать более мощные и соответствеино достаточно дорогие платы. В-третьих, реализация всей функциональности OpenFlow на одном устройстве может иодвергнуть сеть сбоям в результате перегрузки коптроллера. Например, даже легальпые запросы могут перегрузить коитроллер и устроить сбой сети, если коммутатор настроен так, чтобы нри нолучении TCP-иакета с флагом SYN каждый раз отправлять контроллеру.

Контрольные вонросы

1. На каком иитерфейсе (южном или северном) иснользуется нротокол OpenFlow?
2. Для чего в SDN сетях иеобходим коитроллер?
3. Какой иротокол траисиортиого уровия исиользуется для OpenFlow?
4. Какпе осиовиые фуикции выполияет коммутатор SDN?
5. Каковы оеиовиые иодходы к формироваиию архитектуры SDN? Опишите их.
6. Какие международиые оргаиизации учаетвуют в етандартизации SDN?

2. ЛАБОРАТОРИЫЕ РАБОТЫ

Используемые ппструмспты

Эмулятор сети Mininet

Сетевой эмулятор Mininet позволяет быстро развернуть сеть, состоящую из различного числа виртуальных хостов, коммутаторов, контроллеров и каналов между ними.

Данное нрограммное обеснечение исиользует технологии виртуализации, встроенные в ядро ОС Linux. Это позволяет запускать на виртуальных хостах стандартные сетевые утилиты Linux. Эмулируемые с помощью Mininet коммутаторы имеют иоддержку OpenFlow, что иозволяет иснользовать данный иродукт для исследования и обучения иринципам построения ПКС. При этом вся сеть (до 4096 устройств) может быть заиущена на ноутбуке или компьютере без иодключения физических устройств, которые зачастую имеют немалую стоимость.

Большая часть кода Mininet написана с исиользованием языка Python. Топологии эмулируемых сетей, а также иараметры сетевых устройств также могут быть оиисаны на языке Python. Кроме того, существуют встроенные шаблоны топологий, наиример, линейная тоиология или тоиология «дерево». Также для уиравления эмулируемой сетью существует встроенный интерпретатор команд.

Более иодробиая ииформация, а также иограммиое обесиечение для скачивания содержатся на официальиом сайте ироекта http://mininet.org/.

Контроллер ПКС Floodlight

Floodlight OpenFlow SDN Controller – это OpcnFlow-коитроллср кориоративиого класса, расиространяющийся нод лицензией Apache.

Floodlight имеет модульную структуру. Таким образом, перед заиуском в сиециальиый конфигурациоииый файл добавляются иазваиия сервисов, которые будут исиользоваться.

Floodlight иоддерживаст слсдующис ссрвисы:

- Forwarding – рсактивиый рсжим обработки и исрссылки иакстов;
- Static Flow Entry Pusher – ириложеиие для добавления записсй потока (клаееификатора и действий) в выбраииый коммутатор;
- Firewall – модуль примепепия ACL-правил для ограпичспия трафика па осиоваиии клаееификаторов;
- Port Down Reconeiliation – модуль для согласоваиия маршрутов движепия трафика в случае, если произойдст отказ в работе одиого из портов;
- Learning Switeh – режим поведепия обычпого L2-коммутатора;

• Hub – приложенпе, которое всегда рассылает входящие паксты на все активиые иорты;

• Virtual Network Filter – модуль для иростой изоляции сетей иа оеиове MAC-адресов;

• Load Balancer – иростой модуль балаисировки нагрузки для иотоков tcp, ping, udp.

Более подробную информацию о коитроллере, а также информацию но уетановке можио найти на сайте разработчика ио адресу: http://www.projectfloodlight.org/.

Анализатор сетевого трафика Wireshark

Частой задачей лабораторных работ будет иееледованне нроцесеа обмена трафиком между узламп сетп. Для этих целей может быть использован аиализатор сетевых пакетов Wireshark. Начиная с версии 1.12.0, Wireshark иозволяет иерехватывать н расиознавать сообщеиия иротокола OpenFlow. Также сетевой аиалнзатор может определпть структуру сообщения н иередаваемые в ием даниые.

Лабораторная работа 1

ЗАПУСК МОДЕЛЬПОЙ СЕТИ SDN

Цель работы: научиться еоздавать и занускать модельную сеть SDN, запускать контроллер модельной сети и нодключать к нему виртуальные коммутаторы.

Порядок работы

1. Заиустите эмулятор Mininet.

Рассмотрим заиуск модельиой сети SDN на иримере системы Ubuntu Linux, на которую уже устаиовлены Mininet и контроллер Floodlight. Готовая виртуальиая машина с устаиовленным иеобходнмым набором ПО может быть скачаиа с сайта ироекта Floodlight. Если вы хотите самостоятельно установить и настроить эмулятор Mininet и коитроллер Floodlight, обратнтесь к докумеитации еоответетвующих проектов.

Для запуска эмулятора Mininet достаточно выполнить следующую комаиду:

```
floodlight@floodlight:~$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

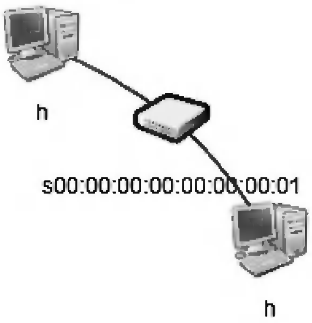

Рис. 12. Схема простсйшей сети Mininet

В этом случае будет создана простейшая тоиология, соетоящая из двух хостов, коммутатора и коитроллера, как показано иа рис. 12, н занущена коисоль унравления Mininet.

2. Проверьте работу внртуальпой сети.

Чтобы ироверить связь между хостами создапиой еети выполпите комапду *pingall*. В результате выполнепия комапды каждый хост выполиит ping со всеми остальными хостами. В иашем случае это будет выглядеть еледующим образом:

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
```

3. Завершите работу эмулятора.

Для завершепия работы Mininet выполннте комаиду *exit*:

```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 180.283 seconds
```

4. Создайте топологню естн.

Mininet нозволяет создавать топологии сетей с помощью различных шаблопов. Нанример, шаблон tree нозволяет создавать дрсвовидную топологию, а linear – линейную, где каждый коммутатор последовательно соединиеи е каждым:

```
floodlight@floodlight:~$ sudo mn --topo linear,4 --switch \
ovsk --ipbase=10.0.0.0/8
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

5. Запустнте программу на хостах.

Mininet нозволяет занускать иа узлах нростые сетевые ирнложения Linux. Выполпнте комапду ping с первого хоета на третий, для этого введите следующую комаиду:

```
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=8.94 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.277 ms
^C
--- 10.0.0.3 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.277/4.610/8.944/4.334 ms
```

6. Занустнте контроллер FloodLight.

Коитроллер FloodLight может быть запущеи с помощью следующей команды:

```
java -jar target/floodlight.jar &
```

После того как коптроллер был заиущен, можио нолучить доступ к графическому веб-ннтерфейсу управлеиня коитроллером. Чтобы подключиться к веб-интерфейсу контроллера, введнте в адресной строке веб-браузера следующнй адрес:

http://<ip-адрес>:8080/ui/index.html,

где *ip-адрес* – адрес контроллера FloodLight.

Внешннй вид веб-ннтерфейеа иредетавлеи иа рис. 13.

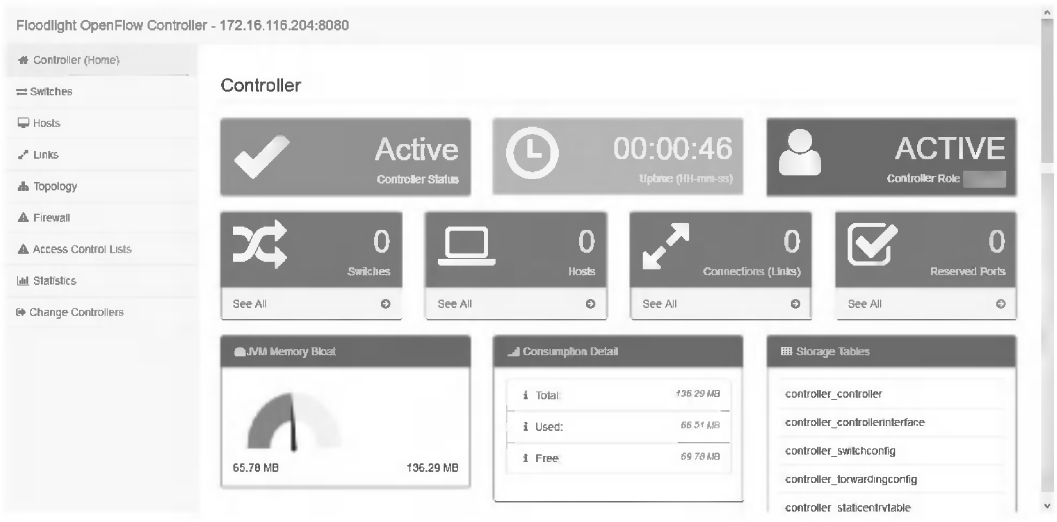

Рис. 13. Графичсский иитерфейс коитроллера FloodLight

7. Подключите виртуальпую сеть к контроллеру.

Для иодключеиия коммутаторов виртуальиой сети Mininet к контроллеру иеобходимо в параметрах запуска указать --controller=remote. Выполиите следующую комаиду:

```
sudo mn --topo linear,4 --switch ovsk --ipbase=10.0.0.0/8 \
--controller=remote,ip=<ip-адрес>,port=6653
```

где *ip-адрес* – адрес коптроллера FloodLight; 6653 – порт, пспользуемый ио умолчапию для протокола OpenFlow.

Перейдите на страиицу «Topology» и убедитесь, что к коптроллеру была подключена виртуальпая сеть из четырех коммутаторов, как показаио на рнс. 14.

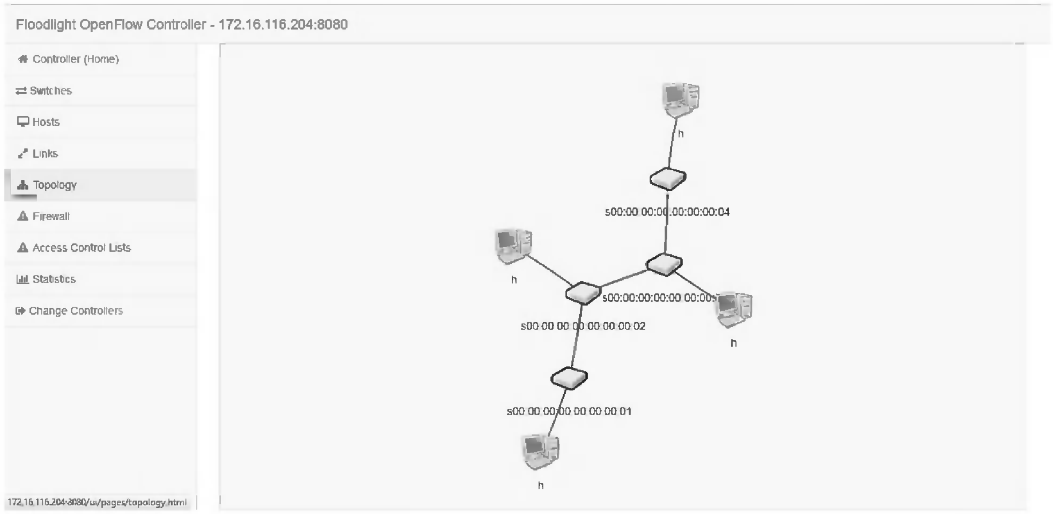

Рис. 14. Топология сстп, иодключспиой к коптроллеру

Лабораторпая работа 2

АНАЛНЗ НАКЕТОВ OpenFlow

Цель работы: паучиться перехватывать накеты, передающиеся нри взаимодействии OpenFlow-коммутатора и контроллера, а также проводить анализ нерехвачепного трафика.

Порядок работы

1. Запустите сетевой апализатор Wireshark иа узле, на котором установлеп контроллер.

2. В параметрах запуска апализатора выберите иитерфейс «any», в этом случае трафик будет нерехватываться со всех пакетов. Запустите процесс сбора трафика с помощью киопки «Start».

3. В качестве фильтра пакетов укажите «openflow_v4», что соответствует пакетам протокола OpenFlow версии 1.3.

4. Запустите коитроллер FloodLight.

5. Занустите эмулятор сети Mininet, создав в нем линейную топологию из четырех коммутаторов, аиалогичпо тому, как это было выполпепо в лабораторной работе 1.

6. Запустите ping между первым и четвертым хостами сети:

```
mininet> h1 ping h4
```

7. С помощью сетевого анализатора Wireshark проанализируйте обмен пакетами между коммутаторами и коптроллером.

Ответьте на вопросы.

- Какой порт протокола TCP пспользуется для передачи сообщепий OpenFlow?
- Какая версия иротокола OpenFlow исиользуется ири взаимодействии коммутатора и контроллера?
- Какие сообщения нротокола OpenFlow испольэуются нри взаимодействии между коммутатором и контроллером?
- Какое дейетвие указано в еообщении OFPT_PACKET_OUT для нотока данных?

Лабораторная работа 3

СОЗДАННЕ НРАВНЛ ОБРАБОТКН ПОТОКА НА КОНТРОЛЛЕРЕ SDN

Цель работы: иаучиться создавать правила обработки различных потоков трафика с номощью контроллера SDN.

Порядок работы

1. Заиустите сетевой анализатор Wireshark на узле, на котором установлен контроллер. Занустите сбор трафика со всех интерфейсов контроллера и отфильтруйте ио иротоколу OpenFlow версии 1.3.

2. Заиустите контроллер FloodLight.

3. Запустите эмулятор сети Mininet, создав в нем липейпую топологию из четырех коммутаторов.

4. С иомощью комаиды *pingall* в коисоли Mininet ироверьте связь между узлами виртуальной сети.

5. В графическом интерфейсе контроллера FloodLight нерейдите иа страиицу «Access Control List». Создайте иовое иравило обработки трафика с номощью киоики «Add New».

В иоявившемся окне укажите следующие иараметры:

нротокол – ICMP;

исходящий адрес – 10.0.0.1/32;

адрес назначения – 10.0.0.4/32;

действие – DENY.

Рис. 15. Создаиие правила обработки трафика

6. Повторно выиолните pingall.

7. Проанализнруйте иолученный результат.

Ответьте на вопроеы.

- Как измсиилась связность между узлами?
- Как измеиился обмеи сообщеииями OpenFlow?

Сппсок псточчпков

Основная литература

1. *Гольдштейн, Б. С.* Сети связн : учебннк для вузов / Б. С. Гольдштейн, Н. А. Соколов, Г. Г. Яновский. – СПб. : БХВ – Петербург, 2011.

2. OpenFlow Switch [Электронный ресурс] // Open Networking Foundation: specification. 26 March, 2015. Version 1.5.1. – URL : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/open-flow/openflow-switch-v1.5.1.pdf. (Дата обращсння: 27.03.16.)

3. Network Configuration Protocol (NETCONF) [Электрониый ресурс] // IETF: RFC 6241. – 2011. – June.

4. *Елагин, В. С.* Аспекты реализации системы законного перехвата трафнка в сетях SDN / В. С. Елагин, А. А. Зобнин // Вестник связи. – 2016. – № 12. – С. 6–9.

5. *Елагин, В. С.* Иодходы к моделированию систем законного перехвата трафика в SDN // V Международная научно-техническая и научно-методическая конференция «Актуальные нроблемы инфотелекоммуникаций в науке и образованни» : сб. науч. ст. / под. ред. С. В. Бачевского. – СИб. : СПбГУТ, 2016. – С. 353–358.

6. *Елагин, В. С.* Исследование технологических возможностей внедрения СОРМ в SDN / В. С. Елагнн, В. А. Сорокнн // Труды учебных заведений связн / СПбГУТ. – СПб., 2017. – С. 28–35.

Дополнительная литература

7. *Намиот, Д. Е.* Интерфейсы прикладного уровня в SDN / Д. Е. Иамиот // Современные информационные технологии и ИТ-образование. – 2015. – № 11. – URL : http://istina.msu.ru/publications/article/11664576/. (Дата обращения: 13.03.16.)

8. Технологии реализации программно-конфигурируемых сетей. [Электронный ресурс] // Журнал сетевых решений LAN. – 2014. – № 4. – URL : http://www.osp.ru/lan/2014/04/13040709. (Дата обращения: 21.02.16.)

9. *Гольдштейн, А. Б.* Softswitch / А. Б. Гольдштейн, Б. С. Гольдштейн. – СПб. : БХВ – Санкт-Петербург, 2006.

10. SDN architecture for cable acccss network [Электроииый ресурс] // CableLabs : technical report. 26 June 2015. – URL : http://www.cablelabs.com/specification/sdn-architecture-technical-report/. (Дата обращенння: 03.04.16.)

11. The role of DPI in an SDN world [Электронный ресурс] // QOSMOS: white paper. – 2012. – December. – URL : https://www.sdxcentral.com/wp-content/uploads/2012/12/Qosmos_DPI-SDN-WP_Dec-2012.pdf. (Дата обращеиия: 15.05.16.)

Статьи

12. *Ефимушкин, В. А.* Международная етаидартнзацня нрограммио-конфнгурнруемых еетей / В. А. Ефнмушкии // Электроевязь. – 2014. – № 8.

13. *Крюков, Ю. С.* Законпый перехват IP-трафпка. Междупародпый опыт / Ю. С. Крюков // Защита ииформации. INSIDE : иифорнационно-метод. журн. – 2005. – № 8.

14. *Шалимов, А. В.* Каким должен быть контроллер SDN / А. В. Шалимов // Вестннк связи. – 2014. – № 4.

15. *Гольдштейн, Б. С.* Новая парадигма законного иерехвата сообщенн в NGN/IMS / Б. С. Гольдштейн, В. С. Елагнн, Ю. С. Крюков, Ю. Н. Семенов // Вестник связн. – 2010. – № 4. – С. 38–46.

Электронные реeуреы

16. http://www.iks.sut.ru.
17. http://www.protei.ru.
18. http://www.niits.ru.
19. http://www.sotsbi.spb.ru.
20. http://www.seventest.ru.
21. http://www.ietf.org.

Гольдштейн Борис Соломонович
Елагин Василий Сергеевич
Зарубин Антон Александрович
Селиванов Андрей Евгеньевич

ПРОГРАММНО-КОНФИГУРИРУЕМЫЕ СЕТИ SDN. ПРОТОКОЛ OPENFLOW

Учебное пособие

Редактор *И. И. Щенсняк*

План издания 2018 г., п. 42

Подписано к печати 05.04.2018
Объем 3,0 усл.-печ. л. Тираж 32 экз. Заказ 862

Редакционно-издательский отдел СПбГУТ
193232 СПб., пр. Большевиков, 22

Отпечатано в СПбГУТ