УДК 004.725
ББК 32.973.202
П 16

Панеш А.Х.
Кандидат технических наук, доцент кафедры прикладной математики, информационных технологий и информационной безопасности факультета математики и компьютерных наук Адыгейского государственного университета, Майкоп, тел. (8772) 593904, e-mail: apanesh@yandex.ru

Достоинства и недостатки программно-конфигурируемых компьютерных сетей

(Рецензирована)

Аннотация. *Рассмотрены в обобщенном виде три направления реализации программно-конфигурируемых сетей (ПКС): ПКС на основе Open SDN, ПКС на основе существующих API и ПКС на основе наложенных сетей и гипервизоров. В первом случае используется протокол OpenFlow, во втором – RESTful API для управления сетью, в третьем – гипервизоры виртуальных машин в центрах обработки данных. Показано, что выбор того или иного ПКС при построении компьютерной сети зависит от конкретных требований к сети с учетом достоинств и недостатков рассмотренных вариантов.*

Ключевые слова: *программно-конфигурируемые сети, протокол OpenFlow, интерфейс программирования приложений API, центры обработки данных.*

Panesh A.Kh.
Candidate of Technical Sciences, Associate Professor of the Department of Applied Mathematics, Information Technology and Information Security, Faculty of Mathematics and Computer Science, Adyghe State University, Maikop, ph. (8772) 593904, e-mail: apanesh@yandex.ru

The advantages and disadvantages of software-defined computer networks

Abstract. *The paper summarizes three directions of the implementation of Software Defined Networking (SDN): SDN based on Open SDN, SDN based on existing APIs and SDN via Hypervisor-based overlay networks. In the first case, the OpenFlow protocol is used, in the second case RESTful API for network management is used and in the third case hypervisors of virtual machines in data center are used. The paper shows that the choice of a SDN when building a computer network depends on the particular network requirements, taking into account the advantages and disadvantages of options considered.*

Keywords: *software defined networks, OpenFlow protocol, application programming interface API, data centers.*

Введение

Технологии программно-конфигурируемых сетей активно развиваются во всем мире. Понятие ПКС – это общая концепция; конкретные реализации ПКС могут значительно отличаться по своей архитектуре. Можно выделить три направления реализаций ПКС:

- ПКС, создаваемые на основе первоначальной, оригинальной версии, предложенной специалистами стэндфордского университета (США) в 2007 г.;
- ПКС, создаваемые на основе существующих API;
- ПКС, создаваемые на основе наложенных сетей и гипервизоров.

Первое направление, то есть реализация ПКС в оригинальной версии, предполагает перемещение функционала управления сетью от сетевых устройств к централизованному контроллеру с использованием протокола OpenFlow [1]. Такая ПКС должна обладать, как считали ее создатели-пионеры, следующими пятью фундаментальными свойствами: иметь разделенные уровни управления, использовать более простые сетевые устройства, осуществлять централизованное управление сетью, использовать автоматизацию и виртуализацию сетевых функций, быть открытой для специалистов, разработчиков. Появился специальный термин – «Open SDN» для сетей, имеющих перечисленные свойства [2, с. 56]. Обязательным элементом такой ПКС является наличие контроллера, работающего с сетью через интерфейс OpenFlow.

Во втором направлении при создании ПКС используются функции API, которые могут быть вызваны удаленно, как правило, с помощью традиционных механизмов, таких как SNMP, CLI или при использовании более новых, гибких механизмов, типа RESTful API.

Реализация ПКС в третьем случае не зависит от нижележащей сетевой инфраструктуры: ПКС накладывается поверх существующей физической сети. И второй, и третий вариан-

ты создания ПКС являются альтернативными вариантами относительно Open SDN, но в некоторых случаях являются более предпочтительными при создании компьютерных сетей.

Рассмотрим целесообразность выбора того или иного варианта ПКС при создании компьютерной сети.

ПКС, создаваемые на основе Open SDN

Изначально появление концепции Open SDN в первоначальной, оригинальной трактовке было встречено с большим энтузиазмом. И действительно, Open SDN-сети потенциально обладают многими достоинствами. Благодаря снятию с коммутаторов задач управления продвижением данных происходит ускорение перемещения трафика, что существенно повышает производительность. При этом за счет виртуализации управления сетью снижаются расходы на их построение и сопровождение. На централизованном контроллере Open SDN системный администратор может наблюдать всю сеть в едином представлении, за счет чего повышаются удобство управления, безопасность и упрощается выполнение ряда других задач. Теоретически неограниченные возможности сетей Open 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 [3].

Но вскоре появилось немало критики, связанной и с архитектурой Open SDN, и с вопросами практического внедрения таких сетей. Наиболее громкими критиками являются производители сетевого оборудования, которые обеспокоены слишком радикальными, на их взгляд, изменениями, которые несет с собой новая сетевая технология. Это и значительная стоимость нового оборудования, то есть коммутаторов с поддержкой интерфейса OpenFlow, и риски, возникающие из-за недостаточного тестирования нового оборудования, которое должно в массовом порядке внедряться у клиентов. Также увеличатся расходы на переобучение многочисленных IT специалистов, которые будут работать с новыми сетями.

Наиболее серьезным аргументом критиков Open SDN с точки зрения ее архитектуры является уязвимость такой сети из-за наличия в ней единой точки отказа. На рисунке 1 показана схема продвижения данных в сети Open SDN, управляемой единственным контроллером. При выходе из строя этого контроллера происходит потеря работоспособности всей сети. То есть изображенный контроллер ПКС является единой точкой отказа.

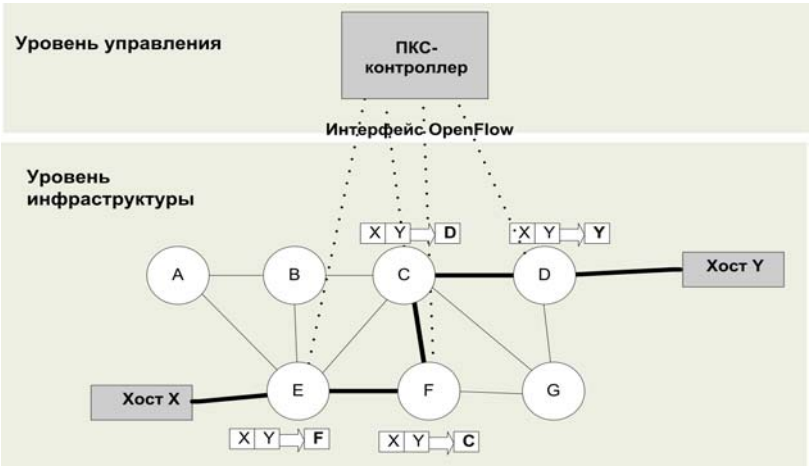

Рис. 1. Продвижение данных в сети, соответствующей архитектуре Open SDN

Чтобы преодолеть указанный недостаток, необходимо иметь в сети несколько контроллеров, взаимодействующих между собой с помощью особо надежных линий связи. Это усложняет и удорожает архитектуру Open SDN.

ПКС, создаваемые на основе существующих API

Если снабдить сетевые устройства возможностью распознавания более широкого набора команд API, с помощью которых контроллер сможет гибко управлять устройствами и всей сетью, то это и будет ПКС, реализованная через существующие API (рис. 2).

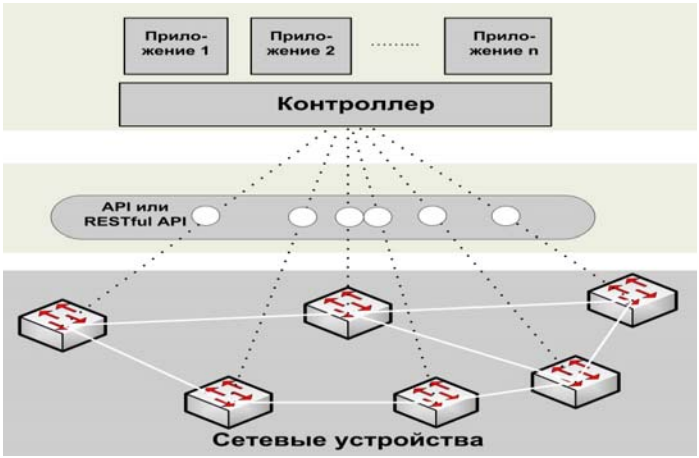

Рис. 2. ПКС, созданная на основе существующих API

Для создания ПКС по такой схеме некоторые производители оборудования модернизируют существующие API на устройствах. Например, вместо традиционных CLI и SNMP внедряется RESTful API [4]. Механизмы CLI и SNMP давно разработаны и используются при выполнении сетевых настроек. Но в настоящее время, когда необходимо оперативно, динамически управлять большой сетью или центром обработки данных, эти механизмы являются слишком громоздкими и неудобными. Им на смену пришел новый механизм – RESTful API. В последние годы этот механизм стал наиболее распространенным при передаче API-запросов по сети. Технология RESTfull API работает с использованием протокола передачи гипертекста HTTP, который в общем случае используется для передачи WEB-трафика. Технология RESTful API является относительно простой и легко расширяемой. Она удобна в применении, так как использует в работе стандартный TCP-порт, что не требует специальных настроек межсетевого экрана для прохождения через него API-запросов. У ПКС, созданных на основе существующих API, имеется ряд преимуществ. Одним из очевидных преимуществ является то, что они работают с обычными, не модернизированными коммутаторами. То есть не требуется внедрение коммутаторов нового типа, с поддержкой стандарта OpenFlow.

Еще одно преимущество такого подхода заключается в том, что в определенной степени повышается гибкость управления сетью. Имеющиеся API упрощают написание программного обеспечения для оркестровки событий в сети, то есть для быстрого и автоматического реагирования на изменения в сети, таких, как динамические перемещения виртуальных машин в центрах обработки данных. Следующее преимущество заключается в том, что использование имеющихся API позволяет построить сеть с централизованным в определенных пределах управлением. Это ведет к большей открытости в сетевых архитектурах, так как производители вынуждены открывать спецификации интерфейсов своего фирменного оборудования. Последнее необходимо для разработки и нормальной эксплуатации приложений сторонними разработчиками.

ПКС, создаваемые на основе существующих API, имеют и свои недостатки. Прежде всего, в таких сетях в большинстве случаев контроллер вовсе отсутствует. Поэтому сетевой программист вынужден программировать напрямую каждый коммутатор. Но если даже имеется контроллер, у программиста отсутствует общий, стандартный механизм взаимодействия с устройствами сети. То есть программист вынужден знать технические характеристики интерфейса каждого коммутатора. Еще одним недостатком является то, что программное обеспечение такой ПКС будет работоспособно только для конкретной конфигурации сети. Это и понятно, так как API устройств разных производителей не подпадают под общий стандарт (в противоположность протоколу OpenFlow). Поэтому ПКС такого типа сможет работать с оборудованием определенного производителя или небольшой группы производителей совместимого оборудования.

Следующий недостаток связан с тем, что предписываемое в архитектуре ПКС перемещение функций управления от коммутатора к контроллеру имело целью создание простых, менее дорогих коммутаторов. Эту задачу решить не удалось: ПКС создаются на основе тех

же сложных и дорогих коммутаторов. Наконец, хотя ПКС, созданные на основе существующих API, и позволяют в определенной мере управлять пересылкой данных, в частности, при построении VLAN и VPN, но они не могут обеспечить более тонкий контроль каждого информационного потока, как это возможно при использовании протокола OpenFlow.

ПКС, созданные на основе существующих API, можно рассматривать в качестве практического расширения текущей функциональности компьютерной сети, когда более радикальное решение на основе OpenFlow еще не доступно или нецелесообразно по каким-либо причинам. В целом, создание ПКС на основе существующих API является шагом в правильном направлении, движением в направлении создания сети с полностью централизованным программным управлением.

ПКС, создаваемые на основе наложенных сетей и гипервизоров

Альтернативным и более инновационным способом создания ПКС является использование наложенных сетей с управлением через гипервизоры [5]. В том случае физическая сеть остается как есть, без изменений. Но над этой физической сетью создаются наложенные виртуальные сети с гипервизорами. Прикладные системы на узлах сети взаимодействуют с этими виртуальными сетями, ничего не зная о физических характеристиках сети, через которую происходит продвижение данных (рис. 3).

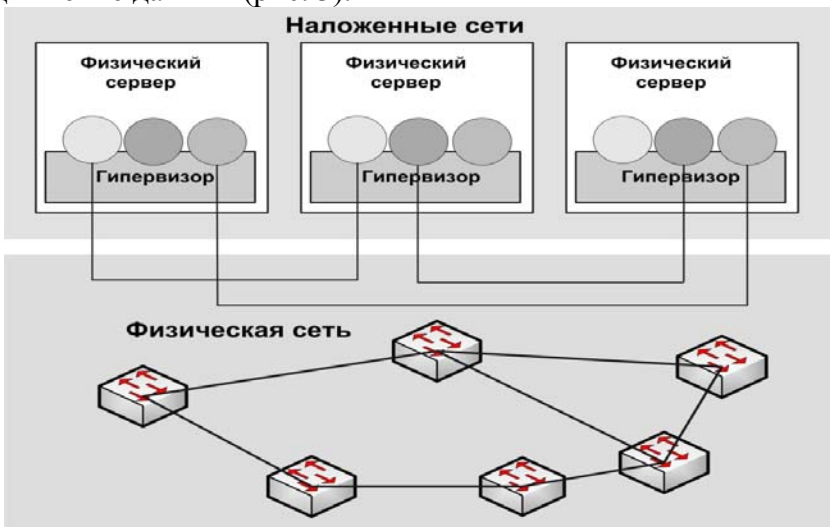

Рис. 3. Виртуальные сети, наложенные на физическую сеть

Поскольку виртуальные сети находятся над физической инфраструктурой, ими могут управлять системы (или устройства), размещенные на конечных узлах сети. В центрах обработки данных (ЦОД) такими системами являются гипервизоры виртуальных машин, которые присутствуют на каждом сервере. Передача трафика в виртуальных сетях осуществляется посредством туннелирования, с использованием инкапсуляции. То есть при поступлении пакета на узел виртуальной сети для передачи сетевое устройство (обычно это гипервизор) инкапсулирует этот пакет в другой кадр (рис. 4). ПКС, созданные на основе наложенных сетей и гипервизоров, хорошо работают в ЦОД, в которых уже установлено программное обеспечение для серверной виртуализации. Они помогают устранить ряд проблем, возникающих при работе ЦОД.

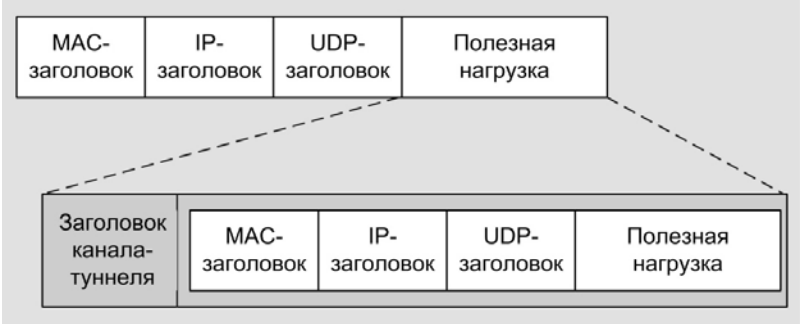

Рис. 4. Инкапсулирование пакета данных в кадр канала-туннеля

Во-первых, устраняется взрывной рост MAC-адресов узлов, так как теперь, при использовании ПКС, MAC-адреса скрыты в инкапсулированных кадрах. Во-вторых, известные ограничения на количество поддерживаемых VLAN в локальных сетях уже не имеют значения, так как здесь используется туннелирование (вместо VLAN) для разделения трафика многочисленных информационных потоков. В-третьих, построение ЦОД на основе ПКС позволяет очень гибко и оперативно менять характеристики сетей, участвующих в вычислительных процессах ЦОД, за счет возможности централизованного программного управления виртуальными сетями ПКС.

ПКС, созданные на основе наложенных сетей и гипервизоров, не решают все проблемы. В частности, физическая сетевая инфраструктура все так же требует ручной настройки и обслуживания. Это касается, например, протоколов QoS, STP. Другим недостатком является то, что в таких ПКС, как и в предыдущем варианте, сетевые устройства остаются неизменными, не модернизируются, не упрощаются.

Заключение

Рассмотренные архитектуры ПКС содержат определенные достоинства и недостатки. Выбор того или иного варианта зависит от конкретных требований к сети, условий ее эксплуатации. Архитектура Open SDN является наиболее многообещающей, открытой. Поэтому для компаний-производителей сетевого оборудования поддержка протокола OpenFlow становится «общим знаменателем», который обеспечит совместимость сетевых устройств от разных производителей.

Устройства с поддержкой OpenFlow уже разработаны и выпускаются во многих странах. В России тема ПКС также не оставлена без внимания. Работа в этом направлении входит в список приоритетных научных задач, сформулированных правительством и утвержденных президентом Российской Федерации [6].

Сокращения*:*

ПКС – Программно-конфигурируемые сети
ЦОД – Центр обработки данных
SDN – Software Defined Networks
API – Application Programmer Interface
CLI – Command-Line Interface
VPN – Virtual Private Network
STP – Spanning Tree Protocol
SNMP – Simple Network Management Protocol
REST – Representational State Transfer
WEB – Всемирная паутина
TCP – Transmission Control Protocol
VLAN – Virtual Local Area Network
QoS – Quality of Service

Примечания:

1. Панеш А.Х. Содержание и перспективы технологий программно-конфигурируемых сетей и виртуализации сетевых функций // Вестник Адыгейского государственного университета. Сер. Естественно-математические и технические науки. 2014. Вып. 2 (137). С. 120–127. URL: http://vestnik.adygnet.ru
2. Göransson P., Black Ch. Software Defined Networks: A Comprehensive Approach. Morgan Kaufmann Publishers, 2014. 352 pp.
3. Смелянский Р.Л. Программно-конфигурируемые сети // Открытые системы. СУБД. 2012. № 9. С. 38–43. URL: http://www.osp.ru/os/2012/09/13032491
4. Learn REST: A RESTful Tutorial. URL: http://www.restapitutorial.com
5. Survey on Network Virtualization Hypervisors for Software Defined Networking. URL: http://arxiv.org/pdf/1506.07275.pdf
6. О приоритетных научных задачах, для решения которых требуется задействовать возможности федеральных центров коллективного пользования научным оборудованием // Сайт правительства РФ. URL: http://government.ru/orders/selection/405/10326/#tret

References:

1. Panesh A. Kh. Contents and prospects of technologies of software defined networks and network functions virtualization // The Bulletin of the Adyghe State University. Ser. Natural-Mathematical and Technical Sciences. 2014. Iss. 2 (137). P. 120–127. URL: http://vestnik.adygnet.ru
2. Göransson P., Black Ch. Software Defined Networks: A Comprehensive Approach. Morgan Kaufmann Publishers, 2014. 352 pp.
3. Smelyansky R.L. Software-defined networks // Open systems. DBMS. 2012. No. 9. P. 38–43. URL: http://www.osp.ru/os/2012/09/13032491
4. Learn REST: A RESTful Tutorial. URL: http://www.restapitutorial.com
5. Survey on Network Virtualization Hypervisors for Software Defined Networking. URL: http://arxiv.org/pdf/1506.07275.pdf
6. On priority research problems the solution of which requires the use of Federal centers for collective use of scientific equipment. The website of the Russian government. URL: http://government.ru/orders/selection/405/10326/#tret