SDN:
вынос мозга

В середине 1980-х профессор Дональд Норман, «гуру» в области проектирования пользовательского интерфейса, проводил семинар в исследовательском центре XEROX PARC в Пало-Альто, Калифорния. Он начал, казалось бы, с постороннего вопроса: «Кто ездит на машине с ручной коробкой передач?». В то время «коробка-автомат» еще только завоевывала американский авторынок, поэтому руки подняли большинство присутствующих. Норман оглядел лес рук и изрек: «Никто из вас не должен заниматься проектированием пользовательского интерфейса!».

Алексей ШАЛАГИНОВ,
директор по решениям, департамент промышленных решений, Huawei Enterprise Business Group

Разница между ручной и автоматической коробкой передач хорошо иллюстрирует различия, в частности, между архитектурой обычной сети и сети SDN. Простота ручной коробки заставляет водителя совершать больше движений при езде и контролировать больше параметров: обороты, скорость, включенная передача. Автоматическая коробка, при своей бóльшей технической сложности, значительно упрощает управление автомобилем, хотя и усложняет его внутреннее устройство. Точно так же основное предназначение программно-конфигурируемых сетей SDN (в русском переводе – ПКС) – упростить управление и администрирование сети передачи данных за счет декомпозиции плоскости управления, хотя и путем внесения усложнений в архитектуру сети.

Один из создателей архитектуры SDN, профессор университета Беркли в Калифорнии Скотт Шенкер, говорит: «Способность справляться со сложностью технической системы – не то же самое, что делать ее простой в обращении» [1]. При начальной реализации технической системы обращают внимание в основном на сложность, при ее развитии и совершенствовании – на простоту в обращении. Именно с этой целью были созданы стартер, стеклоочиститель, парктроник и коробка-автомат.

Однако, как указывает Скотт Шенкер, архитектура сетей до сих пор не совершила перехода от сложности реализации к простоте обращения. До сих пор компьютерные сети сложны в управлении и администрировании. Исследование Yankee Group показывает, что 62% времени простоя сетей предприятий случается из за человеческого фактора, а 80% ИТ-бюджета предприятий уходит на обслуживание сети [2].

Суть SDN – логическая абстракция уровней управления сетью в целях упрощения технической эксплуатации сети, точно так же как «коробка-автомат» скрывает (абстрагирует) техническую сложность управления автомобилем.

Чем не является SDN

SDN – не очередной механизм улучшения работы сети и не очередной набор протоколов для управления сетью. SDN – новая архитектура сети, с абстрагированием уровня управления сетью.

SDN – не какой-то особый вид сети, требующий замены оборудования или кардинальных реконструкций. Напротив, SDN использует все существующее оборудование, хотя и привносит качественно иные принципы его работы и организации управления сетью.

SDN – не революционное преобразование принципов построения сети. Это дорожная карта, которая позволяет шаг за шагом трансформировать архитектуру сети, значительно улучшить ее управляемость и администрируемость, сократить расходы по обслуживанию и, более того, увеличить ее функциональность и адаптивность.

Почему SDN?

Почему логическая (программная) структура Интернета была столь успешной? Потому что она очень хорошо структурирована и абстрагирована: приложения (www, email, видео и голос) работают поверх транспортного уровня (TCP, UDP и RTCP), который, в свою очередь, основан на уровне сети глобальной передачи пакетов (IP), лежащем на уровне канальной передачи (Ethernet), поверх физической инфраструктуры (медная пара, оптоволокно, радиодоступ…). Все это хорошо

взаимодействует при помощи стандартных интерфейсов.

Польза такой уровневой структуры сети – в декомпозиции процесса доставки пакетов, когда не нужно думать о передаче каждого бита информации. Инновации на каждом уровне можно проводить независимо, не затрагивая соседние уровни. Именно поэтому Интернет так быстро и успешно развивается.

Однако в построении сетей (networking) такой подход почему-то не прижился. Научные принципы построения сетей практически отсутствуют. Преподавание в университетах других компьютерных дисциплин (операционные системы, программирование, базы данных и пр.) ведется на основе базовых принципов, как академическая наука. Построение сетей преподается как «нагромождение протоколов». Почему так происходит? Сети остаются довольно простыми по структуре. Например, принципы сети Ethernet очень просты. Это, видимо, рудимент ARPAnet, сети военного назначения, прообраза Интернета. ARPAnet должна была работать даже в случае, если любая, сколь угодно большая часть ее будет уничтожена в ходе боевых действий. Такую сеть нельзя было делать архитектурно сложной. Именно поэтому до сих пор корпоративные сети проектируются, строятся и обслуживаются в «ручном режиме». Однако, несмотря на довольно простые приемы построения сетей, их топологическая структура все больше усложняется. Администрировать такие сети становится все сложнее. Получается замкнутый круг.

В программировании, напротив, переход от управления сложной системой к декомпозиции и упрощению этого управления произошел давно. Сначала были «машинные языки», которые не обладали никаким абстрагированием, где требовалось определять точный адрес хранения каждого бита в памяти. Затем появились операционные системы, скрывающие от программиста особенности реализации «железа», и языки высокого уровня (абстрагирования), файловые системы, виртуальная память, абстрактные типы данных. Это позволяло хорошо структурировать процесс разработки программ, давая возможность программистам не вникать в особенности технической реализации каждого компьютера. Современные языки обеспечивают еще более высокий уровень абстрагирования: объектно-ориентированное программирование и пр.

Абстрагирование – не что иное, как декомпозиция проблемы на отдельные уровни, которая ведет к созданию стандартных интерфейсов между модулями (уровнями) системы, что значительно упрощает как работу с ней, так и ее модернизацию.

Любая сеть имеет две плоскости – плоскость передачи данных (data plane) и плоскость управления сетью (control plane). Data plane производит обработку поступающих пакетов по принципу: политика передачи пакета + заголовок пакета = решение о передаче пакета. При этом обеспечивается абстрагирование услуг (механизм best effort в IP, надежность байтовой передачи в TCP и пр.). Control plane, которая определяет политику передачи на каждом сетевом узле (forwarding policy), напротив, не обеспечивает уровня абстракции для управления. На этой плоскости реализуется множество различных механизмов: распределение алгоритмов маршрутизации, изоляция пользователей и узлов (ACL, VLAN, межсетевые экраны…), инжиниринг трафика (MPLS, алгоритмы взвешивания, OSPF, BGP, IS-IS и пр).

Для обеспечения абстрагирования плоскости управления (control plane) и была создана концепция SDN. В целом она основана на трех основных абстракциях: абстракции распределения политик узлов (distribution), абстракции механизма продвижения пакетов (forwarding) и абстракции конфигурирования сетевых узлов (configuration).

Эволюция SDN

Традиционный механизм управления сетью до SDN упрощенно представлен на рис. 1.

Рис. 1. *Традиционный механизм управления до SDN.*

Сетевые узлы обмениваются между собой данными по конфигурации и топологии сети при помощи протоколов распределения (distribution). Эти алгоритмы достаточно сложны и зависят от назначения сети.

В SDN вводится понятие сетевой операционной системы NOS (рис. 2), которая работает на отдельных серверах в сети (дублированных для надежности). Сервер NOS «общается» с каждым сетевым узлом и таким образом получает глобальную топологию сети (Global Network View) и представляет ее для управляющей платформы сети в виде сетевого графа. Кроме того, NOS отвечает за централизованное конфигурирование сетевых узлов. Управляющая платформа формирует параметры, которые применяются к конкретной сети через ее граф, извлеченный NOS.

Таким образом, конфигурация сетевых узлов является, по сути, функцией сетевого графа. Механизм управления теперь сводится к определению политик работы сетевых узлов (изоляция пользователей, контроль доступа, QoS…). Однако за реализацию этих политик на конкретной физической инфраструктуре сети управляющая программа не отвечает – это задача NOS.

Можно сказать, что NOS выполняет примерно ту же функцию, что и компилятор в программировании, преобразующий выражения языка высокого уровня в блоки машинных команд, которые понимает конкретное «железо» того

Рис. 2. *Архитектура SDN версии 1 (SDN v.1)*

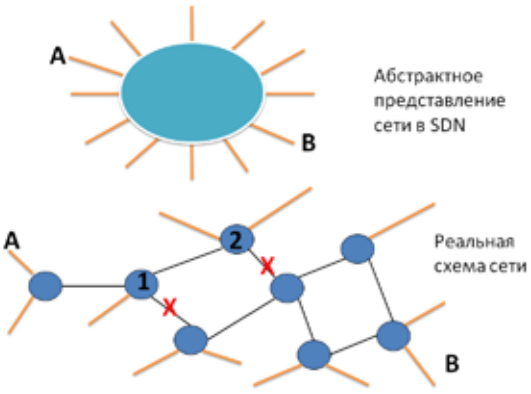

Рис. 3. *Пример абстракции SDN*

или иного компьютера. Как сам компилятор не может создать программу, так и NOS не формирует политики.

Поясним на примере (рис. 3). Предположим, что требуется обеспечить логическую изоляцию пользователя A на сети (например, клиента, подключившегося через гостевой доступ корпоративной сети WLAN в холле предприятия) от сервера базы данных B, где хранится конфиденциальная информация.

В отсутствие SDN сетевой администратор должен проанализировать реальную схему сети и сконфигурировать узлы 1 и 2 таким образом, чтобы при появлении на них пакетов с IP-адресом источника A и IP-адресом назначения B таковые отбрасывались. Однако с ростом предприятия сеть будет развиваться, в ней могут возникнуть новые узлы и маршруты и появится возможность проникновения пакетов от A к B. Сетевой администратор должен быть начеку и вовремя произвести необходимые действия по дополнительной конфигурации узлов.

Рис. 4. *Архитектура SDN версии 2 (SDN v.2)*

В случае абстрактного представления топологии сети в SDN администратору следует лишь прописать правило «маршрут от A к B = сброс пакета» в управляющей программной платформе, и оно будет действовать всегда, а изменения топологии сети будет автоматически отслеживаться в NOS.

Виртуализация сетевых функций (NFV)

NOS облегчает реализацию, выполнение функциональности сети в целом. Но она никак не определяет эту функциональность. Однако в архитектуре SDN глобальную топологию сети тоже можно вывести на уровень абстракции, как на верхней части рис. 3, чтобы обеспечить некие правила относительно того, кто с кем может связываться в сети, какой функционал доступен тому или иному пользователю, и пр. Для этого в архитектуру плоскости управления SDN вводится еще один уровень – виртуализации сетевых функций NFV (Network Functions Virtualization). NFV реализует через сетевой граф такой функционал, как трансляция сетевых адресов (NAT), параметры межсетевых экранов (firewall), распознавание вторжений, службу доменных имен (DNS), кэширование и пр., забирая эти функции у «железа» сетевых узлов, тем самым нивелируя особенности реализации на оборудовании различных производителей. Теперь эти функции можно реализовывать и конфигурировать программно, отсюда и название «программно-конфигурируемые сети».

Инфраструктура сети теперь состоит из хорошо определяемых и отслеживаемых модулей (уровней абстракции) – сетевая виртуализация NVF, сетевая операционная система NOS и интерфейс для описания передачи пакетов (например, OpenFlow). NVF и NOS тем не менее представляют собой достаточно сложный программный код, но все проблемы можно легко отслеживать, локализовывать и решать на соответствующем уровне. Нужно заботиться лишь о том, ЧТО должно происходить в сети, а не о том, КАК сделать так, чтобы это происходило.

OpenFlow

Часто говорят, что SDN – сеть, работающая на базе протокола OpenFlow. Какую же роль протокол OpenFlow играет на самом деле? При помощи этого протокола NOS переносит конфигурацию глобальной топологии сети (network view) на реальные физические устройства сети (маршрутизаторы, коммутаторы, межсетевые экраны, пограничные контроллеры, контроллеры сети беспроводного доступа и пр.), а также получает информацию о глобальной топологии сети. Важен ли этот протокол в структуре SDN? Несомненно да. Хорошее ли это решение? На данный момент – тоже да. Единственно ли возможное это решение? Определенно нет. Возможно, в будущем будет создано что-то другое, благо, что иерархичность и модульность control plane теперь легко позволяет вводить такие усовершенствования. Именно по этой причине на рис. 2 написано «управление через интерфейсы [передачи пакетов]», а не OpenFlow.

Краткая история развития и стандартизация SDN

- 2004: Ранние исследования и разработки новой концепции управления сетью в университетах Стэнфорд и Беркли (RCP, 4D, SANE, Ethane…).
- 2007: Мартин Касадо (Martin Casado), Ник Маккьюэн (Nick McKeown) из университета Стэнфорда и Скотт Шенкер (Scott Shenker) из университета Беркли основали компанию Nicira, которая начала практические разработки технологии SDN и концепции новой сетевой архитектуры.
- 2008: в университете Стэнфорд создан совместный исследовательский центр Open Networking

Research Center, в лаборатории которого продолжились исследования по тематике SDN, а также сетевой операционной системы NOX и протокола OpenFlow.

- 2011: основание некоммерческой исследовательской организации Open Networking Foundation (ONF), в которую вошли более 30 компаний, поддерживающих технологию SDN: Google, Yahoo, Verizon, DT, Microsoft, Facebook, Cisco, Juniper, HP, Dell, Broadcom, IBM, Ericsson, Huawei, VMWare, Citrix…
- 2012 – начало коммерциализации SDN.

В настоящее время стандарты для SDN активно разрабатываются в таких стандартизирующих организациях, как ONF, IETF, IRTF, ETSI и ITU-T. Вопросы построения и эксплуатации сетей SDN разрабатывает и BBF Форум (Broadband Forum). Частные вопросы применения SDN рассматриваются OIF (Optical Internetworking Forum).

В России проблемами разработки и продвижения SDN занимается Центр прикладных исследований компьютерных сетей ЦПИКС (arccn.ru) в сотрудничестве с университетами Стэнфорда и Беркли.

Архитектура SDN – своеобразное «дежа-вю» тех идей, которые в телекоммуникационной отрасли уже продвигались 20 лет назад в концепции «интеллектуальных сетей», 15 лет назад – в концепции гибких коммутаторов (Softswitch), десять лет назад – в концепции IMS (IP Multimedia Subsystem). Все эти концепции имеют в основе разделение уровней управления сетью и передачи трафика данных, или, на слэнге программистов, «вынос мозга» из сетевых устройств на уровень управления и приложений.

В чем практическая польза SDN?

В середине 1990-х, на пике развития Интернета, самой большой проблемой была скорость передачи данных по сети. Важнее всего было то, насколько быстро можно было передать, например, большую базу данных или скачать файл видеофильма с сервера. В середине 2000-х, с появлением приложений, работающих в реальном времени (IP-телефония, видеоуслуги), на повестку дня вышел параметр «качество услуг» QoS (Quality of Service). Однако услуга услуге рознь, «что для голоса хорошо, то для видео – смерть». Например, для голоса большое значение имеет задержка передачи пакетов (при задержке более 50 мс разговор становится некомфортным). Но процент потери пакетов для голоса не столь важен, например речь вполне различима даже при 5% потерянных в сети пакетов. Для видеоуслуг все наоборот – 5% потерянных пакетов существенно ухудшит качество видео, а задержка пакетов для видеовещания не столь важна и иногда даже используется для повышения качества.

Поэтому приходится вводить технологии «инжиниринга трафика» TE (Traffic Engineering), чтобы, например, обеспечить приоритет передачи пактов по протоколу SIP (голос) перед пакетами по протоколу FTP (передача файлов). Сейчас на повестке дня остро стоит задача динамического управления сетью. До «эпохи SDN» формировать сетевой трафик динамически, «в реальном времени» было практически невозможно. SDN позволяет формировать состав трафика (traffic shaping) также «в реальном времени». Например, в SDN в течение рабочего дня для VoIP можно выделить 70% полосы пропускания сети, а для видеонаблюдения – 10%. Ночью можно для голоса оставить 3%, а для видео – 70% и при этом повысить его качество, так как ночная съемка требует более высокого разрешения. Причем сделать это в несколько кликов мышкой.

Или, например, представим экстренную ситуацию. В здание бизнес-центра ворвалась группа террористов. Можно включить сирену, но никто не будет знать, где находятся террористы, куда бежать. SDN позволяет практически мгновенно включить трансляцию видеонаблюдения в высоком разрешении на все рабочие станции сотрудников, выделив для этого все 10 Гигабит корпоративной сети. Все сотрудники сразу увидят, куда именно бежать не надо, чтобы не попасть в заложники. А после того как опасность минует, быстро переключить приоритет сети на трафик VoIP, чтобы сотрудники могли позвонить домой и сообщить, что с ними все в порядке.

SDN уже имеет применение на практике. Например, в центре ядерных исследований CERN в Швейцарии SDN используется для масштабирования услуг центров обработки данных (ЦОД), балансировки нагрузки на серверы ЦОД, обеспечения мобильности виртуальных машин, оперативной связи ЦОД между собой. При исследованиях элементарных частиц в адронном коллайдере ежесекундно генерируются огромные массивы данных (порядка экзабайт), которые нужно быстро отправить в системы хранения для последующей обработки, причем с резервированием, чтобы не потерялся ни один бит. Для этого необходимо оперативно настроить передачу трафика в сети ЦОД под такую задачу. Когда экспериментальная часть закончена, наступает черед научной обработки данных. Надо мгновенно перенастроить сеть под другую задачу, чтобы эта обработка была произведена как можно быстрее, и здесь незаменимым помощником также выступает технология SDN. ■

Ссылки

1. *http://www.opennetsummit.org/archives/apr12/site/talks/shenker-tue.pdf*
2. *Z. Kerravala. Configuration management delivers business resiliency. The Yankee Group. Nov., 2002.*