УДК 004.7, 004.732

РАЗРАБОТКА ИНТЕРФЕЙСА ВЗАИМОДЕЙСТВИЯ С КОНТРОЛЛЕРОМ ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЕЙ

Чугреев Д. А., Шкребец А. Е., Шевель А. Е., Власов Д. В., Грудинин В. А., Каирканов А. Б., Садов О. Л., Титов В. Б., Хоружников С. Э., Сомс Л. Н.

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики» (НИУ ИТМО), г. Санкт-Петербург, Россия (197101, г. Санкт-Петербург, Кронверкский проспект, д.49), e-mail: xse@vuztc.ru

Дается краткое описание архитектуры и преимуществ подхода программно-конфигурируемых сетей (ПКС). Рассмотрены различные классы приложений, которые могут быть созданы на базе ПКС. Среди них выделяются средства реализации гибких и функциональных политик маршрутизации, балансировки нагрузки, системы управления облачными сервисами, средства обеспечения безопасности и механизмы зеркалирования трафика в произвольной точке сети. Сформулированы требования, которые приложения предъявляют к интерфейсу взаимодействия с контроллером. Подчеркнута необходимость высокоскоростного двунаправленного интерфейса для передачи двоичных данных. Проведен анализ существующих стандартов и реализаций интерфейсов взаимодействия с контроллером ПКС. Сравниваются структуры API OpenStack, Floodlight и OpenDaylight. Делается вывод об отсутствии стандартного подхода и необходимости поиска оптимального варианта решения проблемы для каждого отдельного случая.

Ключевые слова: Программно-конфигурируемые сети (ПКС), OpenFlow, интерфейс контроллера ПКС.

SDN CONTROLLER NORTHBOUND API: STATE OF DEVELOPMENT

Chugreev D A., Shkrebets A. E., Shevel A. E., Vlasov D. V., Grudinin V. A., Kairkanov A. B., Sadov O. L., Titov V. B., Khoruzhnikov S. E., Soms L. N.

National Research University of Information Technologies, Mechanics and Optics, Saint-Petersburg, Russia (197101, Saint-Petersburg, Kronverkskiy pr., 49), e-mail: xse@vuztc.ru

A brief description of the architecture and the benefits of Software-defined networks (SDN) is given. The different classes of applications that can be created on the basis of the SDN are discussed. Among them are policy-based routing, load balancers, cloud services management tools, security tools (ACL, firewalls and IDPS) and traffic taps from any node in the network. The application requirements for the Northbound controller API are specified. The need for high-speed duplex interface for binary data transfer is emphasized. The analysis of existing standards and implementations of Northbound SDN controller API is given. The structure of OpenStack, Floodlight and OpenDaylight APIs is analyzed. It is concluded that currently there is no standard approach and in each use case it is necessary to find an optimal solution to the problem.

Key words: Software-defined networks (SDN), OpenFlow, Northbound API.

Рассматриваются вопросы организации взаимодействия приложений с контроллером ПКС, существующие стандарты и реализации. Делается вывод о необходимости поиска для каждой задачи индивидуального решения.

Программно-конфигурируемые сети (ПКС) являются одним из приоритетных направлений развития компьютерных сетей. Основой подхода является логическая централизация управления потоками данных и информации о состоянии сети в сочетании с абстрагированием от сетевой инфраструктуры. Ключевым элементом ПКС является контроллер, на который ложатся функции поддержания актуального состояния сети, конфигурирования оборудования и реализации политик маршрутизации. Все приложения, использующие сетевые возможности, должны взаимодействовать с контроллером, прямой

доступ к сетевому оборудованию исключен. Такая архитектура позволяет избавиться от проблем, связанных с использованием сетевых устройств разных производителей, их неполной совместимостью и долгими циклами обновлений фирменного программного обеспечения. Перенос функций оборудования на программное обеспечение (ПО) должен облегчить процесс создания и развертывания новых сервисов, сделать его оперативным и автоматическим. Очевидно, что для решения стоящих перед ним задач контроллер должен иметь интерфейсы взаимодействия с сетевым оборудованием с одной стороны, и с пользователями и приложениями – с другой. Основным механизмом взаимодействия контроллера и коммутаторов в ПКС в настоящее время является протокол OpenFlow. Его продвижением и стандартизацией занимается международная организация Open Networking Foundation (ONF) [7]. Если не принимать во внимание наличие нескольких несовместимых между собой версий, получивших распространение в той или иной степени, OpenFlow предоставляет унифицированный доступ к оборудованию и обладает функциональностью, достаточной для решения стоящих перед ПКС задач. Фактически, он позволяет манипулировать механизмами продвижения пакетов в коммутаторе. Для конфигурирования оборудования в дополнение к OpenFlow необходимо применение протокола OF-Config. В то же время, взаимодействие контроллера ПКС с приложениями не стандартизовано.

Сам подход ПКС предполагает появление большого количества программных средств, как для решения существующих проблем, так и для создания инновационных сервисов. Для понимания тех требований, которые будут предъявлены к контроллеру ПКС, необходимо выделить основные классы потенциально возможных приложений. Прежде всего, это программы для реализации гибких и функциональных политик маршрутизации. На сегодняшний день из-за аппаратных ограничений основной является маршрутизация по адресу назначения. Применение ПКС позволит использовать для этих целей любые поля пакета. С практической точки зрения маловероятно построение маршрута для каждого индивидуального потока из-за проблем с масштабируемостью такого решения. Поэтому основным типом взаимодействия таких приложений с контроллером станет периодическое заполнение таблиц управления потоками и сбор статистических данных с коммутаторов. Средства балансировки нагрузки также являются очевидным направлением разработки. Их механизмы взаимодействия с контроллером аналогичны. Однако необходимо отметить, что для корректной работы алгоритмов необходимо получение точной и своевременной информации, что требует минимизации задержек. Популярность ПКС во многом определяется развитием облачных технологий. Поэтому одним из основных требований к контроллеру будет возможность интеграции с системами управления облачными сервисами, такими как OpenStack. В первую очередь будут востребованы механизмы виртуализации

сетей, многопользовательской конфигурации и инициализации. Динамические операции и анализ трафика в этом классе приложений не имеют приоритета. Одной из важнейших проблем традиционных компьютерных сетей, построенных с применением оборудования различных производителей, является невозможность проводить унифицированную политику безопасности. Современные тенденции, связанные с необходимостью обеспечивать надежный и безопасный доступ в корпоративную сеть пользователям, использующим персональные мобильные устройства, делают задачу практически неразрешимой с применением существующих технологий. При этом сами по себе средства обеспечения безопасности неоднородны. Они включают в себя как механизмы управления списками доступа (Access Control List, ACL), так и разнообразные межсетевые экраны (firewall), а также средства обнаружения и предотвращения вторжения (Intrusion Detection and Prevention Systems, IDPS), использующие, как правило, механизмы глубокого анализа пакетов (Deep Packet Inspection, DPI). Помимо доступа к полному содержимому пакета, такого рода системы зачастую предполагают и возможность его модификации. Реализация DPI потребует от контроллера наличия скоростного дуплексного интерфейса, оптимизированного для передачи двоичных данных. И, наконец, возможность контроллера ПКС зеркалировать трафик в произвольной точке сети и перенаправлять его для дальнейшего анализа, является одним из ключевых преимуществ ПКС, способным дать экономический эффект и заменить существующие технологии мониторинга (NetFlow, sFlow и другие).

Обзор потенциальных приложений показывает, что создание универсального, расширяемого, удобного в применении, эффективного и функционального интерфейса с контроллером ПКС является сложнейшей задачей. А ведь необходимо предусмотреть и механизмы композиции сервисов, средства взаимодействия приложений, позволяющие избежать дублирования базовых операций, не говоря уже об организации взаимодействия контроллеров внутри кластера.

Неизбежно возникает вопрос, возможно ли создание такого интерфейса. Создатели OpenFlow и разработчики первых контроллеров ПКС отвечают на этот вопрос отрицательно [4]. По их мнению, преждевременная стандартизация пагубно скажется на развитии ПКС в целом. А для решения конкретной задачи необходимо использовать наиболее подходящие для этого средства. Тем не менее, на основании опыта собственных разработок они рекомендуют использовать представление сети в виде потенциально непротиворечивого графа (eventually consistent graph). Вершинами такого графа могут являться как физические объекты (коммутаторы, порты, таблицы маршрутизации), так и логические (туннели, элементы конфигурации). Приложения используют базовые операции поиска, чтения и

записи для взаимодействия с интересующими их элементами графа. Они могут запросить получение сообщений при изменении значения того или иного параметра и дополнять стандартную информацию по мере необходимости. Такой подход избавляет приложения от необходимости учитывать некоторые аспекты взаимодействия контроллера с коммутаторами и другими контроллерами, но оставляет за ними возможность передать часть графа под управление конкретного контроллера в кластере и манипулировать процессами распределения информации о состоянии сети, выбирая наиболее подходящую форму непротиворечивости данных. Позиция разработчиков ясна. Имея опыт создания аналогичных решений и набор необходимых примитивов, им проще для каждого случая выбрать свой собственный вариант, нежели заниматься разработкой универсального подхода, тем более, что речь идет об единичных проектах. Точно также очевидно, что для массового привлечения разработчиков в сферу ПКС необходимо наличие более простого и доступного интерфейса, пусть и имеющего некоторые ограничения по функциональности.

В первую очередь, над созданием такого рода интерфейса должны работать организации, занимающиеся стандартизацией. Действительно, ONF создала рабочую группу (Architecture & Framework), в число задач которой входит и спецификация интерфейса контроллера ПКС (Northbound API). Однако ее председатель (Rob Sherwood, Big Switch Networks) не скрывает, что считает стандартизацию интерфейса преждевременной и в ближайших планах только сбор информации о различных вариантах применения ПКС и рассмотрение общих архитектурных проблем. Помимо ONF, вопросами ПКС также занимается IETF. Аналогом OpenFlow по версии этой организации можно назвать ForCES (Forwarding and Control Element Separation, Разделение Элементов Продвижения и Управления), который разрабатывается более 10 лет. Несмотря на то, что ForCES не нашел практического применения, различные его аспекты описывает целый ряд рекомендаций и стандартов. Вопросы взаимодействия с контроллером ПКС попадают в сферу ответственности рабочей группы I2RS (Interface to the Routing System, Интерфейс к Системе Маршрутизации). В базовом документе [1] сформулированы основные требования к интерфейсу I2RS:

- множественные асинхронные операции;
- минимальная блокировка данных при записи;
- несколько управляющих клиентов;
- высокоскоростные дуплексные операции с минимальным временем отклика;
- многоканальность;
- возможность задавать время начала и срок действия каждой операции;
- масштабируемый доступ к фильтруемой информации.

Кроме этого, рассмотрены основные существующие интерфейсы управления оборудованием: CLI, SNMP и NetConf. Возможность стандартизации интерфейсов командной строки (CLI) разных производителей исключается сразу, отмечается ряд недостатков SNMP, затрудняющих его использование для конфигурации. Предпочтение отдается NetConf, хотя и с оговоркой о необходимости создания дополнительных моделей и расширений. В этой связи интересна предыдущая редакция данного документа, в которой отмечалась невозможность реализации на основе NetConf множественных асинхронных, дуплексных и многоканальных операций. Поскольку изменений в NetConf за это время не произошло, следует сделать вывод об изменении оценок авторов, в первую очередь представителей компании Cisco. Протокол NetConf использует передачу сообщений в формате XML и предполагает использование языка моделирования YANG для семантического описания операционных и конфигурационных данных, сообщений и нотификаций. Трудно найти принципиальные преимущества этого подхода, к примеру, над информационной моделью CIM (Common Information Model, Общая Информационная Модель), разработанной организацией DMTF (Distributed Management Task Force) и также основанной на XML. Эта модель лежит в основе стандарта WBEM (Web-Based Enterprise Management) для управления распределенными вычислительными окружениями. Возможности ее расширения были продемонстрированы SNIA (Storage Networking Industry Association), создавшей на основе CIM механизм управления разнородными сетями хранения данных (СХД) SMI-S (Storage Management Initiative Specification). В то же время, нельзя не отметить, что хотя одной из задач при создании XML была простота восприятия формата человеком, широкое распространение средств автоматической генерации и анализа, а также применяемые сложные и громоздкие схемы, практически превратили его в понятный лишь машинам. В значительной степени это способствовало популярности формата представления данных JSON, который в сочетании с технологией REST доминирует при разработке Web-интерфейсов. Впрочем, учитывая необходимость эффективных дуплексных операций с двоичными данными, основанный на HTTP механизм REST не выглядит оптимальным решением. ПКС предназначены для автоматизации, а Web-интерфейсы изначально предназначались для пользователей. Приложения для ПКС будут создаваться заново, речь о переносе существующих решений с минимальными модификациями не идет. Поэтому можно было бы использовать и более эффективные механизмы. Тем более что существуют и языки для описания произвольных структур данных (ASN.1) с доступными средствами генерации программ. Есть эффективные механизмы передачи двоичных данных и удаленного вызова процедур (RPC), такие как Google Protocol Buffers [3] или максимально оптимизированный MsgPack [5].

Очевидно, что на базе текущих стандартов ONF и IETF невозможна разработка приложений, взаимодействующих с произвольным контроллером. Среди программных интерфейсов, довольно часто стандартом де-факто становится интерфейс, предложенный создателями приложения, получившего наибольшее распространение и признание. В области взаимодействия с контроллером ПКС на эту роль могут претендовать система управления облачными сервисами OpenStack [8] и OpenSource контроллеры Floodlight [9] и OpenDaylight [10]. Для взаимодействия все они предлагают вариант REST интерфейса. Контроллер ПКС NOX [6] также входит в число наиболее известных, но, как правило, он используется в качестве основы для модификации и дальнейшего развития поскольку не обладает развитым интерфейсом взаимодействия с приложениями.

Сетевой API OpenStack основан на операциях CRUD (Create Read Update Delete) и оперирует тремя основными классами сущностей: сети, подсети и порты. Сеть представляет собой изолированный канальный домен (layer 2 domain). Подсеть – блок IP адресов (IPv4 или IPv6). Порт – виртуальный порт коммутатора на заданной сети. Несмотря на предусмотренные механизмы расширения (маршрутизатор и балансировщик нагрузки), OpenStack API ориентирован на пользователей облачных сервисов и предназначен для взаимодействия с контроллером ПКС через дополнительный драйвер.

Основным преимуществом контроллера Floodlight разработчики называют его совместимость с коммерческой версией Big Network Controller. Тем не менее, первое, что пользователь видит на официальном сайте проекта, это предупреждение о том, что API находится в стадии модификации и стабильный вариант появится в ближайшее время. Анализ существующего интерфейса показывает, что основное внимание уделяется сбору статистики с коммутаторов, получению информации о текущей топологии сети и созданию новых виртуальных соединений, управлению встроенным межсетевым экраном, а также операциям с таблицей обработки потоков (Static Flow Pusher). Последнее и является принципиальным отличием от OpenStack API, механизмом более низкого уровня и непосредственной функцией контроллера. На базе этого интерфейса возможна разработка ряда приложений, но при необходимости полного доступа к содержимому пакетов придется создавать модуль для интеграции в контроллер.

Ведущее место среди Open Source ПКС проектов должен занять OpenDaylight, координацией которого занимается Linux Foundation, а в число участников входят как производители сетевого оборудования (Cisco, Juniper), так и разработчики ПО (Microsoft, Red Hat). Компании NEC и Big Switch Networks, которые уже предлагают коммерческие ПКС решения, также принимают участие в проекте и готовы передать часть своих разработок в общее пользование. Не остались в стороне IBM и HP. Потенциал этого проекта,

несомненно, велик. Но первая версия контроллера должна появиться только в третьем квартале 2013 года. Структура текущего REST API OpenDaylight во многом аналогична Floodlight: получение информации о топологии сети, манипуляции с таблицами управления потоками, сбор статистики. Основной архитектурной особенностью является применение уровня абстракции сервисов (SAL, Service Abstraction Layer) через который осуществляется взаимодействие всех компонентов системы. В качестве системы управления модулями выбрана OSGi, а для обеспечения скоростных дуплексных интерфейсов планируется применение механизмов Java RPC. Вопросы организации совместной работы приложений, использующих контроллер ПКС, пока не рассматриваются. Архитектура предусматривает возможность применения не только OpenFlow, но и других протоколов взаимодействия с коммутаторами, что скорей всего является требованием Cisco, которая планирует использовать OpenDaylight и для своей платформы ONE [2]. К сожалению, в настоящий момент трудно оценить удастся ли разработчикам решить поставленные задачи. Пока только известно то, что из-за разногласий по направлениям развития проекта программисты Big Switch Network выходят из активной разработки. Это означает, что все управление останется у Cisco.

Анализ доступной информации позволяет сделать вывод: в 2013 году (а скорей всего и в 2014) не будет принято никаких стандартов, описывающих интерфейс взаимодействия с контроллером ПКС. Оптимальным решением становится разработка собственного контроллера, пусть даже узкоспециализированного. В качестве основы для разработки может быть выбран один из доступных Open Source вариантов, среди которых следует выделить: OpenDaylight, NOX и Floodlight.

При выполнении научно-исследовательских работ в рамках государственных контрактов № 07.514.11.4152 и №14.514.11.4045 авторы сделали выбор в пользу расширения функциональности Open Source контроллера NOX при помощи вновь созданных модулей, получая тем самым узкоспециализированный контроллер для решения конкретной задачи. Тем самым удалось добиться реализации разработанных методов обеспечения качества обслуживания и приоритезации трафика систем хранения данных в сегменте ПКС. Достоинства данного решения очевидны: эффективность и минимальные требования к ресурсам.

Исследования проводились при финансовой поддержке Министерства образования и науки Российской Федерации в рамках государственных контрактов № 07.514.11.4152 и №14.514.11.4045.

Список литературы

1. Atlas A., Ward D. Interface to the Routing System Problem Statement [Электронный ресурс] – Режим доступа: http://datatracker.ietf.org/doc/draft-atlas-i2rs-problem-statement/ (дата обращения: 12.06.2013).
2. Cisco Open Network Environment [Электронный ресурс] – Режим доступа: http://www.cisco.com/go/one (дата обращения: 13.06.2013).
3. Google Developers Protocol Buffers [Электронный ресурс] – Режим доступа: https://developers.google.com/protocol-buffers/ (дата обращения: 13.06.2013).
4. Koponen T., Casado M. What Might an SDN Controller API Look Like? (and should we standardize it?) [Электронный ресурс] – Режим доступа: http://networkheresy.com/page/3/ (дата обращения: 12.06.2013).
5. MessagePack [Электронный ресурс] – Режим доступа: http://msgpack.org/ (дата обращения: 13.06.2013).
6. NOX [Электронный ресурс] – Режим доступа: http://noxrepo.org/ (дата обращения: 13.06.2013).
7. Open Networking Foundation [Электронный ресурс] – Режим доступа: http://www.opennetworking.org (дата обращения: 13.06.2013).
8. OpenStack Networking API v2.0 Reference [Электронный ресурс] – Режим доступа: http://docs.openstack.org/api/openstack-network/2.0/content/ (дата обращения: 13.06.2013).
9. Project Floodlight [Электронный ресурс] – Режим доступа: http://www.projectfloodlight.org/ (дата обращения: 13.06.2013).
10. Project OpenDaylight [Электронный ресурс] – Режим доступа: http://www.opendaylight.org/ (дата обращения: 13.06.2013).

Рецензенты:

Парфенов Владимир Глебович, д-р техн. наук, профессор, декан факультета информационных технологий и программирования Санкт-Петербургского национального исследовательского университет информационных технологий, механики и оптики (НИУ ИТМО), г. Санкт-Петербург.

Горелик Самуил Лейбович, д-р техн. наук, профессор, Санкт-Петербургского национального исследовательского университета информационных технологий, механики и оптики (НИУ ИТМО), г. Санкт-Петербург.