УДК 621.372

Е.В. Дуравкин, Е.Б. Ткачева, Иссам Саад

Харьковский национальный университет радиоэлектроники, Харьков

АРХИТЕКТУРА SDN. АНАЛИЗ ОСНОВНЫХ ПРОБЛЕМ НА ПУТИ РАЗВИТИЯ

В статье приведен анализ существующей архитектуры SDN, приведены особенности развития и взаимодействия ее компонентов. На основе полученных результатов предложены формализмы, которые позволяют определить правила согласования и взаимодействия основных компонентов и элементов управления SDN архитектуры, учитывая последовательность выполнения, тем самым объединяя анализируемые компоненты в единое целое. Приведен метод проверки соответствия компонентов сети, построенной на основе концепции SDN, требованиям спецификации. Приведенный метод позволяет проводить реактивную проверку и учитывать асинхронную природу компонентов управления.

Ключевые слова: *архитектура SDN, протокол Openflow, SDN контроллер, управляющий поток, Openflow коммутатор, верификация протоколов SDN.*

Введение

На сегодняшний день парадигма SDN приобретает все большую популярность. Множество IT организаций и сетевых провайдеров применяет ее, в первую очередь, с целью снижения стоимости сетевой инфраструктуры и издержек на ее содержание, а также обеспечения высокого уровня управляемости, защищенности и надежности сети [1].

Основной идеей SDN является разделение уровня управления сетью (control plane) и уровня передачи трафика (forwarding plane). В традиционных сетях данные функции невозможно отделить, так как они реализованы в одном устройстве, на базе одного набора системной логики.

В соответствии с концепцией SDN вся логика управления переносится на отдельные централизованные устройства, так называемые контроллеры. Именно в контроллере и реализуется функциональность, необходимая для полноценной работы приложений всей сети в целом или отдельных ее зон.

В основе подхода SDN лежит возможность динамически управлять пересылкой данных в сети с помощью открытого протокола OpenFlow. В идеале это должно позволить конфигурировать сеть как единую систему, независящую от производителя оборудования, что достигается путем создания унифицированного интерфейса между уровнем управления и уровнем передачи данных [3 – 5]. Однако на практике построение SDN с использованием оборудования различных производителей вызывает значительные трудности. В первую очередь это связанно с некорректным взаимодействием оборудования и контроллера, что вызвано несоблюдением спецификации или ошибками реализации интерфейса управления. Анализ существующих стандартов и спецификаций показал отсутствие единых понятий относительно объектов и структуры взаимодействия различных компонентов SDN архитектуры. Таким образом, возникает необходимость в разработке средств формализации и верификации процессов взаимодействия в рамках SDN.

Целью данной статьи является анализ существующих элементов SDN и выработка предложений по формализации их функционирования. Разработка средств формализации будущем позволить решить задачу верификации и оценки соответствия стандартам (спецификации) конкретных реализаций интерфейсов управления в рамках SDN.

1. Формальное представление SDN: основные компоненты и их взаимодействие

Основным отличием SDN от традиционных сетей является централизованное интеллектуальное управление и мониторинг сети, которые обеспечивает проверку, контроль и модификацию потоков передаваемых данных [1]. Элементами архитектуры SDN в соответствии с [1, 2] являются: приложения SDN, SDN контроллер [3], управляющие агенты [4], функции которых заложены в OpenFlow коммутаторы, VlowVisor [5] или интерфейс, отвечающий за передачу управляющей информации, компоненты управления и администрирования. Логическое представление архитектуры SDN приведено на рис. 1.

Приложения SDN. Приложения, которые предоставляют конечным пользователям желаемые сервисы. Приложения SDN содержат ряд требований к состоянию и поведению сетевой инфраструктуры.

Контроллер SDN. Контроллер выступает единой централизованной точкой управления, который взаимодействует с уровнем приложений посредством открытого интерфейса API, а также выполняет мониторинг и управления физическими устройствами сети посредством открытого интерфейса – протокола OpenFlow [6].

Контроллер состоит из нескольких модулей или уровней, каждый модуль отвечает за ряд необходимых функциональных возможностей [4, 7].

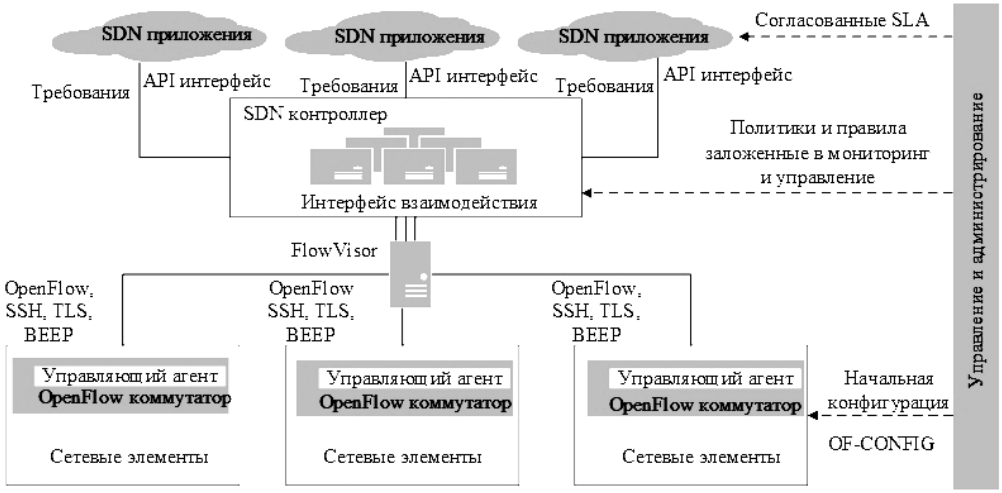

Рис. 1. Архитектура SDN: основные компоненты и их взаимодействие

OpenFlow является первым стандартизированным открытым интерфейсом, отвечающим за взаимодействие между уровнем управления и уровнем передачи данных. OpenFlow обеспечивает доступ, обмен информацией и доставку управляющих команд элементам сетевой инфраструктуры [4].

OpenFlow коммутатор. Коммутаторы OpenFlow обеспечивают непосредственное взаимодействие сетевой инфраструктуры с уровнем управления. Коммутатор содержит одну или несколько таблиц переадресации (flowtable), которые содержат все данные о потоках передаваемой информации. Записи в таблицах переадресации содержат набор полей с информацией о каждом пакете (номер входного и выходного порта, приоритет передаваемых данных, счетчик и виды действий, которые необходимо выполнить после обработки пакета (перенаправление, модификация либо сброс) [4].

FlowVisor. FlowVisor является ответственным за распределение управляющей информации между потоками данных [5]. По своей природе FlowVisor – это прозрачный прокси-сервер между коммутаторами и контроллером. При этом FlowVisor определяет, какие множества потоков относятся к той или иной сети (коммутатору) и, следовательно, передают управляющую информацию определенному контроллеру. FlowVisor обеспечивает виртуализацию потоков управляющих пакетов в отдельные срезы сети (slices) [4, 11], каждый из таких потоков имеет свое логику управления и передачи.

Компоненты управления и администрирования. Набор статических данных, которые включают внешние задачи: координацию политик и правил, установленных при проектировании бизнес-модели архитектуры SDN, начальная конфигурация оборудования и правила распределение сетевых ресурсов.

Исходя из приведенного выше описания, SDN является комплексной системой взаимодействия элементов как логической, так и физической природы. Формально сетевая структура SDN может быть задана как ориентированный мультиграф, описываемый множеством:

$$SDN = \langle N, L, F, Md, Mu, S_0 \rangle , \qquad (1)$$

где N – конечное множество сетевых элементов.

Множество N состоит из объединения следующих подмножеств:

$$N = EndNode \cup Sw \cup C \cup FlowVisor \cup Storage .$$

Под $EndNode$ подразумевается множество вычислительных элементов, основными атрибутом $EndNode$ является тип устройства. Множество Sw и C определяет множество OpenFlow коммутатор и контроллеров соответственно, $FlowVisor$ определяет множество управляющих прокси-серверов; $Storage$ – множество баз данных;

L – конечное множество возможных связей между сетевыми элементами i и j: $\{l_{ij}\} \in L$;

F – конечное множество отношений переходов для каждого элемента из множества L. Каждая сетевая взаимосвязь может быть заданна вектором статических и динамических характеристик: $f_l(l_{ij})$;

Md – управляющая информация, передаваемая от контроллера к коммутаторам;

Mu – информация, передаваемая от коммутатора к контроллеру: набор статических характеристик сети, сообщение об изменении топологии или параметров сети на уровне передачи данных;

S_0 – начальное состояние сети или начальная разметка, множество S_0 включает следующие атрибуты: $S_0 = \langle P, Conf \rangle$, под P подразумевается набор начальных политик и правил, $Conf$ – начальная конфигурация оборудования.

Применение контроллера как единой интеллектуальной точки управления позволяет значительно упростить логику работы и стоимость оборудования SDN архитектуры, так как исчезает необходимость в поддержке и обработке множества стандартов и протоколов управления на транспортном уровне [5].

Архитектура контроллера OpenFlow состоит из нескольких уровней, каждый из которых отвечает за ряд необходимых функциональных возможностей. Основными уровнями OpenFlow контроллера является уровень взаимодействия с сетью; уровень обработки OpenFlow сообщений; уровень обработки событий; уровень сетевых сервисов и внутренних приложений; интерфейс для сетевых приложений контроллера; уровень сетевых приложений [4, 7].

Уровень взаимодействия с сетью, уровень обработки OpenFlow сообщений и уровень обработки событий образуют ядро контроллера. Таким образом, контроллер может быть задан таким образом:

$$C_i = (T_{OSi}, Conf_i, Net_i(t), Mes_i(t), Ev_i(t), TC_i(t), F_{ij}(t)),$$

где $T_{OSi}, Conf_i$ – статические характеристики: T_{OSi} – тип сетевой операционной системы контроллера, $Conf_i$ – начальная конфигурация. К динамическим характеристикам относятся события, $Ev_i(t)$, сообщения $Mes_i(t)$ и уровень взаимодействия $Net_i(t)$ зависят от времени обработки и для каждого контроллера индивидуальны, $TC_i(t)$ – множество управляющих команд и $F_{ij}(t)$ – отношения переходов (определено в формуле (1)).

По своей сути контроллер выступает в роле элемента реагирования: получает сообщения от коммутаторов по каналам управления и вырабатывает отклики, которые изменяют содержимое таблиц коммутации. OpenFlow коммутатор является связующим звеном между сетевой инфраструктурой и контроллером. Коммутатор собирает статистические данные о структуре, состоянии и характеристиках уровня передачи данных, создает метки в таблицах переадресации и перенаправляет их контроллеру для принятия дальнейших решений. Коммутатор имеет набором статических и динамических характеристик:

$$Sw_i = (f_{istat}(Conf), f_{idin}(Conf, t)),$$

где $f_{istat}(Conf)$ – статические характеристики: $f_{istat}(Conf) = (Type_i, Pc_i, OF_i, Tc_i, R_{sw})$, которые включают тип устройства $Type$, количество портов P_i, версию поддерживаемого протокола OpenFlow OF_i, максимальное количество записей таблицы переадресации T_i, R_{sw} – правила обработки пакетов в зависимости от Type of Service (ToS), $f_{idin}(Conf, t)$ - динамические характеристики:

$$f_{idin}(Conf, t) = (FT_i(t), Qp_i(t), P_i(t)), \quad (2)$$

$FT_i(t)$ – состояние таблицы переадресации в заданный момент времени (определяется набором правил), $Qp_i(t)$ – тип очереди на каждом из портов коммутатора (определяется ToS – типом предоставляемого сервиса), $P_i(t)$ – характеристики порта коммутатора в заданный момент времени.

Записи в таблицах переадресации содержат набор полей с информацией о каждом пакете (номер входного и выходного порта, приоритет передаваемых данных, счетчик и виды действий, которые необходимо выполнить после обработки пакета (перенаправление, модификация либо сброс) [4]. Например, если переключатель OpenFlow получает пакет, который ему не встречался ранее, для которого нет записи в таблице переадресации, он посылает этот пакет контроллеру. На основании полученных от коммутатора данных контроллер принимает решение, как обрабатывать пакет. За удаленное конфигурирования коммутаторов, функционирующих на основе управляющей информации OpenFlow, отвечает протокол OF-CONFIG [9]. Протокол OF-CONFIG представляет OpenFlow коммутатор как абстракцию - логический коммутатор (Logical Switch), при этом одно физическое устройство может содержать несколько Logical Switch, которые отвечают за пересылку потока данных разным элементам сети - Logical Switch Capabilities. С помощью протокола OF-CONFIG производится конфигурирование основных характеристик Logical Switch - $f_{istat}(Conf)$.

В общем случае взаимодействие между уровнем управления и уровнем передачи данных осуществляется на основе двух протоколов: OF-CONFIG, который позволяет конфигурировать отдельные Logical Switch для создания качественного канала передачи управляющей информации и непосредственно протокола OpenFlow, который позволяет управлять переадресацией и модификацией пакетов.

Ключевой особенностью технологии SDN является объединение передаваемых по сети данных в отдельные классы (потоки) и применение различной логики управления для каждого потока. Такой подход более всего применим к виртуализированным сетям. Ответственным за распределение управляющей информации между потоками данных является FlowVisor [5]. По своей природе FlowVisor является прозрачным прокси-сервером между коммутаторами и контроллером. К одному FlowVisor может быть подключено несколько контроллеров, которые реализуют разную логику управления. При этом FlowVisor определяет, какие множества потоков относятся к той или иной сети и, следовательно, передают управляющую информацию определенному контроллеру. FlowVisor выделяет заданные множества потоков в отдельные срезы сети (slices) [4, 11], каждый из которых имеет свое логическое представление и логику управления. Формально FlowVisor такжекак и коммутаторы OpenFlow имеет ряд статических и динамических праметров:

$$FlowVisor_i = (f_{istat.fl}(Conf), f_{idin.fl}(Conf, t)),$$

где $f_{istat}(Conf)$ – статические характеристики: $f_{istat.fl}(Conf) = (slice_i, OF_i, R_{fli})$, которые включают количество поддерживаемых срезов или виртуали-

зированных каналов передачи управляющей информации $slice_i$, версию поддерживаемого протокола OpenFlow OF_i, R_{sw} – правила формирования срезов в зависимости от ToS и характеристик сети. $f_{idin.fl}(Conf, t)$ – динамические характеристики:

$$f_{idin}(Conf, t) = (slice_i(t).Fslice_i(t), FR_{fli}(t), B_i(t)),$$

$slice_i(t)$ – количество срезов доступное для контроллера C_i в данный момент времени, $FT_i(t)$ – состояние среза в данный момент, $FR_{fli}(t)$ – правила управления, действующие в данный момент, $B_i(t)$ – пропускная способность $FlowVisor_i$.

Таким образом, FlowVisor предоставляет каждому контроллеру собственное виденье топологии сети и обеспечивает разделение сетевых ресурсов. Контроллер, в свою очередь, ничего не знает о существовании других потоков управляющий информации и другой топологии сети, при этом обеспечивается полная изоляция передачи управляющей информации.

2. Протокол OpenFlow: организация обмена сообщениями

Как было отмечено, основным протоколом на уровне управления SDN является протокол OpenFlow, который отвечает за обнаружение изменений на уровне передачи данных и занесение их в таблицу переадресации, а также пересылку и модификацию управляющей информации между контроллером и коммутаторами.

Протокол OpenFlow, в соответствии с имеющимися спецификациями, определяет организацию обмена сообщениями об изменениях таблиц переадресации, поддерживая при этом стандартный набор параметров. Формальное представление взаимного обмена сообщениями между контроллером и коммутатором имеет вид:

$$Fl(T) : Out_{sw}(T) \xrightarrow{FV} Inp_{cont}(T) \vee Out_{cont}(T') \xrightarrow{FV} Inp_{sw}(T'),$$

где $Fl(T)$ – сообщение об изменении полей в таблице переадресации; $Out_{sw}(T)$ – конечное множество меток пакета T, передаваемого от коммутатора контроллеру, $Inp_{cont}(T)$ – конечное множество меток пакета T, полученного контроллером, $Out_{cont}(T')$ – конечное множество модифицированных меток пакета T', передаваемого от контроллера к коммутатору, $Inp_{sw}(T')$ – конечное множество модифицированных меток пакета T', полученного коммутатором.

При этом логика обработки и внесения изменений в пакет $Inp_{cont}(T)$ и формирования нового управляющего пакета $Out_{cont}(T')$ является частной (закрытой) и полностью зависит от разработчиков контроллера. Также закрытым механизмом является процесс управления виртуализированными каналами управляющей информаций $\xrightarrow{FV}$: она зависит от версии поддерживаемого OpenFlow протокола каждым из участников событий и логики работы FlowVisor.

На рис. 2 приведена схема инициализации управляющего потока в сети, построенной на основе концепции SDN. В рамках данной схемы предполагается, что оконечное устройство A и оконечное устройство B находится в одной и той же зоне обслуживания, т.е. подконтрольны одному контроллеру.

Начальным условием формирования нового потока является изменение топологии сети, в рассмотренном случае добавление нового оборудования. Рассмотрим каждый этап формирования управляющего потока и проблемы, возникающие на каждом этапе, отдельно.

1. Перенаправление пакета контроллеру осуществляется в том случае, если в таблице переадресации коммутатора не найдено ни одного совпадения для поля $Match_j$:

$$Send(t) : Out_{pac}(j) \rightarrow Tf(Match_j \notin Match_{Tf}) \Rightarrow$$

$$\Rightarrow Out_{sw}(J) \xrightarrow{FV} Inp_c(J).$$

В поле $Match_j$ пересылаемого пакета $Out_{sw}(J)$ содержится следующая информация: номер входящего порта, Ethernet адреса источника и получателя, тип Ethernet пакета, идентификатор и приоритет VLAN, IP адреса источника и получателя, тип протокола IP, Type of Service (ToS) биты протокола IP и TCP/UDP порты источника и получателя и т.д.

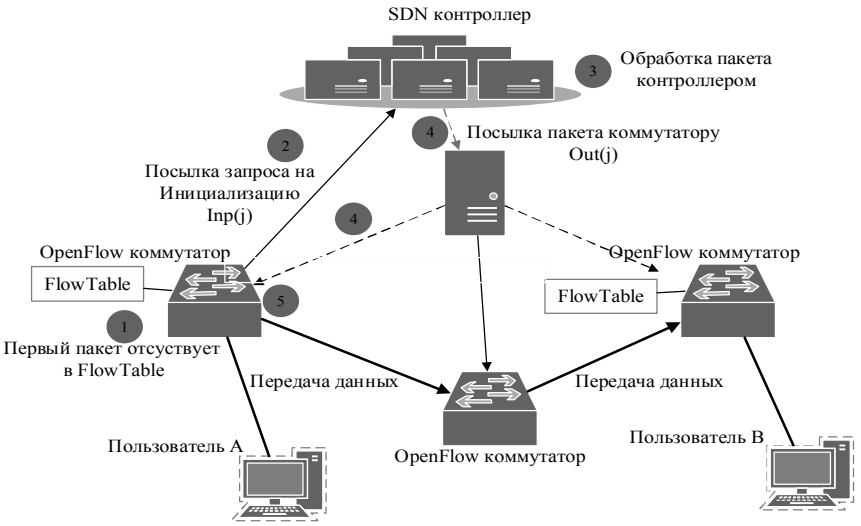

Рис. 2. Формирование управляющего потока в SDN

Обязательным требованием при этом является поддержка одной и той же версии протокола OpenFlow. Только в этом случае пакет будет корректно обработан контроллером. Возникновение ошибки возможно, если коммутатор поддерживает только протокол версии 1.0 - недостатком OpenFlow 1.0 является отсутствие поддержки протокола IPv6, способов обнаружения топологии зоны сети, отсутствие возможности многоадресной рассылки [8]. **2. Обработка пакета контроллером** может происходить различными способами. Это зависит от типа операционной системы контроллера, заложенным алгоритмам обработки, объема буферной памяти. Контроллер SDN является критической точкой отказа в работе всей сети, в процессе функционирования необходимо учитывать его надежность и производительность.

В общем виде надежность сети, построенной на основе концепции SDN, может быть задана как $R(SDN, p)$ и определяется как вероятность того, что все вершины мультиграфа SDN (формула (1)) доступны с вероятностью p на протяжении всего времени функционирования сети. Необходимым условием для полноценного функционирования централизованной структуры сети является требование $p \neq 1$. В общем виде формальное представление надежности сети может быть задано такой системой:

$$R(SDN_i, p) = \begin{cases} \sum_{i=1}^{N} r_i^{N-1}, \text{если\_}SDN_i\text{\_имеет\_} \\ N\text{\_доступных\_вершин}: p \neq 1; \\ 0, \text{если\_}C_i\text{\_недоступен}: p = 1; \\ \sum_{i=1}^{N} r_i^{k-1}, \text{если\_}p_k = 1, \end{cases}$$

где SDN – граф сети (формула (1)), r^{N-1} – последовательность достижимых вершин сети: $r^1, r^2, r^3 ... r^{N-1}$, p – вероятность отказа сетевых элементов.

3. Передача обработанного пакета коммутатору может осуществляться напрямую или с помощью FlowVisor посредствам инкапсуляции различных пакетов управляющей информации в единый канал:

$$\text{Translate}(t): \text{Out}_{\text{cont}}(T') = \text{FV}(T') \rightarrow \text{Inp}_{\text{Sw}}(T'),$$

где под функцией $\text{FV}(T')$ подразумеваются изменения, вносимые в поля заголовка j-го пакета, модификации подвергается поле Match .

Однако, модификация потоков управляющей информации с помощью FlowVisor зачастую затруднительна. Спецификация OpenFlow не определяет процедуру модификации потоков FlowVisor [8, 10, 11]. Отсутствия единых сетевых правил и политик передачи управляющей информации, принадлежащих различным зонам сети, приводит к возникновению ошибок.

Например, при оркестровке управляющих потоков в архитектуре SDN отсутствуют механизмы изоляции двух срезов. FlowVisor не имеет возможности предотвратить вмешательство управляющей информации из одного потока в другой. Это, в первую очередь связанно с тем, что два разных приложения могут быть запущены на одном устройстве (имеют один IP-адрес) [7].

4. Получение пакета OpenFlow коммутатором осуществляется тем портом, который указан в заголовке Match_j, как правило, данный тип заголовка не претерпевает существенных изменений. Пакет обрабатывается в соответствии с установленным приоритетом очередей на портах коммутатора, данные пакета заносятся в таблицу переадресации.

$$\text{Receive}(t): \text{Inp}_{\text{Sw}}(T') \rightarrow \text{Tf}(f_{\text{idin}}(\text{Conf}, t)).$$

Занесение новой записи в таблицу переадресации Tf выполняется в соответствии с текущими параметрами коммутатора. После создание новой записи коммутатор осуществляет пересылку данных сетевым элементам, в соответствии с инструкциями, полученными от контроллера. В соответствии со спецификацией OpenFlow пакеты $\text{Out}_{\text{cont}}(T')$ могут быть доставлены как коммутатору, который инициировал установления соединения, так и все коммутаторам сети. Спецификация протокола не имеет на сегодняшний день четких инструкций, которые описывают тип доставки управляющих сообщений.

3. Построение сетей на основе концепции SDN

В предыдущих разделах приведено формальное представление сети, построенной на основе концепции SDN и механизм формирования управляющего потока OpenFlow протокола (частный случай). В рамках [1, 2, 5] концепция SDN предполагает объединение сетей различных провайдеров воедино. Обобщенная структура сети приведена на рис. 3.

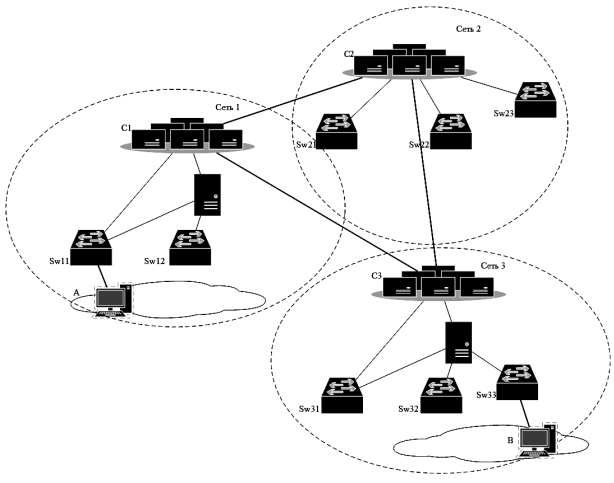

Рис. 3. Обобщенна структура сети, построенной на основе концепции SDN

В данном случае каждая сеть SDN задана своим рядом параметров и атрибутов

$$N, L, F, Md, Mu, S_0,$$

а передача информации из узла А (принадлежащего полю виденья Sw1), в узел В (принадлежащего полю виденья Sw2) может быть задана как прогон вида:

$$Fl(T): (SDN_1, C_1) \xrightarrow{j_1} (SDN_2, C_2) \xrightarrow{j^3_1} (SDN_3, C_3),$$

где SDN_1 – сетевая конфигурация, C_1 – состояние контроллера, j – состояние пакета.

Как было отмечено ранее, сетевая конфигурация, состояние контроллера и состояние являются динамическими параметрами. На их корректное взаимодействие накладывается ряд факторов: однотипное формирование таблиц маршрутизации, поддержка одно и того же протокола OpenFlow, применение однотипной сетевой операционной системы на контроллерах, одинаковые механизмы обработки пакетов. Все эти параметры следует учитывать при проектировании единой сети.

Проектирование SDN отличается от проектирования традиционных сетей и протоколов, так как сети SDN по своей природе являются сервис-ориентированными и должны гибко изменять значение параметров и характеристик в реальном масштабе времени, а также учитывать асинхронный режим взаимодействия элементов и возможность множественного доступа к ресурсам сети [12].

В качестве модели жизненного цикла SDN рядом разработчиков применима спиральная модель [13]. Она наследует все достоинства каскадной модели, ее характерной особенностью является возможность внесения изменений на ранних стадиях разработки и завершение каждого этапа проверкой полученных результатов, что обеспечивает существенное снижение временных и материальных затрат.

Однако полноценное функционирование подобной сети на сегодняшний день затруднительно. Основные проблемы, возникающие в процессе функционирования сетей, построенных на основе концепции SDN, связанны с отсутствием стандартизированных протоколов взаимодействия, наличием неточностей и расхождений в существующих версиях спецификаций.

Отсутствие единых стандартов также существенно влияет и на проектирование SDN: разработчикам необходимо учитывать, самостоятельно формализировать и внедрять в свои решения все требования, заложенные в различных версиях OpenFlow протоколов.

Таким образом, на различных этапах жизненного цикла SDN возможно возникновение ряда ошибок, причинами которых являются [13, 14]:

- неполный анализ предметной области;
- ошибки, возникающие при формализации спецификации;
- различная интерпретация стандартов;
- логические ошибки проектирования;
- неполное соответствие предъявляемым требованиям;
- не корректное сокращение функциональности протокола;
- избыточность в реализации протокола;
- ошибки кодирования реализации протокола.

Для устранения ошибок, можно сформировать ряд требований, применимых на каждом этапе жизненного цикла протоколов SDN, которые позволят избежать ошибок и повысить эффективность функционирования сети:

- формальное описание требований, предъявляемых к приложениям SDN, учитывая особенности среды и характера передаваемых данных;
- разработка формальных методов проверки совместного функционирования контроллеров различных производителей: проверка непротиворечивости команд и правил, заложенных в логику работы контроллера;
- разработка формальных методов проверки взаимодействия между уровнем приложений и передачей данных: проверка корректности выполнения команд и их непротиворечивость;
- формальное унифицированное представление спецификаций, которые обеспечат единый подход к взаимодействию контроллеров и оркестровке приложений, представленных различными разработчиками;
- разработка формальных методов проверки соответствия готового решения SDN его спецификации.

Осуществление всех перечисленных шагов возможно в процессе верификации на каждом этапе разработки. При таком подходе возможно существенное сокращение стоимости разработки и внедрения новых приложений, а также приобретение прогностических знаний о межуровневом взаимодействии и создание шаблонных подходов.

Выводы

Инфокоммуникационные системы, построенные на основе концепции SDN, представляют собой сложные многокомпонентные структуры. Отделение уровня управления от уровня передачи данных, наличие централизованного элемента управления и инновационных механизмов виртуализации каналов управляющей информации накладывает ряд требований на функционирование SDN.

Однако отсутствие стандартом и спецификаций, обеспечивающих единый подход к построению архитектуры SDN и протоколам взаимодействия ее компонентов, приводят к возникновению ряда ошибок, влияющих на эффективную и корректную работу всей системы.

В статье приведены формальная модель архитектуры SDN и процесса формирования управляющего канала на основе потокола OpenFlow (в процессе формализации были рассмотрены версии 1.0.0, 1.2.0, 1.3.0). Анализ спецификаций, выполненный при построении формальных моделей, показал, что, что различные версии спецификации протокола OpenFlow сильно разнятся. Так, в спецификации OpenFlow 1.0 поля Match таблицы FlowTable поддерживают протокол IPv6, инкапсуляцию пакетов на уровне L2 и широковещательные запросы. Отсутствие данной информации приводит к отбрасыванию пакетов или их некорректной обработке.

Отсутствие единых требований к алгоритмам занесения новых данных в таблицу переадресации, обработки новых сообщений от коммутатора $\mathrm{Inp}_{\mathrm{Sw}}(\mathrm{T})$, модификации пакетов $\mathrm{Out}_{\mathrm{cont}}(\mathrm{T}')$ и создания виртуальных каналов $\mathrm{FV}(\mathrm{T}')$ приводит к их различной реализации, что существенно сказывается на качестве работы всей сети.

Список литературы

1. Architecture SDN [Electronic resource] // Open Networking Foundation. – [2014]. – Mode of access: https://www.opennetworking.org/.

2. Software-Defined Networking: The New Norm for Net works [Electronic resource] // Open Networking Foundation. – [2012]. – Mode of access: https:// www.opennet working.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf.

3. OpenFlow Switch Specification, Version 1.3.0 (Wire Protocol 0x04) [Electronic resource] // Open Networking Foundation. – [2012]. – Mode of access: https://www. open-networking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf.

4. Openflow tutorial. [Electronic resource] – 2012. – Mode of access: http://www.openflow.org/wk/index.php/ OpenFlow_Tutorial.

5. Отчёт о НИР (Промежуточный). – 2013 – [Электронный ресурс]. – Режим доступа к ресурсу: cs.msu.ru/sites/cmc/files/scproj/4047_otchet_nir1_0.pdf.

6. FlowVisor: A Network Virtualization Layer [Electronic resource] // Open Networking Foundation. [2009]. – Mode of access: http://archive.openflow.org/downloads/ technicalreports/openflow-tr-2009-1-flowvisor.pdf.

7. OpenFlow Switch Specification (Series) [Electronic resource] // Open Networking Foundation. – [2014]. – Mode of access: https://www.opennetworking.org/sdn-resources/onf-specifications/openflow.

8. Guillermo Romero de Tejada Muntaner. Evaluation of OpenFlow Controllers [Electronic resource] – [2012]. – Mode of access: http://www.valleytalk.org/wp-content/uploads/2013/02/Evaluation_Of_OF_Controllers.pdf.

9. Galicia Supercomputing Center. OpenFlow and SDN Technical Report // Technical Report CESGA-2014-001 [Electronic resource] – [2014]. – Mode of access: https://www.cesga.es/en/biblioteca/.../id/754.

10. OF-CONFIG 1.2. OpenFlow Management and Configuration Protocol [Electronic resource] // Open Networking Foundation. – [2014]. – Mode of access: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.2.pdf.

11. Baldoni M. Verifying the conformance of web services to global interaction protocols: A first step / M. Baldoni, C. Baroglio, A. Martelli // International Workshop on Web Services and Formal Methods. – 2005. – P. 27.

12. Egawa T. SDN standardization Landscape from ITU-T Study Group 13 / T. Egawa // ITU Workshop on SDN Geneva, Switzerland, 4 June 2013.

13. Брауде Э.Дж. Технология разработки программного обеспечения / Э.Дж. Брауде. – СПб.: Питер, 2004. – 656 с.

Поступила в редколлегию 19.01.2014

Рецензент: д-р техн. наук, проф. Д.В. Агеев, Харьковский национальный университет радиоэлектроники, Харьков.

АРХІТЕКТУРА SDN. АНАЛІЗ ОСНОВНИХ ПРОБЛЕМ НА ШЛЯХУ РОЗВИТКУ

Є.В. Дуравкін, О.Б. Ткачова, Іссам Саад

У статті наведено аналіз існуючої архітектури SDN, наведені особливості розвитку та взаємодії її компонентів. На основі отриманих результатів запропоновано формалізми, які дозволяють визначити правила взаємодії основних компонентів та елементів управління SDN архітектури, враховуючи послідовність їх виконання, тим самим об'єднуючи аналізовані компоненти в єдине ціле. Наведено метод перевірки відповідності компонентів мережі, побудованої на основі концепції SDN, вимогам специфікації. Наведений метод дозволяє проводити реактивну перевірку і враховувати асинхронну природу компонентів управління.

Ключові слова: *архітектура SDN, протокол Openflow, SDN контролер, керуючий потік, Openflow комутатор, SDN верифікація протоколів.*

SDN ARCHITECTURE. ANALYZES OF MAIN PROBLEMS IN THE DEVELOPMENT

Ie.V. Duravkin, O.B. Tkachova, Issam Saad

The article provides an analysis of the existing architecture of SDN, are the features of the development and interaction of its components. On the basis of the results proposed formalisms that allow to define matching rules and interaction of the main components and controls SDN architecture, given the sequence of execution. A method for checking the conformity of network components, built on the concept of SDN, requirements specification. The above method allows the reactive test and take account of the asynchronous nature of control components.

Keywords: *architecture SDN, Openflow protocol, SDN controller controlling the flow Openflow switch, verification protocols SDN.*