Лабораторный стенд для тестирования возможностей интеграции ПКС-сетей и традиционных сетей

О.Р. Лапонина, М.Р. Сизов

Аннотация **– В статье рассматриваются возможные варианты миграции к программно-конфигурируемым сетям (ПКС), а также определены требования к лабораторному стенду, позволяющему тестировать различные варианты интеграции ПКС в архитектуру традиционных сетей передачи данных. Основное внимание уделяется возможности тестирования ПКС без использования реального оборудования, исключительно в лабораторных условиях, с применением общедоступных и/или коммерческих программных решений. Эмулируемый сегмент сети реализован на компьютере с ОС Windows, на котором установлен графический симулятор сети GNS3. В среде GNS3 созданы виртуальные копии маршрутизатора Cisco, виртуальная машина VirtualBox с ОС Ubuntu/Linux, сетевой коммутатор и два виртуальных хоста. Работа маршрутизатора в среде GNS3 эмулируется при помощи эмулятора маршрутизатора Cisco Dynamips с использованием реального образа программного обеспечения Cisco IOS.**

Ключевые слова **– программно-конфигурируемые сети, ПКС, ПКС-контроллер, OpenFlow.**

I. Введение. Варианты миграции на ПКС-архитектуру

Определение ПКС

Традиционные сетевые устройства сочетают в себе функционал по передаче цифровой информации в компьютерных сетях и функционал по управлению этой передачей. Такая архитектура сетевых устройств позволяет достичь максимальной автономности в работе, но в то же время, требует аппаратной поддержки функционирования сложных распределённых сетевых протоколов (например, MPLS, BGP и др.), что приводит к усложнению и удорожанию оборудования.

Статья получена 16 августа 2017.

О.Р. Лапонина – МГУ имени М.В. Ломоносова (email: laponina@oit.cmc.msu.ru).

М.Р. Сизов – МГУ имени М.В. Ломоносова (email: marat_sizov@mail.ru).

В основе ПКС лежит тот или иной уровень разделения функционала передачи данных от функционала управления этой передачей (см. [1], [4], [10]). Обычно ПКС представляет собой L2/L3 архитектуру, в которой централизованный контроллер управляет передачей данных набора распределённых коммутаторов с использованием управляющего протокола, например, OpenFlow (см. [2], [5], [6], [7], [8]).

Применение контроллера ПКС позволяет абстрагироваться от управления сетью, как набором отдельных сетевых устройств, а использование стандартного API предоставляет широкие возможности по разработке узко специализированных приложений – надстроек для контроллера, позволяет получить ещё больший уровень абстракции и максимально адаптировать сеть под используемые приложения и типы передаваемых данных.

Некоторые преимущества технологии ПКС:

- Снижение капитальных расходов за счёт упрощения функционала сетевых устройств;
- Снижение операционных расходов за счёт конфигурирования сервисов из одной точки управления и через единый интерфейс;
- Увеличение надёжности за счёт автоматизации низкоуровневых операций по управлению сетью;
- Снижение времени предоставления новых сервисов и выполнения изменений в текущих профилях сети;
- Централизованное видение сети, близкая к реальному времени сетевая статистика и возможность интеграции с сетевой аналитикой.

Варианты миграции к ПКС (см. [9]):

1. Миграция сервиса. Данный вариант подразумевает применение ПКС на коммутаторах, в конечных точках предоставления конкретного сервиса (рис. 1). Эти коммутаторы поддерживают только протокол OpenFlow и находятся

под управлением контроллера ПКС, в то время как все промежуточные сетевые устройства рассматриваются как «чёрный ящик» и принципы передачи данных, применяемые в них, не имеют никакого значения. Данный вариант миграции также можно охарактеризовать как ПКС наложения.

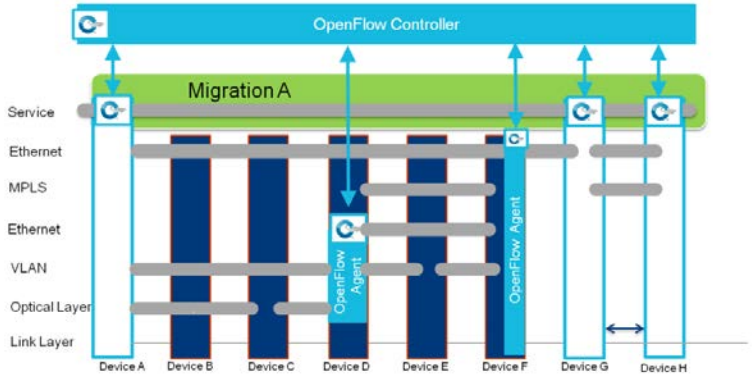


Рис. 1. Миграция сервиса

2. Миграция полного стека TCP/IP. Данный вариант миграции подразумевает применения ПКС на одном или нескольких коммутаторах, выполняющих передачу трафика на определённом участке сети (рис. 2). Здесь на контроллер ПКС и протокол OpenFlow возлагается задача обеспечить полнофункциональное взаимодействие с распределёнными протоколами наследуемых сетей по всей иерархии стека TCP/IP, в зависимости от применяемых технологий передачи данных. Коммутаторы в зоне ответственности контроллера ПКС при этом могут поддерживать только протокол OpenFlow, либо OpenFlow совместно с традиционной коммутацией и маршрутизацией (концепция гибридного коммутатора).

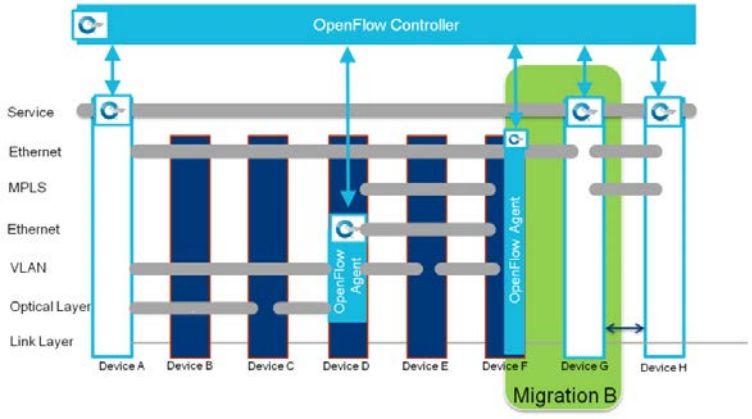


Рис. 2. Миграция полного стека TCP/IP

3. Миграция части стека TCP/IP. Данный вариант миграции подразумевает применение ПКС на одном коммутаторе и только для некоторых уровней стека TCP/IP (например, уровень доступа к сети), в то время как остальные уровни стека (например, Интернет) могут продолжать использовать существующий уровень управления и протоколы передачи данных (рис. 3).

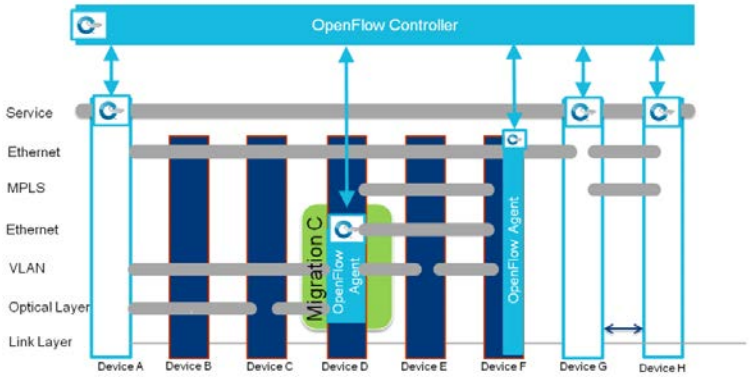


Рис. 3. Миграция части стека TCP/IP

В данной статье будут рассмотрены:

- Тестовые сценарии интеграции ПКС в архитектуру традиционных сетей.
- Лабораторный стенд, демонстрирующий возможности совместного использования ПКС и традиционных сетей.

II. Тестовые сценарии интеграции ПКС с архитектурой традиционных сетей

Тестирование возможностей ПКС и оценку потенциала их внедрения в архитектуру сетей с традиционной коммутацией и маршрутизацией, можно проводить по следующим направлениям:

- Тестирование возможности реализации и решения поставленных задач на основе ПКС, как в лабораторных, так и в реальных условиях, с применением реального оборудования и общедоступных и/или коммерческих программных решений;
- Тестирование возможности реализации и решения поставленных задач на основе ПКС без использования реального оборудования, исключительно в лабораторных условиях, с применением общедоступных и/или коммерческих программных решений.

Преимуществом первого направления тестирования является возможность проверить решение ПКС на реальной сети, с реальными задачами и используя сетевое оборудование, которое планируется установить в окончательном варианте реализации решения ПКС. В этом случае, например, появляется возможность оценить реальные количественные характеристики по пересылке пакетов, и сравнить их с характеристиками наследуемой сети. Недостатком же является необходимость обеспечить наличие необходимого количества соответствующего сетевого оборудования, особенно при сравнительном тестировании устройств разных производителей. Данное ограничение особенно актуально в случаях, когда выгоду от внедрения решения ПКС ещё предстоит определить, и она не очевидна.

Преимуществом второго направления тестирования является возможность проверить принципиальную возможность реализации решения ПКС на практике, используя исключительно программные продукты, до выделения средств на закупку реального сетевого оборудования. Недостатком же является отсутствие возможности оценить реальные количественные характеристики решения ПКС, так как, к примеру, задержка по передаче пакетов напрямую зависит от аппаратных характеристик конкретного оборудования и каналов передачи данных, а в лабораторных условия на виртуализованных сетевых устройствах эти параметры будут отличаться от параметров реальных устройств. Таким образом, данный вариант не заменяет полноценного тестирования, но позволяет сравнивать характеристики и возможности различных контроллеров ПКС на практике, моделировать и проверять различные варианты внедрения ПКС, разрабатывать и тестировать приложения для управления контроллером и сетью через стандартный API.

Далее описывается лабораторных стенд, позволяющий протестировать некоторые возможности сетей ПКС, а также процедуру интеграции сетей ПКС в архитектуру традиционных сетей с использованием общедоступных инструментов.

III. Наиболее популярные ПКС-контроллеры с открытым кодом

Перечислим наиболее популярные ПКС-контроллеры с отрытым кодом (см. [2]).

Таблица 1. Наиболее популярные ПКС-контроллеры с открытым кодом

Контроллер	Описание
OpenDaylight	Фонд Linux, Hydrogen.

	Первый одновременный выпуск OpenDaylight, представляющий три версии для широкого круга пользователей и быстрым развёртыванием: базовый выпуск, выпуск для виртуализации и выпуск для сервисных провайдеров (см. [12]).
Floodlight	Уровня организации, лицензируемый Apache, основанный на Java контроллер OpenFlow, альтернатива Open Daylight для коммерческого использования.
Ryu	Компонентная среда ПКС, основанная на Python.
POX	Младший брат NOX. В основном нацелен на исследовательское применение. Основан на Python.
NOX	Оригинальный контроллер OpenFlow, способствует разработке быстрых C++ контроллеров на Linux.
Trema	Полностековая, простая в использовании среда разработки контроллеров OpenFlow на Ruby и C.
Beacon	Быстрый, кросс-платформенный, модульный, основанный на Java контроллер OpenFlow, поддерживающий одновременно операции на основе событий и потоков.
Ovs-controller	Банальный рекомендуемый контроллер, поставляемый вместе с Open vSwitch.

IV. Лабораторный стенд для совместного использования ПКС и традиционных сетей

Требования к лабораторному стенду:

- На стенде должно применяться общедоступное программное обеспечение и оборудование;
- Стенд должен позволять эмулировать реальные сетевые устройства в среде виртуализации с целью тестирования в отсутствии реального сетевого оборудования, причём позволять эмулировать работу как коммутаторов OpenFlow, так и оборудования с традиционной коммутацией;
- Стенд должен иметь возможность передавать сетевой трафик не только внутри эмулируемого сегмента сети, но также за пределы рабочей станции, на которой он запущен, вплоть до обмена трафиком с сетью Интернет;
- Виртуальные сетевые устройства внутри эмулируемого сегмента сети должны иметь возможность взаимодействовать с реальными сетевыми устройствами, находящимися в реальной сети за его пределами, как если бы они сами были реальными устройствами;
- Программный контроллер ПКС может размещаться как внутри эмулируемого сегмента сети, так и в реальной сети за его пределами;
- Аппаратный контроллер ПКС (при наличии) должен иметь возможность управлять коммутаторами OpenFlow внутри эмулируемого сегмента сети, находясь при этом в реальной сети за его пределами;
- Стенд должен позволять эмулировать рабочие станции, на которых будет возможна установка дополнительного программного обеспечения в целях тестирования.

С учётом вышеуказанных требований, предлагается лабораторный стенд, использующий следующую комбинацию компонентов и инструментов тестирования:

- графический симулятор сети GNS3 (см. [13]);
- эмулятор компьютерной сети Mininet (см. [11]);
- программа - анализатор трафика для сетей Ethernet Wireshark;
- команды интерфейса командной строки ping и traceroute для отслеживания пути прохождения трафика и задержки;
- программа – генератор TCP и UDP трафика для тестирования пропускной способности сети Iperf;
- контроллер ПКС OpenDaylight;
- ноутбук с операционной системой Ubuntu/Linux;
- стационарный компьютер с операционной системой Windows;
- локальная компьютерная сеть (для подключения контроллера ПКС к виртуальной сети Mininet).

Для проведения различных тестов, конфигурация лабораторного стенда может меняться. В общем виде схема стенда представлена на рис. 4.

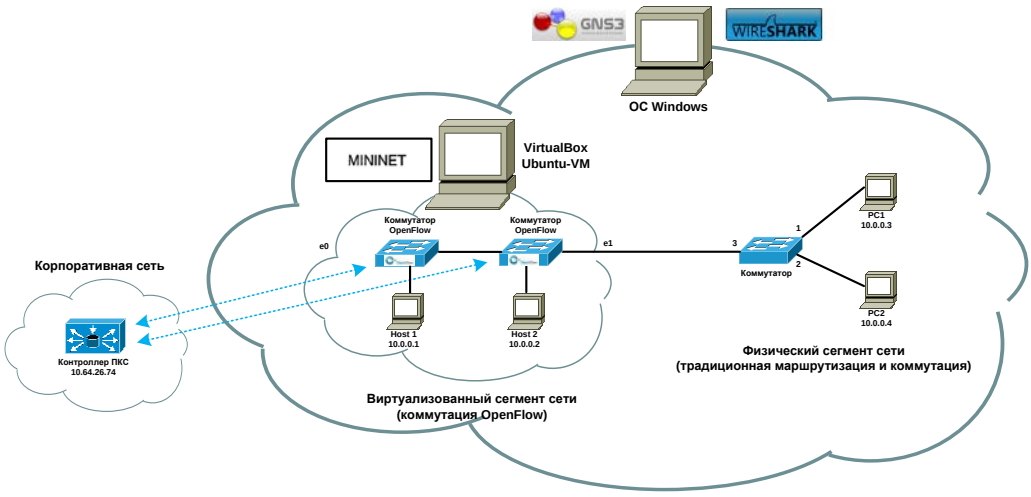


Рис. 4. Схема лабораторного стенда

Эмулируемый сегмент сети реализован на компьютере с операционной системой Windows, на котором установлен графический симулятор сети GNS3. В среде GNS3 созданы виртуальные копии маршрутизатора Cisco (R1), виртуальная машина VirtualBox (см. [14]) с операционной системой Ubuntu/Linux (см. [3]), сетевой коммутатор (SW1) и два виртуальных хоста (PC1 и PC2). Работа маршрутизатора R1 в среде GNS3 эмулируется при помощи эмулятора маршрутизатора Cisco Dynamips (см. [15]) с использованием реального образа программного обеспечения Cisco IOS. Контейнер VirtualBox предназначен для запуска эмулятора компьютерной сети Mininet, позволяющей развёртывать и изучать ПКС. Коммутатор SW1 в среде GNS, в отличии от маршрутизатора R1, не использует реальный образ программного обеспечения IOS, поэтому в текущей конфигурации стенда он поддерживает ограниченный функционал и выполняет скорее роль концентратора. Виртуальные хосты PC1 и PC2, так же, как и коммутатор SW1 не имеют полноценной операционной системы, но могут взаимодействовать с использованием стека протоколов TCP/IP с другими устройствами, а также выполнять большинство команд из интерфейса командной строки, например, Ping и Traceroute. На рис. 5 представлено отображение лабораторного стенда в среде GNS3.

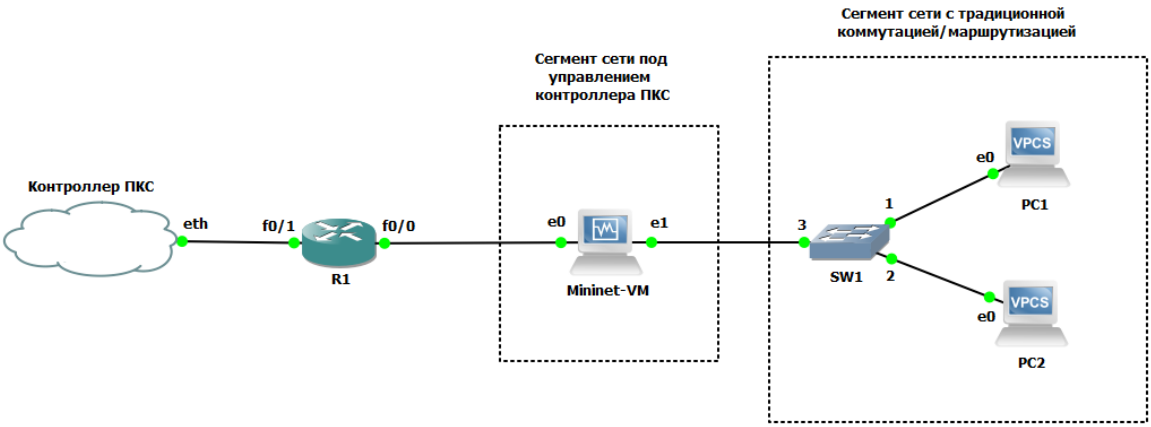


Рис. 5. Отображение лабораторного стенда в среде GNS3

Графический симулятор сети GNS3 позволяет использовать сетевой интерфейс рабочей станции, на которой он запущен, для соединения виртуального сегмента сети с реальной физической сетью. На рис. 5 видно, что интерфейс виртуализованного маршрутизатора R1 подключен к интерфейсу Ethernet рабочей станции Windows, что позволяет внешнему контроллеру ПКС управлять коммутаторами OpenFlow внутри виртуализованного сегмента сети.

На рис. 6 представлен графический интерфейс контроллера OpenDaylight и его видение топологии сети.

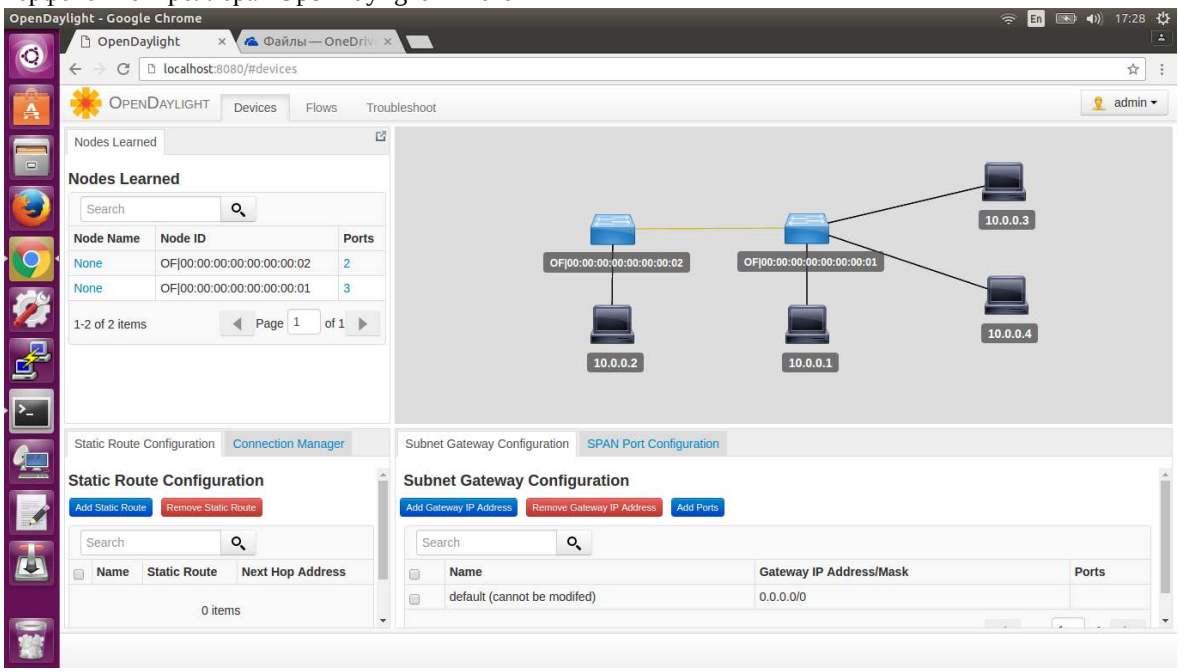


Рис. 6. Графический интерфейс контроллера OpenDaylight и топология сети

Если сравнивать рисунки 5 и 6, видно, что в среде симулятора GNS3 сегмент сети с коммутаторами OpenFlow отображается в виде некоторого «чёрного ящика», и, в то же время, контроллер ПКС не «видит» сетевые устройства традиционной сети (маршрутизатор R1 и коммутатор SW1). Соответственно, в данной конфигурации лабораторного стенда реализован вариант миграции полного стека TCP/IP к ПКС, рассмотренный во введении. Здесь сегмент сети из двух коммутаторов OpenFlow расположен между двумя устройствами наследуемой сети (маршрутизатором R1 и коммутатором SW1).

Внутри контейнера VirtualBox выполняется программа эмуляции виртуальной сети ПКС Mininet. Она позволяет создавать различные конфигурации сетей ПКС, с разной топологией, количеством коммутаторов, хостов и контроллеров, причём контроллер, с точки зрения программы, может быть как внутренним, так и внешним. Интерфейс программы Mininet также позволяет производить настройку элементов эмулируемой сети ПКС и проводить различные тесты, например, проверять прохождение эхо-запросов между узлами и пропускную способность сети.

В текущей конфигурации лабораторного стенда реализована сеть, состоящая из двух коммутаторов OpenFlow, находящихся под управлением контроллера ПКС, соединённых между собой, и к каждому коммутатору напрямую подключен компьютер (Host1 и Host2). Коммутатор SW1 и внешний контроллер ПКС подключены к сегменту ПКС через сетевые интерфейсы виртуальной машины Ubuntu контейнера VirtualBox. В качестве контроллера ПКС используется внешний (по отношению к ПО Mininet) контроллер OpenDaylight, запущенный на отдельной рабочей станции с ОС Ubuntu, вне виртуальной среды GNS3, и подключен к рабочей станции Windows с лабораторным стендом через локальную сеть.

Стенд обладает следующими возможностями:

- С использованием общедоступных и бесплатных программных средств позволяет изучать и тестировать различные конфигурации и топологии традиционных сетей, сетей ПКС, так и их интеграции друг с другом;
- Проводить тестирование исключительно с использованием виртуализованных сетевых устройств;
- Подключать виртуализованную сеть стенда к локальной сети и Интернет для тестирования с частичным использованием реальных сетевых устройств или удалённого контроллера ПКС, а также для встраивания виртуализованного

сегмента сети в определённый участок локальной сети;

- Полностью моделировать работу сетевых устройств традиционной сети, с использованием реальных образов программного обеспечения маршрутизаторов;
- Перехватывать пакеты на любом интерфейсе устройств программой WireShark для последующего анализа передаваемых пакетов и изучения алгоритмов работы сетевых протоколов;
- Использовать интерфейс командной строки виртуальных хостов для ввода поддерживаемых команд, например, ping и traceroute;
- Использовать контейнеры VirtualBox с виртуальными машинами любых операционных систем для запуска любых приложений и программ внутри лабораторного стенда;
- Производить конфигурирование виртуализованных сетевых устройств и хостов стандартными для них средствами (интерфейс командной строки, web – интерфейс);
- Производить конфигурирование виртуализованных коммутаторов OpenFlow как посредством интерфейса программного обеспечения Mininet, так и с помощью удалённого контроллера ПКС.

Примеры, демонстрирующие возможности лабораторного стенда:

1. Перехват трафика, пересылаемого коммутаторами OpenFlow контроллеру ПКС, при помощи программы WireShark (Рис. 7).

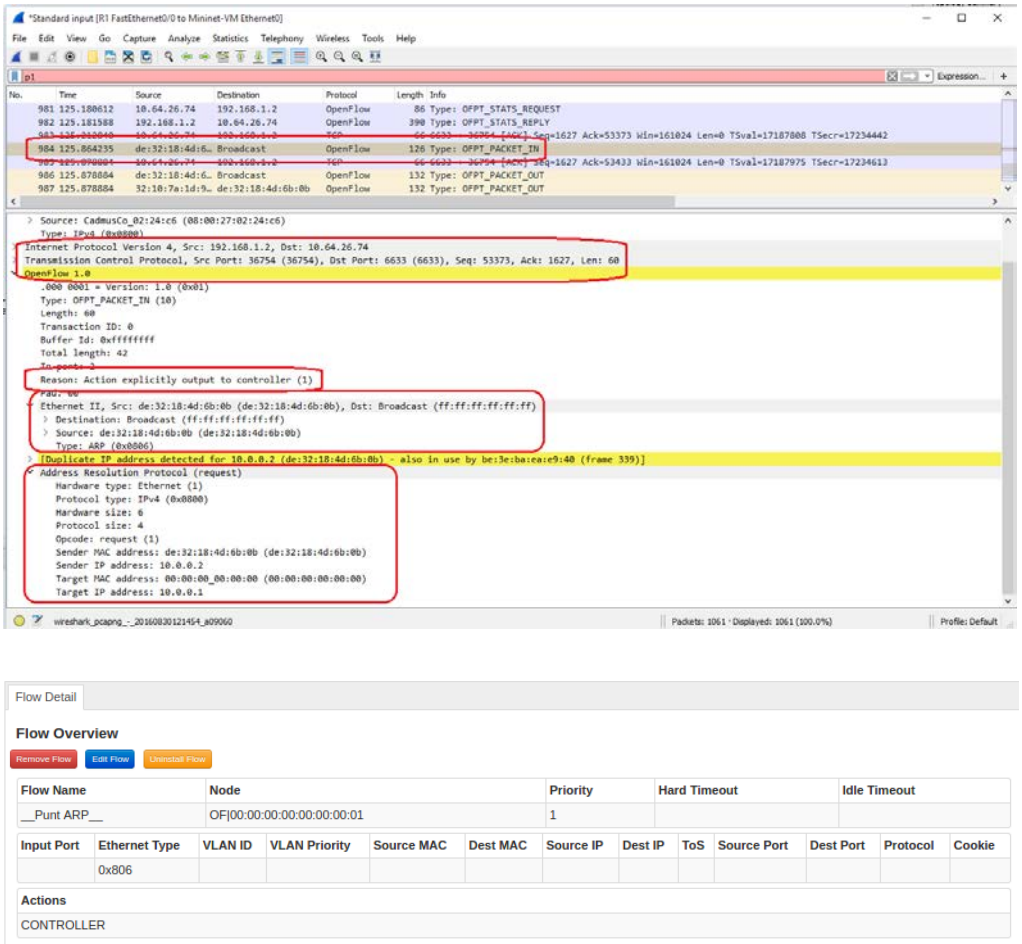
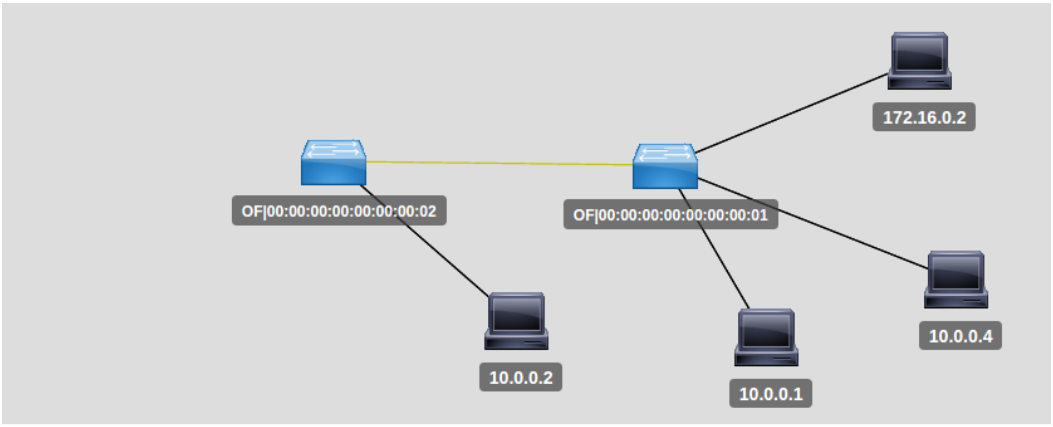


Рис. 7. Коммутатор OpenFlow по умолчанию направляет все пакеты протокола ARP контроллеру ПКС

2. Конфигурирование виртуализованных коммутаторов OpenFlow при помощи внешнего контроллера ПКС.
   a) Настройка таблицы маршрутизации (рис. 8).



Subnet Gateway Configuration | SPAN Port Configuration

	Name	Gateway IP Address/Mask	Ports
☐	1	10.0.0.254/8	eth1 @ OF|00:00:00:00:00:00:00:01 Remove s2-eth2 @ OF|00:00:00:00:00:00:00:02 Remove s1-eth1 @ OF|00:00:00:00:00:00:00:01 Remove s2-eth1 @ OF|00:00:00:00:00:00:00:02 Remove s1-eth2 @ OF|00:00:00:00:00:00:00:01 Remove
☐	2	172.16.0.1/24	eth1 @ OF|00:00:00:00:00:00:00:01 Remove

Рис. 8. Настройка шлюзов подсетей на коммутаторе OpenFlow

б) Настройка межсетевого экрана (рис. 9)

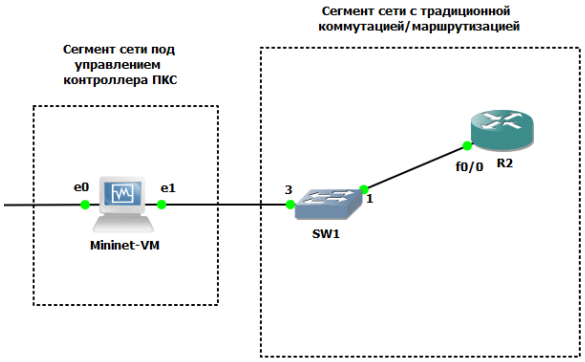


Рис. 9. Маршрутизатор R2 выступает в качестве сервера HTTP и Telnet

Создано правила потока, разрешающие прохождение трафика HTTP (порт 80), и запрещающие весь прочий трафик между узлом H1 и маршрутизатором R2 (рис. 10, 11).

Flow Detail

Flow Overview

Remove Flow | Edit Flow | Uninstall Flow

Flow Name	Node	Priority	Hard Timeout	Idle Timeout
AllowHTTP	OF|00:00:00:00:00:00:00:01	700		

Input Port	Ethernet Type	VLAN ID	VLAN Priority	Source MAC	Dest MAC	Source IP	Dest IP	ToS	Source Port	Dest Port	Protocol	Cookie
2	0x800			9e:31:47:69:ce:c2	cc:01:0d:e8:00:00	10.0.0.1	10.0.0.4			80	TCP	

Actions
OUTPUT=eth1(3)

Рис. 10. Правило потока, разрешающее прохождение трафика HTTP

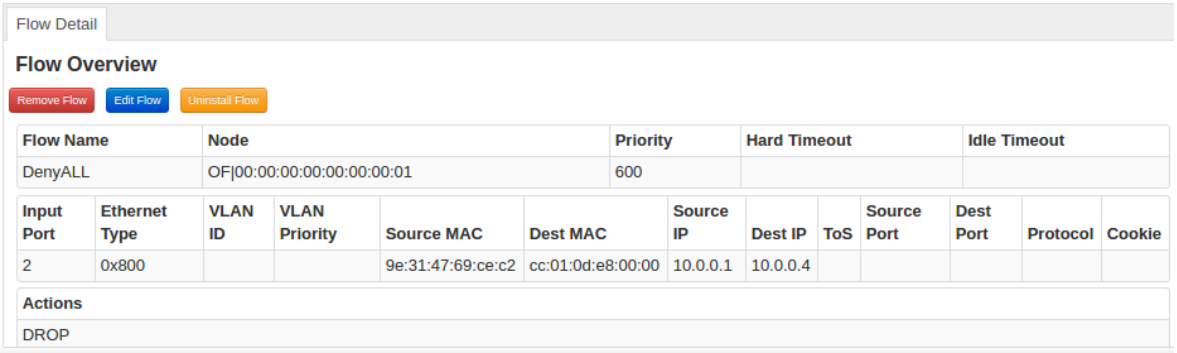


Рис. 11. Правило потока, запрещающее прохождение всего трафика

V. Заключение

На основе существующих подходов по интеграции традиционных сетей и ПКС разработаны тестовые сценарии интеграции ПКС в архитектуру традиционных сетей.

Для отработки тестовых сценариев на практике разработан лабораторный стенд, демонстрирующий возможности совместного использования сетей ПКС и традиционных сетей.

VI. Библиография

[1] Thomas D. Nadeau, Ken Gray. SDN: Software Defined Networks, First Edition, O'Reilly Media, 2013.

[2] Doug Marschke, Jeff Doyle, Pete Moyer. SDN: Anatomy of Openflow, Volume I, 2015.

[3] Эви Немет, Гарт Снайдер, Трент Р.Хейн, Бэн Уэйли. Unix и Linux. Руководство системного администратора, 4-е издание. Вильямс, 2014.

[4] Testing-Interop Working Group. Interoperability Event Technical Issues. Report June 2013. Version 0.4. ONF TR-501. [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-test/ONF_AppFest_Technical_Report_2015.pdf)

[5] Conformance Test Specification For OpenFlow Switch Specification v1.3.4. Basic Single Table Conformance Test Profile. Version 1.0. April 15, 2015. ONF TS-026. [HTML] (https://www.opennetworking.org/images/stories/downloads/working-groups/OpenFlow1.3.4TestSpecification-Basic.pdf)

[6] OpenFlow Switch Specification 1.4.0 (Wire Protocol 0x05). October 14, 2013. ONF TS-012. [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf)

[7] OpenFlow Management and Configuration Protocol (OF-CONFIG 1.2). ONF TS-016 [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.2.pdf)

[8] OpenFlow Table Type Patterns. Version No. 1.0, 15 August 2014. ONF TS-017 [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/OpenFlow%20Table%20Type%20Patterns%20v1.0.pdf)

[9] Migration Use Cases and Methods. Migration Working Group. [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/use-cases/Migration-WG-Use-Cases.pdf)

[10] Google's Inter-Datacenter WAN Using SDN and OpenFlow [HTML] (https://www.opennetworking.org/images/stories/downloads/sdn-resources/customer-case-studies/cs-googlesdn.pdf)

[11] Mininet project URL: http://www.mininet.org

[12] OpenDaylight project URL: http://www.opendaylight.org

[13] GNS3 project URL: http://www.gns3.com

[14] VirtualBox project URL: http://www.virtualbox.org

[15] https://en.wikipedia.org/wiki/Dynamips

Laboratory bench for testing the integration capabilities of SDN networks and traditional networks

Olga R. Laponina, Marat R. Sizov

Abstract **– The article examines possible migration options for software-defined networks (SDN), as well as the requirements for a laboratory bench that allows testing various options for integrating SDN into the architecture of traditional data networks. The main focus is on testing SDN without using real equipment, exclusively in the laboratory, using public and/or commercial software solutions. The emulated segment of the network is implemented on a computer with a OS Windows on which the graphical simulator of the GNS3 network is installed. In the GNS3 environment, virtual copies of the Cisco router, VirtualBox virtual machine with OS Ubuntu/Linux, network switch and two virtual hosts are created. The router in GNS3 is emulated using the Cisco Dynamips router emulator using the real Cisco IOS software image.**

Keywords **– software-defined networking, SDN, SDN Controller, OpenFlow.**