УДК 004.724

Аспекты информационной безопасности архитектуры SDN

А. А. Захаров, Е. Ф. Попов, М. М. Фучко

OpenFlow является наиболее распространенным протоколом архитектуры Software Defined Networking (SDN), которая разделяет функции управления сетевыми устройствами и передачи данных. Она переносит основные функции управления с маршрутизаторов и коммутаторов на централизованные контроллеры. Благодаря централизованному управлению сетью и программируемости контроллеров, архитектура SDN обладает потенциалом, который может быть использован для модернизации средств обеспечения безопасности и реализации более эффективных методов противодействия угрозам, свойственных традиционной архитектуре сетей передачи данных. В работе представлен обзор архитектуры SDN и протокола OpenFlow, анализ угроз и технологии их нейтрализации для архитектуры SDN и протокола OpenFlow, а также определены критичные угрозы для тех сетей OpenFlow, которые в скором времени могут появиться в РФ; предложены способы противодействия данным угрозам.

Ключевые слова: защита информации, SDN, протокол OpenFlow, программно-конфигурируемые сети, информационная безопасность.

1. Введение

Основными движущими причинами развития современных сетевых технологий являются повышенные требования к качеству обслуживания и безопасности непрерывно возрастающего трафика. Традиционные сети передачи данных имеют сложную структуру и трудны в управлении, поскольку для внедрения глобальной сетевой политики приходится отдельно выполнять настройку каждого сетевого устройства, что приводит в том числе к рискам, связанным с некорректной конфигурацией. Отметим, что традиционные функции автоматической реконфигурации крупной сети в случае ее изменения и/или неисправности опираются на механизмы, снижающие производительность сетевой инфраструктуры, поскольку протоколы, реализующие реконфигурацию, повышают нагрузку на сетевые устройства, усложняя и программное, и аппаратное обеспечение.

Термин *Software Defined Networking (SDN*) или *Программно-Конфигурируемые Сети (ПКС)* относительно не новый. Первые работы и исследования по данной тематике появляются начиная с 1995 года [1]. Технология появилась в ответ на потребности распределенных вычислений, использование сетевых технологий в облаках (где в сетях количество маршрутизаторов, которыми надо управлять, может достигать нескольких десятков тысяч), появление крупных дата-центров, начало виртуализации интернета [2]. По сути, это новая архитектура, которая изменяет организацию традиционных сетей передачи данных. В сетях SDN основные функции коммутаторов и маршрутизаторов перенесены на центральный сетевой контроллер, что упрощает и применение сетевых политик, и мониторинг состояния сети. При таком подходе передающие устройства отвечают только за передачу данных, опираясь

на таблицу потоков, которая строится централизованным сетевым контроллером, взаимодействующим с передающим устройством [3].

Традиционно при внедрении новых IT технологий с некоторым запаздыванием обнаруживаются и новые проблемы с информационной безопасностью. В своей работе на основе анализа имеющихся подходов к определению и противодействию угрозам для сетей SDN и протоколу OpenFlow мы определяем направления действий по защите такого рода сетей, которые в недалеком будущем могут появиться в РФ.

2. Архитектура SDN

Архитектура программно-конфигурируемой сети (ПСК, SDN) задается четырьмя основными принципами [4]:

- Функции управления и передачи данных разделены. Сетевое устройство становится простым передающим устройством без функций управления.
- Решения о передаче данных принимаются на основе потоков, а не адреса назначения. Поток представляет собой сочетание полей заголовка пакета, действующих в качестве фильтра, сопоставляющего сетевой трафик с набором инструкций по его обработке. В контексте SDN абстракция потока используется для обеспечения одинаковой обработки и передачи на сетевых устройствах для каждого пакета в последовательности от источника к месту назначения.
- Контроллер SDN, или сетевая операционная система, является внешним объектом для передающих устройств. Контроллер SDN – это программная служба, которая предоставляет ресурсы и интерфейс для разработки программных расширений, которые используют преимущества централизованного управления сетью и ограничены только набором критериев и действий передающих устройств.
- Программное обеспечение, запущенное на сетевом контроллере, взаимодействует с передающими устройствами и открывает возможности программирования функций сетевой инфраструктуры. Это фундаментальное свойство SDN, которое является основным преимуществом архитектуры.

На основании определения архитектуры SDN сетевая инфраструктура делится на три уровня. Уровень передачи данных и уровень управления являются результатом упрощения передающих устройств и переноса функций управления на централизованную программную платформу. Третий уровень (уровень приложений) является абстракцией, тесно связанной с программируемостью сетевого контроллера и его способностью взаимодействовать с другими приложениями. В результате архитектура SDN имеют следующую структуру [5]:

- Уровень передачи данных – передающие устройства, включая как физические, так и виртуальные коммутаторы, которые доступны через унифицированный интерфейс и выполняют инструкции таблиц потоков для передачи сетевого трафика.
- Уровень управления состоит из сетевого контроллера или набора сетевых контроллеров, обеспечивающих функции управления сетевой инфраструктурой. Взаимодействие с устройствами уровня передачи данных – через унифицированный программный интерфейс. На данном уровне есть три интерфейса для взаимодействия с другими элементами сетевой инфраструктуры: южный интерфейс (для уровня передачи данных), северный интерфейс (для уровня приложений), восточный/западный интерфейс (для взаимодействия между элементами уровня управления).
- Уровень приложений состоит из высокоуровневых приложений, которые взаимодействуют с сетевым контроллером или набором контроллеров для реализации конкретных функций, отвечающих требованиям сетевой инфраструктуры.

Взаимодействие между сетевым контроллером и передающими устройствами реализуется посредством программного интерфейса, который используется для прямого управления

группами устройств. Наиболее развитым программным интерфейсом на данный момент является протокол OpenFlow [6]. Архитектура OpenFlow-коммутатора базируется на одной или нескольких таблицах правил, определяющих механизм обработки потоков сетевого трафика. Каждое правило является записью в таблице OpenFlow-коммутатора. Запись сопоставляется с определенным потоком трафика. В зависимости от результата сопоставления применяется соответствующее действие (блокирование, передача, модификация и т.д.) к пакетам из данного потока.

В зависимости от набора правил, установленных сетевым контроллером, коммутатор OpenFlow может выполнять роль маршрутизатора, коммутатора, межсетевого экрана и прочих элементов традиционной сетевой инфраструктуры. Важным следствием использования OpenFlow является гибкость, привнесенная централизованным управлением сетевой инфраструктуры, которая ведет к упрощению создания и модификации конфигурации сетевой инфраструктуры. Помимо этого, ключевым преимуществом архитектуры SDN является возможность расширения функций сети за счет интеграции различных программных модулей (в том числе пользовательских) в сетевой контроллер.

Архитектура SDN открывает ряд возможностей для инноваций в области управления процессом передачи данных и обеспечения информационной безопасности. Сочетание программируемости сети и централизованной точки управления позволяет доработать современные средства и системы защиты данных, расширив их функциональные возможности и повысив производительность [7]. Механизмы захвата трафика и сбора статистики, активированные на сетевых устройствах, позволяют формировать данные, которые регулярно передаются на сетевой контроллер. Приложения, функционирующие на контроллере, могут обрабатывать и анализировать данные, получаемые со всей сети. На основании результатов анализа новые или модифицированные политики могут быть применены на сетевых устройствах. Данный подход может значительно увеличить эффективность контроля и противодействия угрозам безопасности сети.

3. Протокол OpenFlow

OpenFlow является основным используемым стандартом, который определяет процесс взаимодействия между уровнем передачи данных и уровнем управления архитектуры SDN. OpenFlow представляет собой открытый интерфейс для управления передающих устройств сетевым контроллером. Программируемые сетевые контроллеры позволяют устанавливать правила обработки и передачи данных, опираясь на критерии, которые являются редкими для традиционных сетей, такие как действия пользователей, поведение приложений или ресурсов инфраструктуры [8].

OpenFlow первоначально был разработан для сетей Ethernet, но может быть адаптирован для развертывания в сетях других типов. Сетевые устройства могут поддерживать как OpenFlow, так и традиционный метод передачи данных, что значительно упрощает интеграцию протокола OpenFlow в корпоративные системы [9]. OpenFlow широко внедряется производителями сетевого оборудования из-за простой (а следовательно, и более дешевой в реализации) структуры OpenFlow-коммутатора, которая может быть реализована за счет небольших модификаций программного и аппаратного обеспечения. В результате переход на протокол OpenFlow является относительно легким и может быть произведен пошагово с внедрением протокола в те сетевые сегменты, которые требуют функций OpenFlow.

Логическая структура коммутатора OpenFlow базируется на двух основных элементах:

- таблицы (таблицы потоков и таблицы групп), которые используются для сопоставления передаваемых пакетов с действиями, которые необходимо к нему применить;
- интерфейс управления OpenFlow, который используется для взаимодействия с одним или несколькими сетевыми контроллерами [6]. Контроллер через интерфейс управления

OpenFlow создает и обновляет записи в таблицах потоков и отвечает на запросы коммутатора.

4. Уязвимости архитектуры SDN

Архитектура SDN, предполагая существенно иной подход к реализации сетевой инфраструктуры, не лишена потенциальных уязвимостей с точки зрения информационной безопасности. Необходимость разделения доступа сетевых приложений при работе с контроллером, вопросы аутентификации и авторизации при работе приложений с контроллером – это лишь немногие аспекты безопасности, которые приходится принимать во внимание при проектировании SDN-сетей.

Контроллер как ключевой компонент в управлении всей инфраструктурой SDN является наиболее уязвимым элементом, атака на который может повлечь критичные для всей инфраструктуры последствия [10]. Разделение доступа сетевых приложений при их работе с контроллером SDN – актуальная проблема разграничения зон ответственности сетевых приложений. Ситуация, когда любое сетевое приложение способно изменять flow-таблицы любого управляемого данным контроллером коммутатора, не отвечает современным требованиям информационной безопасности. Различные виды приложений требуют различного уровня доступа, и чем более детально описаны ограничения каждого приложения (в соответствии с характером выполняемой задачи), тем более надежной будет сеть. Различные модели разделения доступа могут применяться для решения этой задачи, например, ролевые, мандатные и дискреционные, а также комбинации данных моделей с учетом специфики защищаемой инфраструктуры.

Основными угрозами, возникающими со стороны сетевых устройств, работающих по принципу программно-конфигурируемой сети, остаются вариации таких атак, как «отказ в обслуживании», подмена контроллера и т.д. Перенос «аналитической» компоненты сети на контроллер естественным образом переносит акцент многих атак с сетевого оборудования на обеспечивающее функционирование сети ПО: контроллер сети и сетевые приложения, обращающиеся к контроллеру [11].

Наиболее простым и одновременно эффективным способом нарушения целостности работы сети SDN являются атаки типа «отказ в обслуживании». Опасность атаки следует из самого алгоритма работы SDN-коммутатора при получении неизвестного (т.е. не подходящего под имеющиеся в flow-таблице правила) пакета. В такой ситуации возможны два варианта:

- Пакет целиком отправляется на контроллер для анализа.
- Пакет остается в памяти коммутатора, на контроллер отправляются исключительно заголовки пакета.

Оба способа оставляют для атакующего широкое поле для эффективной реализации отказа в обслуживании путем формирования потока различных пакетов в SDN-сети. Рассмотрим реакцию сети в обоих вышеприведенных случаях:

1. Коммутатор начинает формирование большого количества сообщений для передачи неизвестных пакетов на контроллер. Расходуются процессорные ресурсы коммутатора, увеличивается расход памяти. Особенно сильно расходуется память в том случае, если коммутатор буферизирует сами пакеты и пересылает контроллеру только их заголовки.
2. Поток пакетов от коммутатора на контроллер нагружает канал связи между контроллером и коммутатором. Если среда связи является разделяемой, то снижение оперативности доставки сообщений могут ощутить на себе все коммутаторы. Повышенное влияние на канал связи будет оказано в ситуации, когда коммутатор пересылает пакеты для анализа целиком.

3. Контроллер принимает и обрабатывает поток сообщений, расходуя процессорное время и память своей среды исполнения. Формирование очередей сообщений заставит легитимные сообщения ожидать своей очереди и снизит оперативность принятия решений в сети.
4. Контроллер генерирует поток различных сообщений в ответ на запросы атакованного коммутатора. Расходуются ресурсы канала связи между коммутатором и контроллерами.
5. Коммутатор принимает команды от контроллера и выполняет их, расходуя ресурсы процессора и память. Если команды содержат в себе создание новых правил таблиц потоков, то происходит их лавинообразное увеличение, время проверки каждого нового пакета по таблице увеличивается, растут расходы на обслуживание такой таблицы, а также возможно переполнение таблиц потоков.

В результате реализация атаки может привести к следующим последствиям:

- Исчерпание ресурсов коммутатора. Легитимные пакеты либо вообще не будут обработаны данным сетевым узлом, либо их обработка будет сопровождаться задержками.
- Канал связи между контроллером и коммутатором не обеспечит доставки управляющих сообщений, будучи загруженным потоками данных.
- Контроллер будет перегружен приходящими запросами и не сможет обрабатывать управляющие сообщения, вызванные легитимным трафиком.

5. Уязвимости протокола OpenFlow

Преимущества и недостатки архитектуры SDN для безопасности сетевой инфраструктуры широко изучены. Но оценка уязвимостей архитектуры должна основываться не только на рассуждениях о теоретической архитектуре, но и на экспериментах и результатах внедрения протокола OpenFlow в промышленные сети. В настоящее время определены следующие типы угроз для OpenFlow-сетей.

Режимы работы контроллера (реактивной или проактивный) могут быть легко идентифицированы злоумышленником без применения специфичных подходов и программного обеспечения. Идентификация основана на задержке первого пакета для нового потока трафика и доступна для каждого пользователя, подключенного к сети или использующего сервисы данной инфраструктуры из внешних сетей. В результате некоторые атаки могут использовать конкретный режим работы контроллера [12].

Например, несанкционированная установка правил обработки на коммутаторах, которая приводит к снижению эффективности или нарушению работы сети, проще реализуется в реактивном режиме в связи с особенностью подхода контроллера к управлению таблицами потоков в этом режиме. В то же время выполнение этой атаки на реактивный контроллер возможно, но является более трудной, так как требуется комплексная атака и вероятность быстрого выявления факта атаки значительно выше.

Угрозы безопасности, актуальные для большинства информационных систем, такие как сканирование портов и определение сетевых служб, являются критическими для архитектуры SDN по причине уязвимости канала OpenFlow и наличия большого количества трафика управления, передаваемого между коммутаторами и сетевыми контроллерами. Отметим уязвимость архитектуры SDN к DoS-атакам – одним из самых опасных для архитектуры с централизованной точной управления. Так как передающие устройства не имеют функций управления, они должны иметь стабильное подключение к сетевому контроллеру, чтобы обеспечивать передачу данных. В реактивном режиме контроллера даже короткий период недоступности OpenFlow-контроллера может вызвать много проблем для сетевой инфраструктуры, а долгосрочное блокирование сетевого контроллера может быть использовано, чтобы полностью остановить обработку сетевого трафика или произвести более сложную

атаку. Например, ложный сетевой контроллер может занять место заблокированного, что ставит под угрозу всю сетевую инфраструктуру.

Другой потенциальной проблемой является уязвимость программной инфраструктуры OpenFlow-сети. Возможность программирования сети и наличие открытых программных интерфейсов, используемых для интеграции с сетевым контроллером, открывают еще одну дверь для появления уязвимостей программного обеспечения. В то время как контроллеры OpenFlow, разработанные профессиональными командами и поддерживаемые большими сообществами, теоретически могут быть надежно защищенными, программные модули для контроллера, которые разрабатываются сетевыми инженерами для нужд конкретной сетевой инфраструктуры, могут создать непредсказуемые уязвимости, которые повлияют на безопасность всей сети. Эта проблема безопасности является результатом низкого уровня стандартизации уровней управления и приложения в архитектуре SDN [13].

Еще одна угроза, характерная для традиционной сетевой инфраструктуры, – атаки с подменой – имеет более высокий потенциал для сетей SDN, чем в традиционных сетях. В то время как вся сеть управляется централизованным контроллером, риск подмены менеджмент-трафика в сети OpenFlow достаточно высок. Подмена может повлечь несанкционированный доступ к сетевым устройствам, получение контроллером некорректной статистики или данных о состоянии сети, что может сказаться на работе всей сети.

Одной из причин уязвимости OpenFlow-сетей к атакам подмены является чрезмерная гибкость стандарта OpenFlow. Стандарт позволяет реализовать взаимодействие между сетевым контроллером и коммутаторами на базе протокола TCP без шифрования, а поддержка протокола TLS является необязательной для реализации.

Интересно, что все наблюдаемые угрозы безопасности связаны не со специфичными версиями протокола OpenFlow. Эти угрозы актуальны для всех выпущенных версий протокола (1.0–1.5) и, возможно, не могут быть устранены из-за фундаментальных особенностей архитектуры SDN.

6. Противодействие угрозам информационной безопасности в сетях SDN

В результате того, что централизованное управление сетью доступно не только сетевому контроллеру, но и интегрированным с ним приложениям, архитектура SDN предоставляет возможность как усовершенствовать имеющиеся, так и создавать принципиально новые средства защиты информации. Существует широкий спектр исследований, в которых изучаются методы обеспечения защиты OpenFlow сетей от конкретных, специфичных для OpenFlow-угроз информационной безопасности. Кроме того, совершенствуются средства противодействия традиционным сетевым угрозам, которые можно предотвратить, используя особенности протокола OpenFlow.

Эффективным методом предотвращения атак для OpenFlow-сетей является предварительное выявление несоответствий в конфигурации сети, что может быть достигнуто применением моделей верификации. E. Al-Shaer и S. Al-Haj в работе [14] описан FlowChecker, который использует бинарные диаграммы обнаружения, используя метод разделения сетевых ресурсов на слои. S. Son и др. [15] предлагают проверять политики обработки потоков трафика с использованием модульной теории. Проект VeriFlow реализует политику проверки правил потоков в режиме реального времени. Разработана утилита, перехватывающая правила потоков, посланные контроллером сети на коммутатор OpenFlow с целью дальнейшего анализа этих потоков. C. Schlesinger и др. [16] предлагают технологию проверки изоляции трафика определенного приложения, разделяя сетевую инфраструктуру на слои для достижения конфиденциальности и целостности данных.

Важно отметить, что особенности SDN позволяют усовершенствовать традиционные подходы к контролю потоков трафика. X. Liu и др. [17] предлагают оригинальный подход к

управлению сетью, в котором потоки трафика разрешаются или запрещаются на основе уровней безопасности источника и назначения. G. Yao и др. [18] предлагают сетевую архитектуру с обязательной проверкой адреса отправителя для каждого источника данных. Проверка выполняется сетевым контроллером в реактивном режиме для каждого пакета, который передан на контроллер по причине несоответствия ни одному правилу таблиц потоков. J. Jafarian и др. [19] представляют реализацию Moving Target Defence (MTD) для сетей, основанных на протоколе OpenFlow. MTD реализуется путем сокрытия реальных адресов внутренних хостов для внешних источников. Контроллер назначает виртуальные IP-адреса для сетевых хостов и производит трансляцию в процессе пересылки. Благодаря непредсказуемому распределению виртуальных адресов атака на конкретные элементы инфраструктуры становится значительно более трудоемкой.

В ряде работ исследуются механизмы аутентификации и авторизации на уровне приложений. Предлагаются механизмы, адаптированные для разграничения доступа различных служб и сетевых сервисов. Фреймворк FortNox [20] разрешает конфликты, связанные с получением противоречивых правил из разных источников. Это расширение для контроллера NOX [21], которое проверяет правила потоков перед применением на сетевых устройствах. Shin и др. [22] разработали фреймворк для контроля безопасности OpenFlow-сетей (FRESCO), который позволяет интегрировать в OpenFlow-сети приложения, связанные с обеспечением информационной безопасности. Wen и др. [23] представляют систему контроля PermOF, которая направлена на то, чтобы предоставлять OpenFlow-приложениям минимально необходимые права с целью недопущения злоупотребления.

Поскольку наиболее вероятной областью внедрения SDN-сетей в РФ являются облачные технологии, для которых наиболее серьёзной с коммерческой точки зрения угрозой безопасности является «отказ в обслуживании» (DoS-, DDoS-атаки), то эта тематика заслуживает особого внимания. Принципиально важно отметить, что наличие OpenFlow технологически позволяет использовать SDN-сеть как сенсор, поскольку информация о потоках может быть получена из всех критических и важных точек.

Расширение для контроллера AVANT-GUARD [24] оптимизирует использование ресурсов сети, уменьшая объем управляющего трафика, передаваемого между контроллером и коммутаторами, для смягчения DDoS-атак. J. Suh и др. [25] предлагают сетевую архитектуру с идентификацией приложений для каждого нового потока трафика. Защита от DDoS реализуется ограничением суммарного количества доступных ресурсов для одного приложения для каждого источника данных. C. YuHunag и др. [26] предлагают механизм обнаружения DdoS-атак на основе анализа статистики сетевого трафика. Контроллер анализирует количество передаваемого трафика и частоту определенных событий, которые связаны с DoS-атаками. R. Braga и др. [27] разработали метод анализа трафика данных на основе самоорганизующихся карт, которые позволяют выявлять вредоносные потоки трафика.

Одним из наиболее эффективных методов защиты против DoS-атак в сетях традиционной архитектуры являются системы предотвращения вторжений (IDS). Ряд исследователей предположили архитектуры или реализованные экспериментальные IDS, основанные на принципах работы OpenFlow-сетей. Авторы работы [28] разработали IDS, анализирующую события в OpenFlow-сети с несколькими контроллерами; архитектура разработанной системы позволяет выполнять анализ большого количества потоков трафика, опираясь на правила обнаружения вторжений, сконфигурированных сетевым инженером. NICE [29] представляет собой механизм обнаружения DoS-атак в сетях OpenFlow, построенный на аналитической модели графов атак.

7. Заключение

Архитектура SDN существенно изменяет структуру сети, следовательно, появляются новые угрозы безопасности, вызванные уязвимостями отдельных компонентов инфраструктуры. Кроме того, большинство угроз, связанных с традиционными сетями передачи данных, являются критичными в той же или большей степени в контексте сетей SDN. С другой стороны, SDN архитектура открывает возможности для инноваций в развитии инструментов безопасности. Сочетание централизованного управления и программируемости сети позволяет повысить эффективность средств обеспечения безопасности сети.

Наличие централизованного сетевого контроллера, который является критичным для нормального функционирования сети, влечет уязвимость к DoS-атакам, направленным на контроллер, которые могут привести к отказу всей сетевой инфраструктуры. Помимо этого, атаки сканирования зачастую являются предшественниками направленных DoS-атак, а также могут повлечь использование специфичных уязвимостей программного обеспечения контроллера, что делает их критичными в рамках сетей SDN.

Одним из наиболее эффективных средств противодействия DoS- и скан-атакам в традиционных сетях передачи данных являются системы обнаружения вторжений. Они могут быть применены для обеспечения защиты от угроз, специфичных для SDN, но это потребует адаптации методов анализа и разработки новых правил и паттернов угроз для наиболее эффективного выявления. Однако большинство угроз в сетях SDN унаследованы из традиционных сетей, что подразумевает возможность применения традиционных IDS без значительных модификаций.

Помимо возможности адаптации IDS для защиты от сетевых уроз в контексте SDN, существует возможность модернизации традиционных методов обработки данных и анализа с принесением потенциала данной архитектуры. Система обнаружения вторжений, интегрированная с сетевым контроллером, может использовать функции централизованного контроля для получения большего объема данных, чем в традиционной архитектуре сетей, и имеет больше возможностей для быстрого реагирования на выявленные вредоносные потоки трафика. Модернизация традиционных IDS посредством адаптации для сетей SDN обладает высоким потенциалом, и возможность её реализации должна быть исследована более детально.

Литература

1. *K. Calvert, S. Bhattacharjee, E. Zegura, and J. Sterbenz*, Directions in Active Networks IEEE Communications magazine, p. 72–78, October 1998.
2. *Diego Kreutz, Fernando MV Ramos, P Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig*. Software-defined networking: A comprehensive survey. proceedings of the IEEE, 103[1]:14–76, 2015.
3. *Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner*. Openflow: enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 38[2]:69–74, 2008.
4. *Diego Kreutz, Fernando Ramos, and Paulo Verissimo*. Towards secure and dependable software-defined networks. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, p. 55–60. 2013.
5. *Смелянский Р. Л.* Программно-конфигурируемые сети. Открытые системы. СУБД 9. 2012. с. 23–26.
6. OpenFlow Switch Specification Ver 1.5.1, 2016 [accessed January 11, 2016]. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf.

7. *Никульчев Е. В., Паяин С. В., Плужник Е. В.* Динамическое управление трафиком программно-конфигурируемых сетей в облачной инфраструктуре. Вестник РГРТУ. № 3. 2013. с. 45.
8. *Sandra Scott-Hayward, Gemma O'Callaghan, and Sakir Sezer*. Sdn security: A survey. In Future Networks and Services (SDN4FNS), 2013 IEEE SDN For, p. 1–7. IEEE, 2013.
9. Open Networking Fundation. Software-defined networking: The new norm for networks. ONF White Paper, 2012.
10. *Kevin Benton, L Jean Camp, and Chris Small*. Openflow vulnerability assessment. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, p. 151–152. 2013.
11. *Seungwon Shin and Guofei Gu.* Attacking software-defined networks: A first feasibility study. In Proceedings of the second ACM SIGCOMM work- shop on Hot topics in software defined networking, p. 165–166. 2013.
12. *Diego Kreutz, Fernando Ramos, and Paulo Verissimo*. Towards secure and dependable software-defined networks. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, p. 55–60. 2013.
13. *Margaret Wasserman and Sam Hartman.* Security analysis of the open networking foundation (onf) openflow switch specification. 2013.
14. *Ehab Al-Shaer and Saeed Al-Haj. Flowchecker*: Configuration analysis and verification of federated openflow infrastructures. In Proceedings of the 3rd ACM workshop on Assurable and usable security configuration, p. 37–44. 2010.
15. *Seuk Son, Seungwon Shin, Vinod Yegneswaran, Phillip Porras, and Guofei Gu*. Model checking invariant security properties in openflow. In Communications (ICC), 2013 IEEE International Conference on, p. 1974–1979. 2013.
16. *Cole Schlesinger, Alec Story, Stephen Gutz, Nate Foster, and David Walker*. Splendid isolation: Language-based security for software- defined networks. In Proc. of Workshop on Hot Topics in Software Defined Networking, 2012.
17. *Xiong Liu, Haiwei Xue, Xiaoping Feng, and Yiqi Dai.* Design of the multi-level security network switch system which restricts covert channel. In Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on, p. 233–237. 2011.
18. *Guang Yao, Jun Bi, and Peiyao Xiao*. Source address validation so- lution with openflow/nox architecture. In Network Protocols (ICNP), 2011 19th IEEE International Conference on, p. 7–12. 2011.
19. *Jafar Haadi Jafarian, Ehab Al-Shaer, and Qi Duan.* Openflow random host mutation: transparent moving target defence using software defined networking. In Proceedings of the first workshop on Hot topics in software defined networks, p. 127–132. 2012.
20. *Philip Porras, Seungwon Shin, Vinod Yegneswaran, Martin Fong, Mabry Tyson, and Guofei Gu.* A security enforcement kernel for openflow networks. In Proceedings of the first workshop on Hot topics in software defined networks, p. 121–126. 2012.
21. noxrepo/nox - C++ - GitHub, 2016 [accessed January 11, 2016]. `https://github.com/noxrepo/nox`.
22. *Seungwon Shin, Phillip Porras, Vinod Yegneswaran, and Guofei Gu.* A framework for integrating security services into software- defined networks. Proceedings of the 2013 Open Networking Summit (Re- search Track poster paper), ser. ONS, 13, 2013.
23. *Xitao Wen, Yan Chen, Chengchen Hu, Chao Shi, and Yi Wang.* Towards a secure controller platform for openflow applications. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, p. 171–172. 2013.
24. *Seungwon Shin, Vinod Yegneswaran, Phillip Porras, and Guofei Gu.* Avant-guard: Scalable and vigilant switch flow management in software-defined networks. In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, p. 413–424. 2013.

25. *Junho Suh, H-g Choi, W Yoon, T You, T Kwon, and Y Choi*. Implementation of a content-oriented networking architecture (cona): A focus on ddos countermeasure. In Proceedings of European NetFPGA developers workshop, 2010.
26. *Chu YuHunag, Tseng MinChi, Chen YaoTing, Chou YuChieh, and Chen YanRen.* A novel design for future on-demand service and security. In 2010 IEEE 12th International Conference on Communication Technology, p. 385–388. 2010.
27. *Rodrigo Braga, Edjard Mota, and Alexandre Passito.* Lightweight ddos flooding attack detection using nox/openflow. In Local Computer Networks (LCN), 2010 IEEE 35th Conference on, p. 408–415. 2010.
28. *Yung-Li Hu, Wei-Bing Su, Li-Ying Wu, Yennun Huang, and Sy- Yen Kuo.* Design of event-based intrusion detection system on openflow network. In Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on, p. 1–2. 2013.
29. *Chun-Jen Chung, Pankaj Khatkar, Tianyi Xing, Jeongkeun Lee, and Dijiang Huang.* Nice: Network intrusion detection and countermeasure selection in virtual network systems. Dependable and Secure Computing, IEEE Transactions on, 10[4]:198–211, 2013.

Статья поступила в редакцию 04.02.2016

Александр Анатольевич Захаров
д.т.н., профессор, зав. кафедрой информационной безопасности Тюменского государственного университета, тел. 8-982-920-96-65, e-mail: azaharov@utmn.ru.

Евгений Федотович Попов
аспирант кафедры информационной безопасности Тюменского государственного университета, тел 8-908-874-36-48, e-mail: efpopov@gmail.com.

Михаил Михайлович Фучко
аспирант кафедры информационной безопасности Тюменского государственного университета, тел 8-22-045-64-24, e-mail: mikefgtab@gmail.com.

Alexander A. Zakharov, Evgenii F. Popov, Mikhail M. Fuchko

SDN architecture, cyber security aspects

Open Flow is the most deployed Software Defined Networking (SDN) protocol, which allows us to decouple the control plane and data plane function of a network. This architecture transfers core functions from the routers and switches to centralized controllers. Due to centralized control, SDN architecture has specific capabilities that can be exploited to improve security tools and mitigate threats of traditional architecture networks more effective.
This paper presents an overview of SDN architecture and Open Flow protocol, analysis of related security threats and mitigation techniques. Moreover, the most critical threats for data networks, which may be deployed soon in RF, are determined and mitigation approaches for these threats are proposed.

Keywords: protection of information, SDN, Open Flow protocol, software-configurable network, information security.