ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ТЕЛЕКОММУНИКАЦИИ
СПбГУТ))) Июнь 2017 Том 5 № 2 ISSN 2307-1303 http://www.itt.sut.ru
TELECOM IT
June 2017 Vol. 5 Iss. 2 ISSN 2307-1303 http://www.itt.sut.ru

РАЗРАБОТКА МЕТОДОЛОГИИ ДЕТЕКТИРОВАНИЯ БОТНЕТОВ С ПОМОЩЬЮ SOFTWARE-DEFINED NETWORKING

Г. И. Штеренберг[1*], А. К. Сагдеев[1]

[1] СПбГУТ, Санкт-Петербург, 193232, Российская Федерация
* Адрес для переписки: shterenberg.grigory@yandex.ru

Аннотация
В статье SDN-сети рассматриваются как следующее поколение сетевых технологий. Отделение ControlPlane от сетевых устройств позволяет лучше оптимизировать и то и другое. Контроллер SDN осуществляет контроль над всей сетью и конфигурирует таблицы потоков SDN-коммутаторов, таких как OpenFlow-коммутаторы, посредством программных интерфейсов API. Процесс управления может быть, как реактивным (в ответ на пакеты), так и проактив-ным посредством защищенных каналов. На сегодняшний момент доступно много контроллеров, в зависимости от языка программирования, используемого в них и целей их работы. Например, контроллер NOX основан на языках C++ и Python; контроллеры POX и Ryu работают на Python; а Beacon и Floodlight используют Java.

Ключевые слова
SDN-сети, SDN-коммутатор, контроллер, язык программирования.

Информация о статье
УДК 4.056
Язык статьи – русский.
Поступила в редакцию 28.03.17, принята к печати 02.06.17.

Ссылка для цитирования: Штеренберг Г. И., Сагдеев А. К. Разработка методологии детектирования ботнетов с помощью Software-Defined Networking // Информационные технологии и телекоммуникации. 2017. Том 5. № 2. С. 106–113.

DEVELOPMENT OF METHODOLOGY FOR BOTNET DETECTION USING SOFTWARE-DEFINED NETWORKING

G. Shterenberg[1*], A. Sagdeev[1]

[1] SPbSUT, St. Petersburg, 193232, Russian Federation
* Corresponding author: shterenberg.grigory@yandex.ru

Abstract—In the article, SDN-networks are considered as the next generation of network tech-nologies. Separation of Control Plane from network devices allows you to better op-timize both. The SDN controller monitors the entire network and configures the SDN switch stream tables, such as Open Flow switches, through the APIs. The manage-ment process can be either reactive (in response to packets) or proactive through secure channels. Today many controllers are available, depending on the programming language used in them and the purposes of their operation. For example, the NOX controller is based on C ++ and Python; The POX and Ryu controllers work in Python; Beacon and Floodlight use Java.

Keywords— SDN-network, SDN-switch, controller, programming language.

Article info
Article in Russian.
Received 21.08.16, accepted 22.09.16.

For citation: Shterenberg G., Sagdeev A.: Development of Methodology for Botnet Detection Using Software-Defined Networking // Telecom IT. 2017. Vol. 5. Iss. 2. pp. 106–113 (in Russian).

Проект системы детектирования ботнетов

Архитектура SDN основана на подходе, подразумевающем непосредственное программирование сетей. A SDN-сети рассматриваются как следующее поколение сетевых технологий. Отделение ControlPlane от сетевых устройств позволяет лучше оптимизировать и то и другое. Контроллер SDNосуществляет контроль над всей сетью и конфигурирует таблицы потоков SDN-коммутаторов, таких как OpenFlow-коммутаторы, посредством программных интерфейсов API. На сегодняшний момент [1] доступно много контроллеров, в зависимости от языка программирования, используемого в них и целей их работы. Например, контроллер NOXоснован на языках C++ и Python; контроллеры POXи Ryu работают на Python; а Beaconи Floodlight используют Java.

В традиционных сетях, где сетевые устройства получают трафик от хоста, решения принимаются на основании таблицы маршрутизации, которая есть на каждом устройстве (предварительно сконфигурирована или получена с помощью протокола динамической маршрутизации). Коммутатор, получив новый поток (*flow*) от хоста, пересылает трафик контроллеру и решение по установлению канала связи принимается на основании информации о трафике. Контроллер, приняв решение о том, как направить трафик к месту назначения, настраивает соответствующие правила потока на всех коммутаторах между источником и местом назначения. Все последующие пакеты, относящиеся к одному потоку,

направляются непосредственно в соответствии с правилами потока коммутаторов.

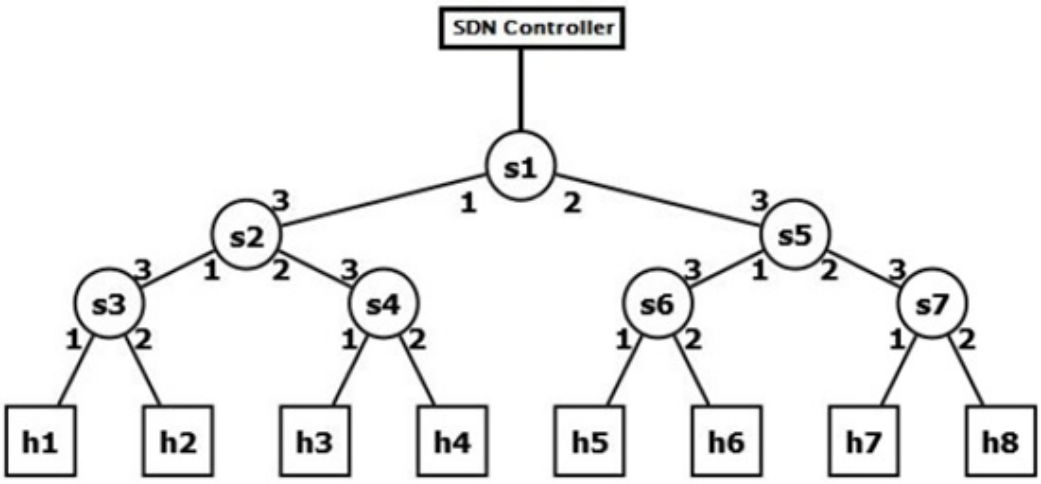

Рис. 1. Пример стандартной топологии SDN

На рис. 1 показана типовая топология SDN, сгенерированная в эмуляторе Mininet, в котором используется контроллер POX. Где *s*1-*s*7 – OpenFlow-коммутаторы; *h*1-*h*8 – персональные компьютеры (хосты).

На рис. 2 показано успешное создание сквозных потоков между всеми хостами.

```
mininet@mininet-vm:~$ sudo mn --topo tree,3 --switch ovsk
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6)
 (s5, s7) (s6, h5) (s6, h6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet>
```

Рис. 2. Создание потоков между хостами *h*1-*h*8

Однако злоумышленники могут использовать установленные потоки между узлами сети для создания атак в таких сетях. Например, скомпрометированные хосты могут использоваться для генерации атак после того, как контроллер создаст потоки. Но в то же время централизованная архитектура SDN может значительно повысить эффективность контроля над сетью и противодействия угрозам информационной безопасности. В данной работе [3] предлагается подход, в котором SDN-контроллер использует такой инструмент как типовые шаблоны для сбора информации о трафике от OpenFlow-коммутаторов и использует эту информацию для детектирования такого вредоносного программного обеспечения как ботнеты.

Модель атаки на SDN-сеть

Рассмотрим приведенную ранее топологию участка SDN-сети с несколькими хостами, которые подключаются с помощью OpenFlow-коммутаторов, а контроллер используется для установления сквозных потоков между хостами.

На рис. 3 коммутатор *s*3, к которому подключен хост *h*1, не знает, как обрабатывать пакеты (в его таблице потоков нет записей, связанных с *h*1). Поэтому *s*3 переадресовывает трафик на контроллер. Контроллер принимает решение о том, как перенаправить трафик к месту назначения, и конфигурирует таблицы потоков в требуемых коммутаторах. Например, контроллеру необходимо настроить таблицы потоков в коммутаторах *s*3, *s*2, *s*1, *s*5 и *s*7, чтобы установить канал связи между *h*1 и *h*8. Любые следующие пакеты от хоста будут напрямую переадресовываться коммутаторами в соответствии с правилами потоков, сконфигурированными контроллером.

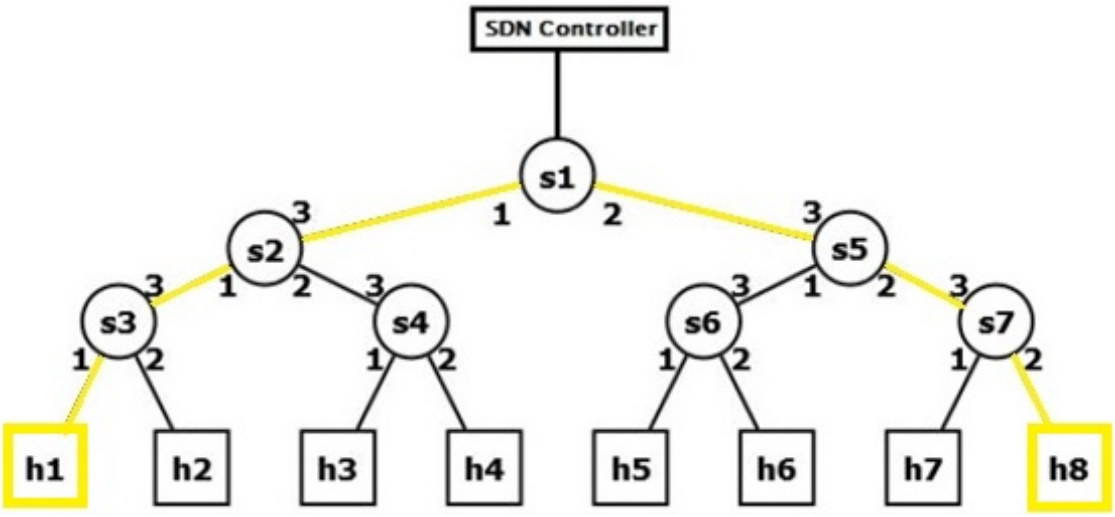

Рис. 3. Создание канала связи между хостами *h*1-*h*8

Вредоносное программное обеспечение оказывает крайне серьезное воздействие на современные сети передачи данных и SDN не является исключением. Среди вредоносных программ ботнеты [4] представляют собой наиболее опасное ВПО, которое может выполнять сложные синхронизированные действия. Поэтому основное внимание в предложенном подходе детектирования ботнетов уделяется борьбе с атаками в области DataPlane.

Наш подход к детектированию ботнетов с помощью SDN

Контроллер SDN является критически важным компонентом, который используется для защиты и управления сетевыми устройствами в SDN. В предлагаемом подходе контроллер конфигурирует OpenFlow-коммутаторы для захвата информации о потоках трафика в соответствии с шаблоном IPFIX и использует эту информацию для обнаружения ботов.

На рис. 4 показана архитектура предлагаемого подхода детектирования ботов. Из этой схемы видно, что в архитектуру подхода входят такие компоненты как типовые шаблоны, коллектор потоков, фильтрация, механизмы обнаружения ботов и предотвращения атак. Типовые шаблоны используются для сбора информации о потоке с использованием IPFIX. Коллектор потоков используется для хранения и организации данных, собранных с помощью типовых шаблонов. Фильтрация используется для уменьшения набора данных за счет исключения информации, не связанной с ботнетами. Механизм детектирования ботов коррелирует информацию о потоке, используя методы машинного обучения, чтобы найти шаблон и с его помощью обнаружить ботов. Механизм предотвращения атак используется для изоляции узлов, скомпрометированных ботом.

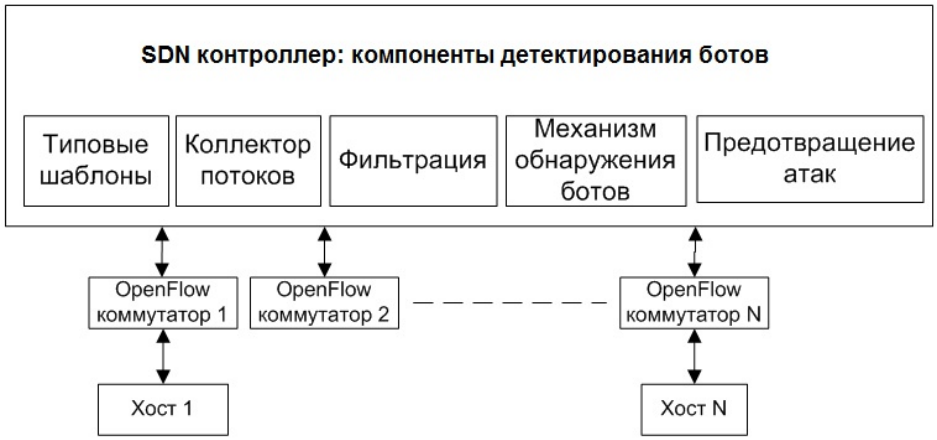

Рис. 4. Схема детектирования ботов с помощью SDN

Как запрограммированная система, ботнет запускает автоматические алгоритмы на протяжении всего своего жизненного цикла, чтобы проводить различные действия. Несмотря на то, что создатели ботнетов прилагают значительные усилия для рандомизации действий ботов, анализ показывает, что создать эффективные шаблоны для их идентификации возможно:

- Некоторые из типов ботов, такие как IRC-, HTTP- и DNS-ориентированные, имеют специфические особенности, когда они реагируют на инструкции от C&C-серверов.
- Такие боты, как SpyEye и некоторые версии Zeus, имеют уникальное поведение потока, которое может использоваться в качестве сигнатуры для обнаружения ботов.
- Обмен данных в одноранговых ботах; к примеру, запросы на обновление и инструкции от ботов создают множество небольших унифицированных пакетов. Кроме того, такие обмены пакетами имеют фиксированный формат и их можно отличить от легитимного трафика.

Проанализировав ряд ботов и объединив все уникальные особенности сетевых потоков, стало понятным, что в теории возможно создать типовой шаблон для обнаружения разных типов ботов. Например, такие функции, как адрес источника и адрес назначения, являются общими для обнаружения любого бота. Следовательно, типовой шаблон должен включать в себя все функции для обнаружения различных семейств ботов. Отметим, что если требуется обнаружить какой-то конкретный бот, то потребуется использовать только определенные функции для детектирования этого бота. Нет необходимости создавать дополнительный типовой шаблон для таких случаев.

Рассмотрим, как типовой шаблон может быть использован для обнаружения различных типов ботов. Для определения типового шаблона используем стандарт IPFIX. В современных условиях у различных поставщиков разработаны собственные проприетарные протоколы, такие как Cisco-NetFlow, Juniper-jflod или с flowd, Huawei-Netstream для захвата потоков трафика. Следовательно, определив типовой шаблон, созданный с помощью IPFIX, его можно будет применять к коммутаторам SDNразных производителей.

Типовые шаблоны должны применяться на OpenFlow-коммутаторах. То есть по замыслу они должны анализировать проходящий через них трафик и передают информацию о потоке трафика коллектору потоков. Все пакеты с унифицированными IP-адресами источника и назначения, портами, протоколом, интерфейсом и классом службы подразделяются на потоки. После чего количество пакетов и количество байт сопоставляются. Эта информация о потоках трафика чрезвычайно важна для анализа вредоносного поведения в сетях с высокими скоростями и большими объемами трафика.

Далее используется фильтрация для минимизации захваченного набора данных за счет устранения избыточности и трафика, не связанного с ботнетами. Например, обычные OpenFlow-коммутаторы могут передавать информацию о сетевых потоках между внутренними и внешними хостами. А шлюзовые OpenFlow коммутаторы могут сообщать информацию о потоках внешних хостов. Поэтому сохраняется только информация о внутренних связях хостов, передаваемая обычными коммутаторами, и информация о связях внешних хостов, передаваемая шлюзовыми коммутаторами.

Механизм обнаружения ботнетов используется для анализа фильтруемого трафика и выявления трафика ботов. Так как типовые шаблоны используются для сбора информации о потоках, то захваченные с их помощью данные содержат необходимую информацию для обнаружения ботов. Следовательно, боты могут быть детектированы путем анализа подмножества функций в потоках сетевого трафика.

Если механизм детектирования ботов определяет, что какой-либо хост скомпрометирован, то механизм предотвращения атак применяет фильтр управления доступом и реконфигурирует OpenFlow-коммутаторы так, чтобы изолировать данный хост от сети.

Разработка элементов системы детектирования ботнетов на основе SDN-контроллера

Шаблоны используются для захвата потоков сетевого трафика с сетевых устройств для обнаружения ботов. Потоки – это набор IP-пакетов, которые имеют

общие атрибуты, такие как IP-адрес источника, IP-адрес назначения, порт источника, порт назначения, тип протокола и интерфейс сетевого элемента. Фиксированный характер захвата IP-потоков ограничивает возможности анализа, пропуская при этом некоторые важные функции из данных потока. IETF относительно недавно представила открытый стандарт для гибкого захватаIP-потоков, названный IPFIX. Таким образом, предлагается использовать атрибуты IP-потоков, которые являются эффективными для детектирования различных типов ботов, для разработки типовых шаблонов. После того как созданы индивидуальные шаблоны, они применяются на сетевых устройствах. Поскольку для создания шаблона используется IPFIX, то шаблон можно использовать к коммутаторам SDN различных производителей.

Заключение

После того как набор данных был выбран для всестороннего анализа с помощью ряда фильтров, описанных на предыдущем этапе, набор данных передается в механизм обнаружения бот-сетей. Первым шагом механизма является классификация или кластеризация потоков в наборе данных с целью выявления сходства потоков. Методы машинного обучения, которые первоначально используются для идентификации специфических особенностей каждого бота, используются на этом этапе для обнаружения конкретных ботов. Существует три основных модели поведения в предлагаемом подходе – поведение бота, поведение ботнета и временное поведение.

В поведении бота мы анализируем потоки, сгенерированные конкретным ботом или машиной, для идентификации его коммуникационных или атакующих графов. В поведении ботнета мы анализируем потоки, сгенерированные группой ботов или машин, для обнаружения активности ботнета. Наконец, при временном или вертикальном методе мы анализируем потоки, сгенерированные ботами или ботнетами, в течение выбранного периода. Как только мы находим сходства или корреляции с такими потоками, мы сравниваем их с шаблонами, которые мы уже идентифицировали посредством машинного обучения.

После того, как механизм обнаружения ботов выявляет хосты, скомпрометированные злоумышленником, необходимо изолировать такие хосты от сети. Компонент «Предотвращение атак» используется для изоляции узлов, зараженных ботом, создавая фильтр контроля доступа. Он динамически настраивает политики управления доступом в коммутаторах OpenFlow, чтобы блокировать весь трафик от скомпрометированного хоста. Таким образом, наша система может обнаружить скомпрометированный хост и изолировать его от других сетевых устройств сети.

Литература

1. Shterenberg S. I., Krasov A. V., Ushakov I. A. Analysis of using Equivalent Instructions at the Hidden Embedding of Information Into the Executable Files // Journal of Theoretical and Applied Information Technology. 2015. Vol. 80. Iss. 1. pp. 28–34.

2. Красов А. В., Левин М. В., Штеренберг С. И., Исаченков П. А. Модель управления потоками трафика в программно-определяемой сети с изменяющейся нагрузкой // Наукоемкие технологии в космических исследованиях Земли. 2016. Т. 8. № 4. С. 70–74.

3. Андрианов В. И., Романов Г. Г., Штеренберг С. И. Экспертные системы в области информационной безопасности // IV Международная научно-техническая и научно-методическая конференция «Актуальные проблемы инфотелекоммуникаций в науке и образовании». 2015. С. 193–197.

4. Сахаров Д. В., Мельников С. Е., Штеренберг С. И. Инфраструктура связи на крайнем севере как база для формирования единой инфосреды // Электросвязь. 2016. № 5. С. 18–20.

5. Штеренберг С. И., Виткова Л. А., Просихин В. П. Методика применения концепции адаптивной саморазвивающейся системы // Информационные технологии и телекоммуникации. 2014. № 4 (8). С. 126–133. URL: http://www.sut.ru/ doci/nauka/review/4-14.pdf

References

1. Shterenberg S., Krasov A., Ushakov I. Analysis of using Equivalent Instructions at the Hidden Embedding of Information Into the Executable Files // Journal of Theoretical and Applied Information Technology. 2015. Vol. 80. Iss. 1. pp. 28–34.

2. Krasov A., Levin M., Shterenberg S., Isachenkov P. Traffic Flow Management Model in Software-Defined Networks with Unequal Load Metric // H&ES Research. 2016. Vol. 8. No. 4. pp. 70–74.

3. Andrianov V., Romanov G., Shterenberg S. Expert Systems in the Field of Information Security // IV International Scientific-Technical and Scientific Methodical Conference "Actual Problems of Education in Science and Education". 2015. pp. 193–197.

4. Sakharov D., Melnikov S., Shterenberg S. The Communication Infrastructure of the Extreme North as a base of for Creating a Common IT-Environment // Elektrosvyaz'. 2016. № 5. pp. 18–20.

5. Shterenberg S., Vitkova L., Prosihin V. Method of Application Idea of Adaptive Self-Developing System // Telecom IT. 2014. Vol. 8. No. 4. pp. 126–133. URL: http://www.sut.ru/doci/nauka/review/4-14.pdf

Штеренберг Григорий Игоревич – студент, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, shterenberg.grigory@yandex.ru

Сагдеев Александр Константинович – кандидат технических наук, доцент, старший преподаватель, майор, СПбГУТ, Санкт-Петербург, 193232, Российская Федерация, uvc_bonch@mail.ru

Shterenberg Grigory – Student, SPbSUT, St. Petersburg, 193232, Russian Federation, shterenberg.grigory@yandex.ru

Sagdeev Alexandr – Candidate of Engineering Sciences, Associate Professor, Senior Lecturer, major, SPbSUT, St. Petersburg, 193232, Russian Federation, uvc_bonch@mail.ru