© Антошина Е. Ю., Чалый Д. Ю., 2017

DOI: 10.18255/1818-1015-2017-6-755-759

УДК 517.9

Семантические средства обеспечения безопасности в программно-конфигурируемых сетях

Антошина Е. Ю.¹, Чалый Д. Ю.²

получена 15 марта 2017

Аннотация. Программно-конфигурируемые сети являются многообещающей технологией построения коммуникационных сетей, в которой управление сетью в отличие от традиционного подхода, основанного на конфигурировании отдельных устройств, представляет собой программу, автоматически задающую конфигурацию сети. Это управляющее программное обеспечение позволяет динамически настраивать маршрутизацию информационных потоков внутри сети в зависимости от требований к качеству обслуживания. Однако при этом возникает задача обеспечения безопасности передачи данных [2, 3], например, чтобы потоки от конфиденциальных узлов не могли достигать открытого сегмента сети. В статье показано, как эта задача может быть решена с использованием семантических средств.

Ключевые слова: безопасность, семантика, программно-конфигурируемые сети

Для цитирования: Антошина Е. Ю., Чалый Д. Ю., "Семантические средства обеспечения безопасности в программно-конфигурируемых сетях", *Моделирование и анализ информационных систем*, **24:6** (2017), 755–759.

Об авторах:

Антошина Екатерина Юрьевна, orcid.org/0000-0003-1081-1758, аспирант, ассистент,
Ярославский государственный университет им. П.Г. Демидова,
ул. Советская, 14, г. Ярославль, 150003 Россия, e-mail: kantoshina@gmail.com

Чалый Дмитрий Юрьевич, orcid.org/0000-0003-0553-7387, канд. физ.-мат. наук, декан факультета информатики и
вычислительной техники

Ярославский государственный университет им. П.Г. Демидова,
ул. Советская, 14, г. Ярославль, 150003 Россия, e-mail: chaly@uniyar.ac.ru

Благодарности:

¹ Работа частично поддержана грантом РФФИ № 17-07-00823-а «Разработка и анализ моделей и алгоритмов адаптивной организации передачи данных в коммуникационных сетях динамической структуры».

² Работа выполнена в рамках инициативного проекта АААА-А16-116070610022-6.

Введение

Программно-конфигурируемые сети (ПКС) — перспективное направление в развитии компьютерных сетей. Управление сетью, отделенное от аппаратной реализации, предоставляет новые возможности для реализации сетевых приложений, используя современные практики разработки программного обеспечения. Программные решения имеют сложную природу, в связи с чем возникает необходимость в дополнительных инструментах верификации программ как со стороны корректности, так и со стороны безопасности. В данной работе представлена модель безопасности ПКС, построенных на базе протокола OpenFlow [5].

1. Модель коммуникационной сети

В работе рассматривается модель коммуникационной сети, в которой сеть состоит из конечных узлов, генерирующих трафик, а также промежуточных узлов, осуществляющих его маршрутизацию. Мы предполагаем, что конечными узлами являются процессы, использующие виртуальные порты транспортного уровня, а промежуточные узлы – это OpenFlow-коммутаторы [5]. В такой модели передача данных может осуществляться внутри отдельного вычислительного узла между выполняющимися на нем процессами, что ставит дополнительные задачи в обеспечении безопасности всей системы. В коммуникационной сети существует выделенный управляющий узел – контроллер, который по защищённым каналам связи взаимодействует со всеми коммутаторами. Контроллер обладает динамически обновляемой информацией о структуре и топологии сети. Основываясь на этой информации, он генерирует и передает команды для каждого коммутатора, решая задачи, поставленные при создании коммуникационной сети.

Коммутатор использует для работы одну или несколько таблиц потоков и групповую таблицу. Таблицы потоков заполняются только на основании полученных от контроллера команд. При помощи этих таблиц происходит обработка и перенаправление пакетов. Таблицы групп в данной работе не рассматриваются, так как могут быть промоделированы при помощи таблиц потоков.

Контроллер отправляет набор команд для коммутатора в ответ на события, которые произошли в сети (например, приход пакета на коммутатор) либо являются событиями окружающей среды. Каждая запись этого списка состоит из набора логических условий (*matchfields*), инструкций (*action*) и счётчиков. Этот набор заносится в таблицу потоков коммутатора.

Набор логических условий *match* накладывает условия на отдельные заголовки пакета, включая Ethernet, IP и TCP заголовки. Весь предикат *match* можно представить как $match = m_{src} \wedge m_{dst}$, где m_{src} и m_{dst} – предикаты, соответствующие условиям для адресов источника и назначения.

Если обрабатываемый пакет удовлетворяет набору логических условий *match*, то для него выполняются инструкции из соответствующего списка команд. В данной модели рассматриваются команды: *Output*, *Drop*, *Set* и *Delete*. Если пакет не соотносится ни с одной из записей таблицы потоков, то в рамках данной модели он передаётся на контроллер.

Предыдущий подход заключался в разработке модели безопасного воздействия в ПКС [6], акцентируя внимание на политике конфиденциальности для ПКС сети. Сделать обоснованный вывод о соблюдении этой политики стало возможно благодаря сформированной формальной системе правил, составляющих семантическую модель [1].

2. Построение семантической модели безопасности

В этой работе мы предлагаем общий принцип, согласно которому можно определить семантики, отражающие различные свойства безопасности. На предмет соблюдения политики безопасности рассматривается список команд, который контроллер в ответ на событие отправляет на коммутатор. Этот упорядоченный набор записей в

рамках данного подхода можно представить как пары $match \times action$. Само событие определяет первую команду в списке, называемую контекстом обработки пакета.

Для решения этой задачи необходимо построить множество правил задания типов S , согласно которому устанавливается, какой уровень безопасности должен быть назначен списку команд.

Для определения уровней безопасности для потоков данных задаются уровни безопасности конечных узлов сети, которые представляют пользовательские процессы. Каждому конечному узлу p ставится в соответствие уровень безопасности (высокий, $p : high$, или низкий, $p : low$). Мы предполагаем, что уровень безопасности всех узлов сети известен контроллеру и может быть определен, например, при помощи протокола аутентификации.

Поскольку отдельные записи таблиц потоков в коммутаторах ПКС-сети могут перенаправлять трафик между отдельными подсетями, которые состоят из многих узлов, для определения того, на какие потоки влияют эти записи, мы введем предикат $forall$, позволяющий определить уровень безопасности множества потоков.

Начнем формирование множества правил задания типов S : внесём в него правила типизации предиката $forall$, который помогает определять свойства, связанные с уровнями безопасности множеств узлов:

$$\frac{\{p_1 : low, \dots, p_n : low\}}{\vdash forall(p_1, \dots, p_n) : low}, \quad (1)$$

$$\frac{\{p_1 : high, \dots, p_n : high\}}{\vdash forall(p_1, \dots, p_n) : high}. \quad (2)$$

Таким образом подсеть в целом типизируется в соответствии с типом безопасности узлов, которые в нее входят. Если в подсеть входят узлы разных типов, то типизировать предикат $forall$ нельзя. Это позволяет жестко разделить потоки данных [4], относящиеся к разным уровням безопасности, хотя и ограничивает практическую применимость нашего метода. Впоследствии мы покажем, как можно ослабить эти требования с целью большей практичности.

Продолжим формирование множества S правилами, которые справедливы для желаемой политики безопасности. В это множество добавляются правила определения типа безопасности для каждого возможного события. Тип безопасности события определяет контекст, в котором происходит установка команд. Чтобы задать политику безопасности сети, для каждой команды необходимо определить, над какими информационными потоками возможно её применение в каждом контексте безопасности событий.

Возьмём абстрактную команду и определим для неё политику безопасности: для каждого контекста и информационного потока укажем, нарушает или соблюдает данная команда требуемое свойство безопасности.

Теперь можно определить семантические правила для этой команды. Если в контексте безопасности $[C]$ разрешен поток типа $t_{src} \rightarrow t_{dst}$, то правило принимает вид:

$$\frac{\vdash forall(src(match)) : t_{src} \quad \vdash forall(dst(match)) : t_{dst}}{[C] \vdash match \times action}. \quad (3)$$

Имеет смысл попытаться представить эти правила в более ёмком виде. Для этого предлагается построить дерево решений, которое можно редуцировать, и использовать для построения более компактного множества S . Полученные правила вывода для каждой команды добавляются в множество S .

Также в это множество необходимо добавить правило, позволяющее типизировать композиции команд:

$$\frac{[pc] \vdash A \quad [pc] \vdash B}{[pc] \vdash A; B}.$$

Совокупность правил из полученного множества S представляет собой формальную систему безопасности для ПКС, с помощью которой можно сделать вывод о соблюдении или нарушении заданной политики безопасности: если весь список команд может быть типизирован при помощи представленной системы, то данный список соблюдает установленную политику безопасности.

Заключение

Данная работа предлагает подход, основанный на разработанной формальной модели безопасности ПКС. Сформированная согласно представленному алгоритму система гарантирует, что приложение на контроллере не нарушает требуемую политику безопасности. После создания модели предлагается расширение: введение правила деклассификации, которое увеличивает её практическую ценность. Система безопасности может быть реализована как программный модуль контроллера, который будет проверять, не нарушают ли приложения, запускаемые на контроллере, свойства безопасности.

В качестве дальнейших направлений исследований интересно рассмотреть проблему безопасности ПКС в более комплексном виде. Поскольку контроллер может реагировать на события внешнего мира и является сложным программным средством, то перспективным подходом является разработка семантической системы безопасности, которая учитывает также поток управления контроллера.

Список литературы / References

- [1] Sabelfeld A, Myers A.C., “Language-Based Information-Flow Security”, *IEEE Journal on Selected Areas in Communications*, 2003, № 21, 5–19.
- [2] Smeliansky R.L., “SDN for Network Security”, *Proc. of Int. Conf. "Modern Networking Technologies (MoNeTec)*, 2014, 86–95.
- [3] Casado M., Foster N., Guha A., “Abstractions for Software-Defined Networks”, *Communications of the ACM*, **57**:10 (2014), 86–95.
- [4] Foster N., Freedman M.J., Monasanto Ch., Rexford J., Story A., Walker D., “Splendid Isolation: A Slice Abstraction for Software-Defined Networks”, *CM Int. Conf. on Functional Programming*, 2011, 279–291.
- [5] McKeown N., Anderson T., Balakrishnan H., Parulkar G., Kozen D., Peterson L., Rexford J., Shenker S., Turner J., “OpenFlow: Enabling Innovation in Campus Networks”, *SIGCOMM Computer Communications Review*, **38**:2 (2008), 69–74.

- [6] Antoshina E.Ju., Nikitin E.S., Chalyy D.Ju., Sokolov V.A., "End-to-end Information Flow Security Model for Software-Defined Networks", *Modeling and Analysis of Information Systems*, **22**:6 (2015), 735–749.

Antoshina E. Ju., Chalyy D. Ju., "Semantic Security Methods for Software-Defined Networks", *Modeling and Analysis of Information Systems*, **24**:6 (2017), 755–759.

DOI: 10.18255/1818-1015-2017-6-755-759

Abstract. Software-defined networking is a promising technology for constructing communication networks where the network management is the software that configures network devices. This contrasts with the traditional point of view where the network behaviour is updated by manual configuration uploading to devices under control. The software controller allows dynamic routing configuration inside the net depending on the quality of service. However, there must be a proof that ensures that every network flow is secure, for example, we can define security policy as follows: confidential nodes can not send data to the public segment of the network. The paper shows how this problem can be solved by using a semantic security model. We propose a method that allows us to construct semantics that captures necessary security properties the network must follow. This involves the specification that states allowed and forbidden network flows. The specification is then modeled as a decision tree that may be reduced. We use the decision tree for semantic construction that captures security requirements. The semantic can be implemented as a module of the controller software so the correctness of the control plane of the network can be ensured on-the-fly.

Keywords: security, semantics, software-defined networks

On the authors:

Ekaterina Ju. Antoshina, orcid.org/0000-0003-1081-1758, graduate student,
P.G. Demidov Yaroslavl State University,
14 Sovetskaya str., Yaroslavl 150003, Russia, e-mail: kantoshina@gmail.com

Dmitry Ju. Chalyy, orcid.org/0000-0003-0553-7387, PhD,
P.G. Demidov Yaroslavl State University,
14 Sovetskaya str., Yaroslavl 150003, Russia, e-mail: chaly@uniyar.ac.ru

Acknowledgments:

¹ The work was partially supported by the RFBR grant № 17-07099823-a.

² This work was partially supported by the initiative project AAAA-A16-116070610022-6.