***Д.А. Паккерт, Е.А. Черкашин***

Иркутский научно-исследовательский технический университет, Иркутск, Россия

Институт динамики систем и теории управления им. В.М.Матросова СО РАН, Иркутск, Россия

**РЕАЛИЗАЦИЯ УСТАНОВЛЕНИЯ СОСЕДСКИХ ОТНОШЕНИЙ В ПРОТОКОЛЕ OSPFv3**

***Аннотация.*** *Рассматривается задача синтеза части функций протокола OSPFv3 при помощи декларативных спецификаций YANG структур данных и P4 процессов приема, обработки и порождения пакетов IPv6, а также моделирования состояния маршрутизаторов. Представлены спецификации YANG и P4 для процесса установления соседских отношений.*

***Ключевые слова.*** *OSPFv3, динамическая маршрутизация, IPv6, сети ЭВМ*

С 2011 года произошел качественный переход от интернета людей к интернету вещей. Быстрый рост количества сетевых устройств обеспечивается протоколом IPv6, разработанным в 1996 году, когда стало понятно, что количества адресов IPv4-диапазона будет недостаточно. IPv6 значительно упрощает регистрацию устройств в сети. Конфигурирование сегментов (подсетей) таких устройств на практике требует внедрение автоматизации управления маршрутизацией пакетов между сегментами, т.е. *динамической маршрутизации* (ДМ) пакетов, что значительно упрощает администрирование сетей.

Другим направлением развития современных сетей передачи данных являются *программно-определяемые* *сети* (ПОС), одна из форм форм виртуализации сети. В ПОС (англ. *Software-defined networking, SDN*) [5] уровень управления сетью отделён от устройств передачи данных и реализуется программно. Ключевые принципы ПОС — это разделение процессов передачи и управления данными, централизация управления сетью при помощи унифицированных программных средств, виртуализация физических сетевых ресурсов. ПОС — это основа инфраструктуры передачи данных облачных сервисов в виртуальных сетях организаций.

Существует множество протоколов ДМ как для сетей IPv4, так и для IPv6. OSPFv3 – протокол ДМ класса “по состоянию канала”, основанный на идеях протокола OSPFv2 (IPv4), но предназначенный для сетей IPv6. Протоколы данного класса основаны на хранении маршрутизатором информации о каналах связи (link) с другими маршрутизаторами: названия маршрутизатора и соответствующей подсети, метрику расстояния канала, приоритет и т.д. Протоколы “по состоянию канала” так же учитывают пропускную способность всех каналов до сети назначения и, основываясь на наилучшей пропускной способности, строят оптимальный маршрут. На основе анализа структуры сети производится изменение таблиц маршрутизации узлов сети.

Для построения таблицы маршрутизации в протоколах OSPFv2 и OSPFv3 применяется модификация алгоритма Э.Дейкстры, позволяющая вычислять кратчайший путь между маршрутизаторами (узлами графа) в условиях неполной информации о структуре нагруженного графа. Стоимость дуг графа (метрика расстояния между маршрутизаторами по общему каналу) оценивается как динамически, так и задается в конфигурации. Каждое устройство строит путь до адреса назначения самостоятельно. При разделении сети на области граф OSPF (информация о его структуре) ограничивается одной областью. Разделение сети на области позволяет улучшить масштабируемость OSPF, иерархически объединяя области; оптимизировать использование вычислительных ресурсов маршрутизаторов, фиксируя размер известной части графа.

В данной работе протокол OSPFv3 реализуется при помощи декларативных языков P4 и YANG, при этом на традиционном для программирования демонов маршрутизации языке С разрабатываются только алгоритмы на графах и взаимодействие с операционной системой. Это позволяет задать (запрограммировать) как уровни управления сетью в устройствах передачи данных (P4), так и задать в рамках некоторого отраслевого стандарта структуры данных для реализации алгоритмов планирования маршрутов (YANG). Это, в свою очередь, позволит продвинуться в решении проблемы с функционированием OSPFv3 в пакетах FRRouting и Quagga. Решаются следующие задачи:

1. Представление уровня управления ПОС/Маршрутизатором;
2. Анализ существующих спецификаций YANG для протокола OSPFv3;
3. Реализация уровней L2, L3 в среде P4;
4. Реализация алгоритмов и интерфейса c уровней L2, L3 к алгоритмам и операционной системе.

**Задача установления соседских отношений**. Протокол маршрутизации OSPF решает задачу управления сетевым трафиком за несколько этапов: а) установление соседских отношений (neighbor state) с маршрутизаторами (узлами), б) распространение информации о структуре сети, в) построение оптимальных маршрутов на графе, г) внесение изменений в таблицы маршрутизации узлов сети. В данном докладе рассмотрим реализацию установления соседских отношений.

В процессе установления соседства узел проходит следующие состояния:

*Down*, начальное состояние OSPF-соседа. Для обнаружения соседей рассылаются специальные hello-пакеты, сообщающие, что с данной стороны канала существует OSPF-маршрутизатор. Данное состояние означает, что от соседей не было получено никакой информации (hello-пакетов).

*Init* указывает, что маршрутизатор получил hello-пакет от соседа, но идентификатор RouterID соседа не был включен в этот hello-пакет, т.е. сосед не знал еще о существовании данного узла. В ответ на сообщение посылается hеllo-пакет соседу, но при этом теперь добавляется идентификатор соседа.

*2-Way -* между маршрутизаторами установлено двунаправленная связь, т.е. оба соседа получили hello-пакеты друг друга с соответствующими идентификаторами. Такое состояние устанавливается, когда узел, принимающий hello-пакет, видит свой собственный RouterID в приходящем hello-пакете. В этом состоянии маршрутизатор решает, будет ли через этот узел маршрутизироваться трафик. В широковещательной (broadcast) среде или в среде точка-мультиточка (point-to-multipoint), маршрутизатор переходит в состояние *Full* только с основным роутером (Designated Router, DR) и с резервным (Backup Designated router, BDR). Со всеми другими соседями он остается в состоянии *2-Way*. Если в сети нет DR- или BDR-узлов, то запускается процедура выбора DR и BDR среди соседних узлов.

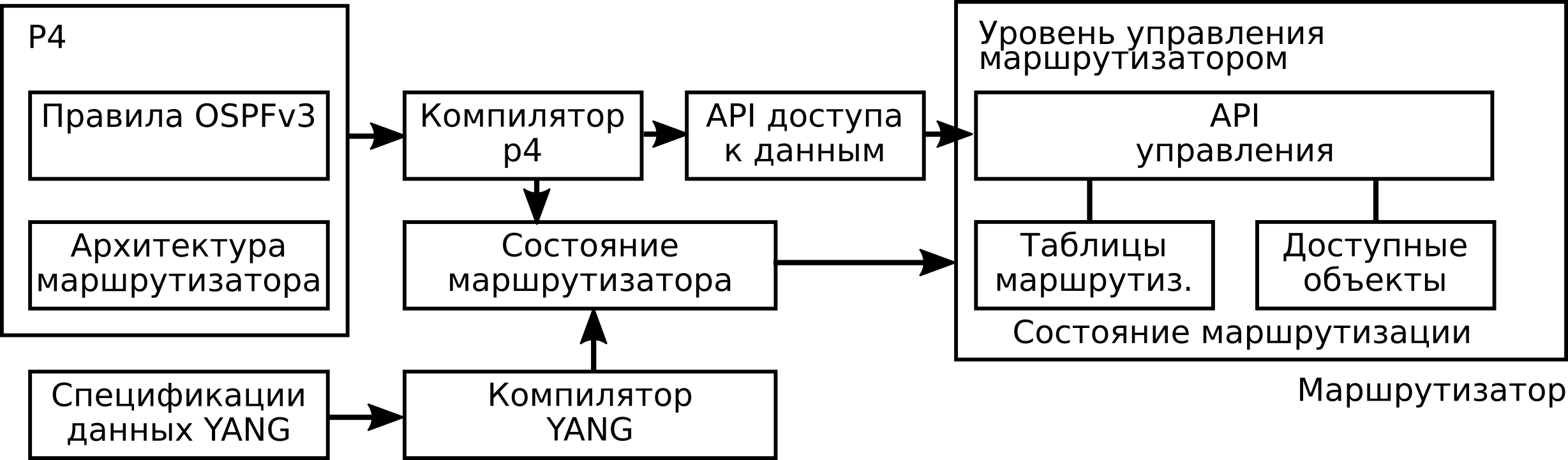
*Exstart.* Как только выбраны DR и BDR, между маршрутизаторами и их DR и BDR начинается процесс обмена информацией о состоянии каналов. В этом состоянии, узлы и их DR и BDR устанавливают отношения ведущий-ведомый, создают соединение как пронумерованную последовательность пакетов.

*Exchange*:OSPF-маршрутизаторы обмениваются DBD-пакетами (дескрипторами базы данных). BDB содержат только заголовки LSA (объявления о состоянии канала) и описывают содержимое базы данных о состоянии каналов. Содержимое принимаемого DBD-пакета обновляет информацию в базе данных узла.

*Loading* -происходит непосредственно обмен информацией о состоянии канала. Основываясь на информации полученной в DBD-пакетах, маршрутизаторы посылают пакеты-запросы о состоянии канала. Затем сосед предоставляет запрошенную информацию в пакетах-обновлениях.

*Full* - маршрутизаторы являются полностью связанными друг с другом. Узлы обменялись сетевыми и маршрутными LSA, и их базы данных полностью синхронизированы. В этом состоянии модель (топология) сети известна всем маршрутизаторам. Узлы достигают состояния Full только со своими DR и BDR. С остальными соседями состояние остается 2-Way.

**Использование декларативных спецификаций**. Предлагаемая реализация протокола базируется на использовании декларативных спецификаций структур данных YANG, опубликованных в Интернет, а также реализации динамической модели протокола при помощи языка P4. На рисунке 1 представлена технология синтеза модулей, реализующих протокол OSPFv3 в предлагаемом подходе.



**Рис. 1. Синтез структур данных логического уровня управления маршрутизатором**

Ключевым блоком в схеме является “Состояние маршрутизатора” - набор объектов, представляющих состояние узла как элемента структуры сети (высокий уровень представления). Состояние маршрутизатора интерпретируется (загружается) в виде записей таблицы маршрутизации и состояния доступных (external) объектов маршрутизатора (низкий уровень представления). Интерпретация осуществляется при помощи запросов к API уровня управления маршрутизатором.

Структуры данных “Состояния маршрутизатора” формируются из спецификаций YANG и структур данных “Правил OSPFv3”: модель динамики маршрутизатора задается в виде правил “Если … то …”, где с левой стороны правила задается условие на свойства входного пакета, а справа - некоторое действие, например, переход в новое состояние. Для формирования “Копилятором p4” “API доступа к данным” необходимо задать на языке p4 “Архитектуру маршрутизатора”. Алгоритм и API SPF реализуются на языке C и в схеме находятся в блоке “Уровень управления маршрутизатором”. Реализация алгоритма создается на основные функции модуля ospf6d пакета FRRouting, обеспечив взаимодействие с p4 и структурами YANG.

Приведем пример спецификации IETF структуры - “состояние маршрутизатора” в процессе установки соседских отношений.

|  |  |
| --- | --- |
| grouping neighbor-state {  description  "OSPF neighbor state.";  leaf address {  type inet:ip-address;  config false;  description "Neig. IP";  } | leaf dr-router-id {  type rt-types:router-id;  config false; }  leaf dr-ip-addr {  type inet:ip-address;  config false;  description "Neig's DR IP.";  } … } |

Спецификация P4 должна включать архитектуру маршрутизатора (перечень физических и виртуальных портов), API уровня управления, процедуры трансляции/генерации заголовков пакетов и набор правил обработки пакетов согласно их заголовкам. Приведем пример правил для нашей задачи (типы данных опущены).

|  |  |
| --- | --- |
| control neighbor\_state(…h, …inp, …err, …out, …rou) {  table down\_hello {  key = {h.ospf.nei.id==none;  rou.st == DOWN;}  actions = {set\_hello;}}  table init\_hello {  key = {h.ospf.nei.id!=none;  rou.st == DOWN}  actions = {set\_hello;  set\_nrID;}} | action set\_hello{  out.ospf.my.id =  API.get\_my\_id();}  action set\_nrID {  out.ospf.nei.id=rou.nei.id;  rou.st = INIT;}  apply {  down\_hello.apply();  init\_hello.apply();  … } … } |

Аналогичным образом на P4 задается весь перечень функций маршрутизатора. В примере переменная API задает интерфейс к системным функциям узла, в частности, функциям вычисления контрольных сумм данных, доступа к тексту конфигурации, а также алгоритму вычисления кратчайших путей.

Таким образом, в докладе представлены основные идеи подхода к реализации этапа установления соседских отношений в проектировании программного обеспечения ДМ на основе протокола OSPFv3 для сетей IPv6; показаны примеры спецификаций структур данных и части автомата, моделирующего состояние маршрутизатора и его соседей, а также архитектура процесса проектирования средств ДМ. Следующей задачей, решаемой в данном проекте, является адаптация компилятора P4 к API алгоритмов вычисления кратчайших расстояний Э.Дейкстры.

Библиографический список

1. [OSPF for IPv6 - RFC 5340](https://tools.ietf.org/html/5340), URL: https://tools.ietf.org/html/5340
2. [P4: Programming Protocol-IndependentPacket Processors](https://www.sigcomm.org/sites/default/files/ccr/papers/2014/July/0000000-0000004.pdf) // ACM SIGCOMM Computer Communication Review, Vol. 44, No 3, July 2014.
3. YANG Catalog. URL: <https://yangcatalog.org/contribute.html>
4. Программно-определяемая сеть - Википедия [Электронный ресурс] URL:[https://ru.wikipedia.org/wiki/Программно-определяемая\_сеть](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE-%D0%BE%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%8F%D0%B5%D0%BC%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C). (дата доступа - 05.05.2019)