Глава 3

Глобальная система маршрутизации и передачи данных

> В недалеком будущем мы будем рассматривать Интернет как набор автономных систем.
>
> *RFC 827[1], октябрь 1982 г.*

Принципы маршрутизации данных в Интернете

Когда говорят о маршрутизации данных, обычно имеют в виду процесс определения маршрута пакетов — от источника к получателю. В Интернете этот маршрут состоит из множества участков: каждый из маршрутизаторов по пути предполагаемого следования данных определяет его следующий сегмент — какому из соседних маршрутизаторов переслать пакет. Это решение принимается каждым маршрутизатором для каждого пакета, поскольку Интернет — это сеть коммутации пакетов, а не каналов.

Картину глобальной связности маршрутизатор создает самостоятельно, обмениваясь информацией о топологии сети со своими соседями — какие сети подключены непосредственно к маршрутизатору, а какие достижимы через другие соседние узлы. Для обмена этой информацией применяются протоколы маршрутизации, например BGP, который используется в глобальной инфраструктуре.

---

[1] RFC 827: Exterior Gateway Protocol (EGP), URL: https://www.rfc-editor.org/rfc/rfc827

Такая архитектура позволяет автоматически реагировать на изменения топологии Интернета — выход из строя каналов, отдельных узлов или целых сетей. Это также означает, что поток данных может вдруг изменить свой путь вследствие изменения топологии сети (например, выхода из строя одного из транзитных узлов). Все это делает глобальную инфраструктуру Интернета устойчивой и адаптивной.

За последние три декады своего существования глобальная система маршрутизации выросла неимоверно — от 15000 маршрутов NSFNET до более миллиона маршрутов (включая IPv6) в 2023 году (рис. 35).

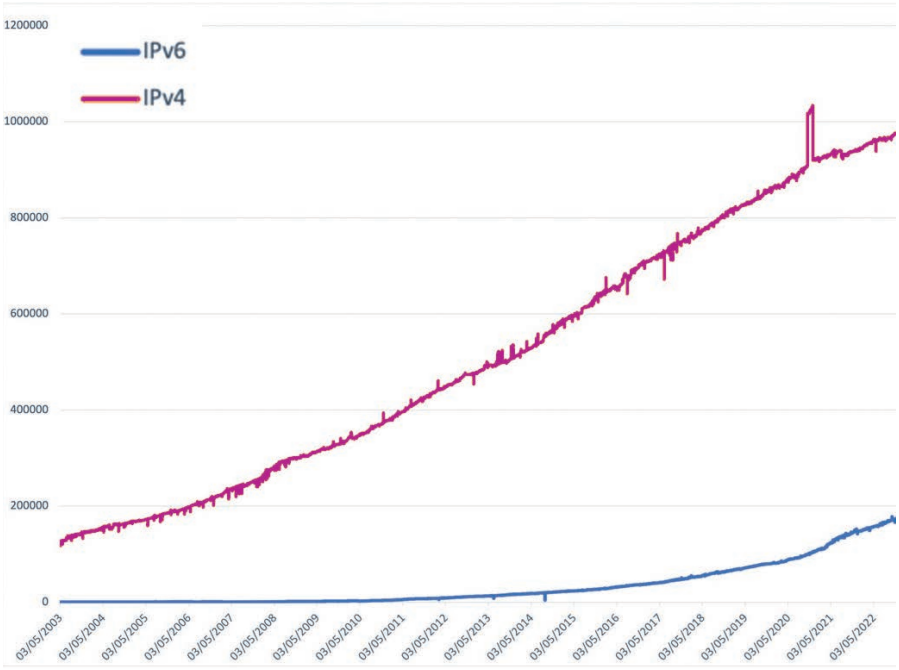

Рис. 35. Рост глобальной таблицы маршрутизации.
Источник данных: «BGP Analysis Reports» http://bgp.potaroo.net/index-bgp.html

Тем не менее, как это ни странно, основные протоколы и архитектура маршрутизации в Интернете почти не изменились. И удивительно, что, несмотря на столь стремительный рост, устойчивость и производительность системы по-прежнему соответствуют потребностям. Как заметил научный руководитель APNIC Джеф Хьюстон (Geoff Huston), это частично может быть объяснено тем, что «рост Интернета в большей степени означает увеличение плотности топологии, нежели ее размера». В качестве свидетельства он приводит относительно постоянную среднюю длину пути — среднее

число сетей, через которые проходит трафик от отправителя к получателю. На протяжении последнего десятка лет он составляет около четырех сегментов[2].

Это, впрочем, не означает, что в глобальной системе маршрутизации и ее основном протоколе BGP отсутствуют проблемы. Напротив, острота вопросов безопасности, обеспечения качества и эффективной конфигурации сети только усиливается.

Но об этом чуть позже. Сначала поговорим об истоках сегодняшней инфраструктуры и основных принципах маршрутизации в Интернете.

Междоменная маршрутизация: от EGP до BGP

Маршрутизация между сетями в Интернете осуществляется с помощью протокола BGP[3]. Суть его в том, что каждая сеть получает от своих соседей — пиров — информацию о связности, а именно — через какую цепочку сетей доступен конкретный префикс. Каждая сеть обрабатывает эту информацию в соответствии с собственной политикой, выбирая для каждого доступного префикса наилучший путь. Выбор этот основан на нескольких параметрах, основным из которых является длина пути. Таким образом, сеть формирует свое «видение» Интернета — и, в свою очередь, также сообщает о нем своим пирам.

Здесь следует отметить, что глобальная маршрутизация осуществляется не между отдельными компьютерами или маршрутизаторами, а между сетями, точнее, доменами маршрутизации. Эти домены (отсюда термин «междоменная маршрутизация» — Inter-Domain Routing, IDR) также называются автономными системами (Autonomous System, AS). Они представляют собой совокупность узлов, находящихся под единым административным контролем и имеющих согласованную политику маршрутизации. Этот аспект является одним из фундаментальных архитектурных принципов Интернета. Он позволяет эффективно разделить глобальную систему маршрутизации на области с четким внутренним контролем, относительно высокой степенью безопасности — и на «межобластное» пространство, основанное на кооперации и взаимодоверии.

Протокол BGP пришел на смену раннему протоколу маршрутизации EGP (Exterior Gateway Protocol), предложенному в 1982 году, когда тогдашний Internet (Catenet) начал расширяться, задействовав не только ARPANET, но и другие опорные сети. Добавление все новых и новых сетей несло в себе риск несовместимости и нестабильности, вызванных различными версиями протокола маршрутизации. Также в новой топологии появились транзитные сети — они передавали трафик, источник и получатель которого находились в других сетях. Протокол EGP

---

[2] См. статью «Update on AS Path Lengths Over Time» https://labs.ripe.net/Members/mirjam/update-on-as-path-lengths-over-time

[3] Border Gateway Protocol, https://ru.wikipedia.org/wiki/Border_Gateway_Protocol

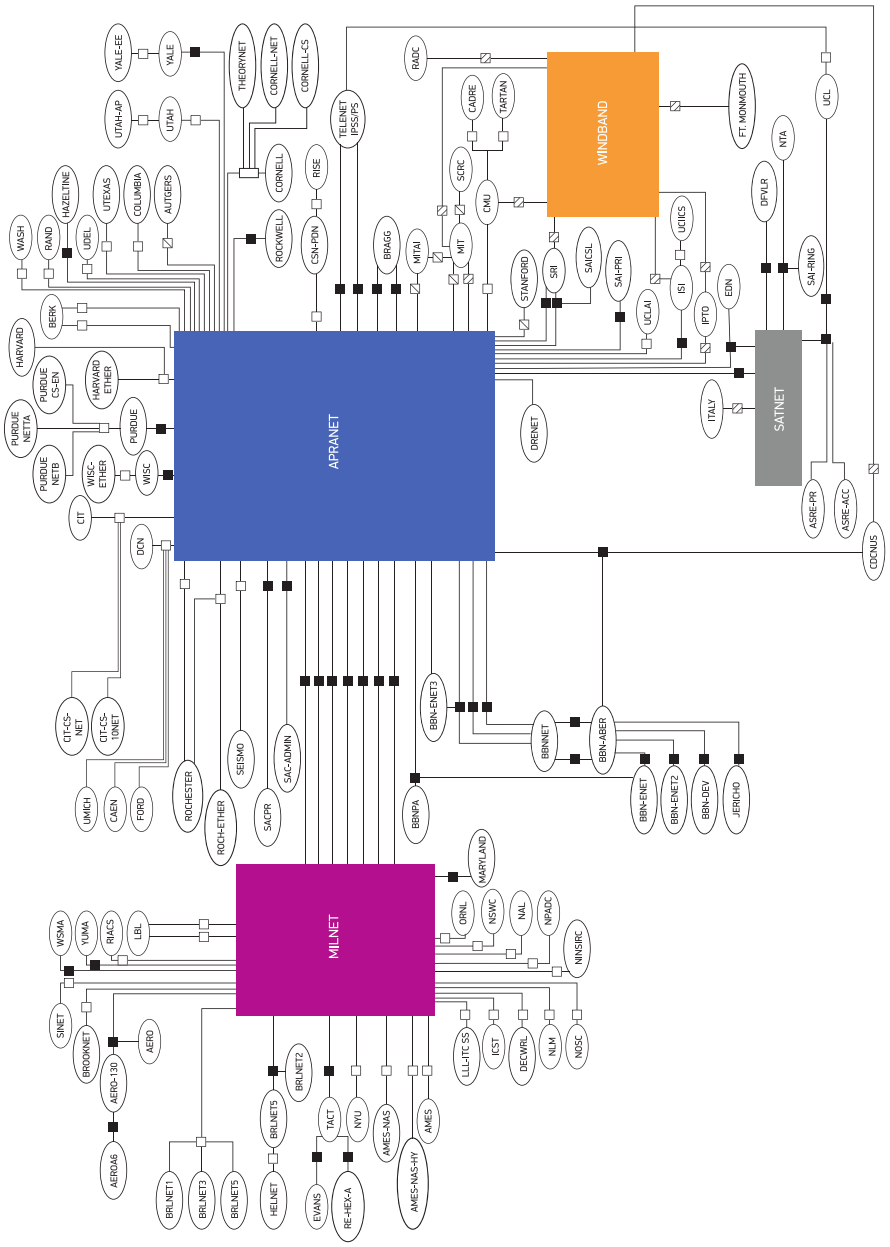

Рис. 36. Карта Интернета образца начала 1986 г.: небольшие квадраты — маршрутизаторы, овалы — сети, а крупные прямоугольники — опорные сети.
Источник: материалы второго совещания IETF в апреле 1986 г. (http://www.ietf.org/proceedings/02.pdf)

предусматривал строго иерархическую топологию и отсутствие колец, что хорошо соответствовало топологии Сети того времени. Топология тогдашней Сети показана на рис. 36.

В 1984 году NSF начал программу по созданию научно-исследовательских суперкомпьютерных центров, к которым могли бы иметь доступ американские ученые. Важной частью программы было создание высокоскоростной сети (NSFNET), связывающей эти суперкомпьютерные центры, а также все существовавшие региональные и академические сети. Сеть NSFNET, соединившая в 1986 году шесть суперкомпьютерных центров в США, являлась единым административным доменом с единой опорной сетью. Эта сеть представляла собой логическую точку обмена трафиком и, по существу, была в начале 1990-х опорной сетью Интернета. NSFNET также обменивалась трафиком с другими опорными сетями, такими как научная сеть NASA или ARPANET. В качестве протокола маршрутизации на начальном этапе было решено использовать протокол EGP.

Изначально предполагалось, что сеть будет иметь древовидную иерархическую топологию, напоминающую ARPANET, но на практике сети часто подключались к нескольким региональным сетям. Поскольку EGP не предоставлял необходимых метрик, на уровне опорной сети выбрать оптимальный маршрут было проблематично, и это иногда приводило к потере связности. Для решения проблемы было предложено использование так называемых политик маршрутизации — набора правил, которые сеть определяет для обмена маршрутизационной информацией с другими сетями. В данном случае каждой сети предлагалось[4] описать, с какими региональными сетями она связана, какой линк является основным, а какие — вторичными. Таким образом опорная сеть сможет использовать эту дополнительную информацию для принятия решений. Однако недостатком такой методики была плохая масштабируемость. Когда в конце 1980-х началось значительное расширение сети, назрела необходимость внедрения нового протокола.

BGP был разработан как для работы в иерархических сетях, таких как изначальная сеть NSFNET, так и в сетях с неиерархической топологией, когда сети одного уровня иерархии — пиры — могут быть непосредственно соединены друг с другом. Благодаря этому фундаментальному изменению BGP в дальнейшем обеспечил развитие Интернета как коллекции независимых, различным образом соединенных друг с другом сетей.

Первая версия протокола была стандартизована в 1989 году, хотя к тому времени он уже использовался в некоторых сетях. Согласно модели BGP каждый узел обменивается со своими соседями информацией о доступных путях к другим сетям, а именно о последовательности узлов, через которые должен быть передан трафик, чтобы достигнуть сети получателя. Таким образом,

[4] RFC 1092: EGP and policy based routing in the new NSFNET backbone, URL: https://www.rfc-editor.org/rfc/rfc1092

каждый узел имеет собственное представление о различных путях, но не о топологии Интернета в целом. Также, поскольку сети уже не могут рассматриваться как иерархические, недостаточно и информации о суммарной «стоимости» пути (как было в ранних протоколах маршрутизации), так как это может приводить к возникновению циклов. Чтобы избежать данной проблемы, каждый анонсируемый маршрут имеет специальный атрибут — AS_PATH, который содержит последовательность всех сетей, через которые этот анонс был передан. Если сеть обнаружит себя в списке AS_PATH полученного маршрута, это свидетельствует о цикле и такой маршрут должен быть отброшен.

Итак, отдельные сети, находящиеся под единым административным контролем и имеющие согласованную политику маршрутизации, обеспечивают отсутствие внутренних циклов. А значит, межсетевую и внутрисетевую маршрутизацию можно рассматривать независимо. Соответственно, в контексте межсетевой маршрутизации такие сети могут рассматриваться как метаузлы, или автономные системы (AS).

С одной стороны, данный подход позволил значительно упростить глобальную систему маршрутизации и обеспечить масштабируемость Интернета, с другой — он был основан на транзитивном доверии между взаимодействующими сетями.

Доверие — интересная вещь. В случае протокола BGP оно радикально упрощает взаимодействие между сетевыми операторами и, как следствие, систему маршрутизации в целом. Это, безусловно, явилось одним из факторов, обеспечивших бурное развитие Интернета. С другой стороны, доверие открывает существенные возможности для игроков не по правилам. Атаки «захват префикса» и перехват трафика случаются в Интернете постоянно, хотя прозрачность системы и сотрудничество между операторами играют существенную положительную роль. Об этом мы поговорим в следующих разделах.

BGP: принятие решений

В модели BGP каждая сеть, или автономная система, обменивается информацией об известных ей маршрутах с соседними сетями (в английской терминологии используются термины BGP neighbours, adjacent networks — смежные сети). Эта информация в терминах BGP называется NLRI (Network Layer Reachability Information, дословно — информация досягаемости сетевого уровня) и содержит префикс и его длину, что необходимо для поддержки CIDR. Например, NLRI/24, 198.51.100 указывает на маршрут к устройствам с IP-адресами в диапазоне 198.51.100.0–198.51.100.255.

Помимо NLRI в сообщении BGP (BGP Update) также содержатся так называемые атрибуты — параметры, связанные с анонсированным NLRI, которые используются при выборе «лучшего» маршрута и реализации политики маршру-

тизации. Дело в том, что маршрут к определенной сети, или NLRI, может быть получен от разных сетей, а в задачу BGP входит выбор «лучшего» из них. Слово «лучший» не случайно взято в кавычки — критерии и предпочтения в отношении маршрутов и потоков трафика определяются экономическими соображениями, а также топологией, распределением трафика и т.п. Они формируются в каждой автономной системе независимо. Обязательными и «общеизвестными» являются три атрибута: ORIGIN, AS_PATH и NEXT_HOP, — они присутствуют в любом анонсе BGP, их должны понимать все маршрутизаторы. Все три атрибута содержат информацию о пути маршрута:

1. **ORIGIN** — данные о том, как маршрут попал в глобальную систему BGP (ниже мы обсудим роль этого атрибута при выборе маршрута).
2. **AS_PATH** — список номеров автономных систем, через которые был передан этот маршрут.
3. **NEXT_HOP** — IP-адрес следующего маршрутизатора, которому нужно направить трафик, адресованный анонсированному NLRI. Обычно это маршрутизатор, от которого был получен анонс.

Говоря о процессе обработки маршрутов, стоит отметить, что анонсирование маршрута соседней сетью предполагает обязательство доставить данные, адресованные получателям этой сети. Сеть не контролирует, какие анонсы она получает от соседей, но она может проводить их фильтрацию и модификацию для влияния на стандартный процесс принятия решений BGP. Какие обязательства сеть готова взять на себя и какие сети-соседи использовать для передачи данных? Это и определяет политика маршрутизации, на которой мы остановимся в следующем разделе.

Процесс можно разделить на несколько этапов (см. рис. 35).

- Этап обработки входящих анонсов, полученных от соседних автономных систем. Для каждой автономной системы, с которой установлен сеанс BGP, маршрутизатор поддерживает отдельную базу маршрутов RIB-In (Routing Information Base). Если анонс вызывает изменение в этой базе, тогда обработка продолжается. Перед передачей полученных новых маршрутов (или информации об удалении существующих) процессу выбора маршрутов BGP для каждого соседа применяются правила, установленные политикой импорта. О политике маршрутизации мы поговорим в следующем разделе, здесь же скажем, что правилом может быть, например, фильтрация определенных маршрутов или изменение определенных атрибутов для влияния на процесс выбора.
- Этап выбора лучшего маршрута и его установка в локальную таблицу маршрутизации и таблицу передачи — Forwarding Information Base (FIB), которая отвечает за коммутацию пакетов с одного интерфейса маршрутизатора на другой на низком уровне. При выборе лучшего маршрута принимаются во внимание новые маршруты, полученные от соседей и прошедшие обработку, определенную политикой импорта.

- Этап обработки исходящих анонсов, передаваемых соседним автономным системам, об изменениях в локальной таблице маршрутизации. Эти изменения также сначала обрабатываются в соответствии с политикой экспорта, которая может, например, указывать, что некоторым соседям анонсы не должны отправляться вообще.

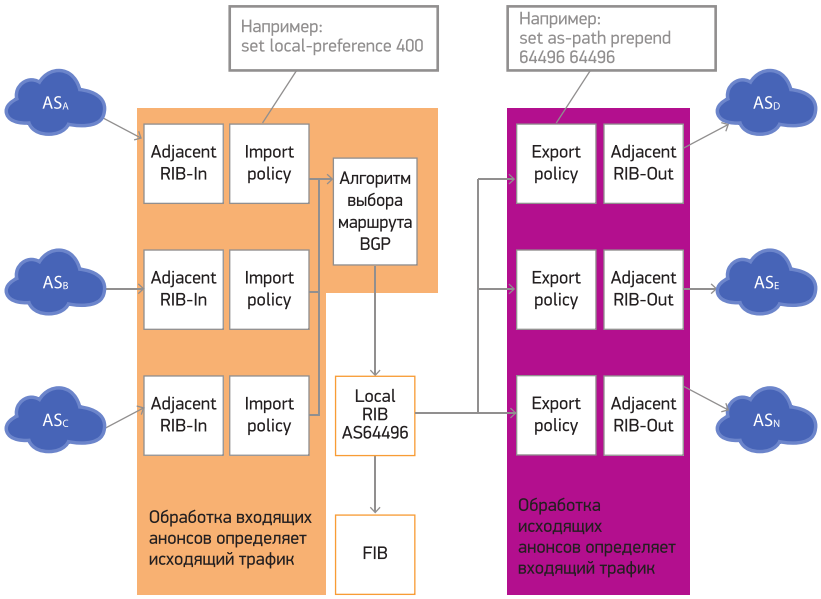

Рис. 37. Схема обработки анонсов BGP-маршрутизатором.

Алгоритм выбора лучшего маршрута к определенной сети (разумеется, в случае, когда маршрутизатором получено более одного маршрута) описан в RFC 4271 «A Border Gateway Protocol 4 (BGP-4)»[5]. В таблице ниже приведены шаги выбора в порядке приоритета. Если результатом какого-либо шага является единственный маршрут, процесс останавливается. Верно и обратное — пока у BGP есть выбор, процесс будет продолжаться, сравнивая более низкоприоритетные параметры маршрутов.

Таблица 6. Шаги выбора маршрута

Приоритет	Метрика	Правило
1.	LOCAL_ PREF	Выбирается маршрут (NLRI) с более высоким значением атрибута LOCAL_PREF. Этот атрибут устанавливается в рамках политики импорта и используется только внутри автономной системы для информирования других граничных маршрутизаторов о приоритете того или иного маршрута.
2.	AS_PATH	Выбирается маршрут с наименьшей AS_PATH. Эта метрика содержит последовательность всех сетей, через которые он был передан. Соответственно, BGP выбирает кратчайший маршрут. В общем случае AS_PATH является основной метрикой принятия решений.

[5] RFC 4271: A Border Gateway Protocol 4 (BGP-4), URL: https://www.rfc-editor.org/rfc/rfc4271#section-9.1.2

Приоритет	Метрика	Правило
3.	ORIGIN	Выбирается маршрут с наименьшим значением этого атрибута. ORIGIN указывает, как маршрут изначально появился в системе BGP, и является своего рода рудиментом со времени, когда Интернет перешел от протокола EGP к BGP. Возможных значений этого атрибута три: ● (0) IGP или internal, когда сеть непосредственно подключена к маршрутизатору, анонсирующему ее в BGP; ● (1) EGP – если маршрут был получен от EGP (это значение может быть установлено искусственно, поскольку EGP давно не используется в Интернете); ● (2) Incomplete – указывает, что маршрут был получен от других протоколов маршрутизации, поэтому реальный путь (за пределами AS_PATH) неизвестен.Выбор происходит в порядке 0–1–2. Обычно этот атрибут не привлекает особого внимания операторов, но при желании может использоваться для влияния на выбор маршрута другими автономными системами, поскольку он обязательно присутствует во всех анонсах BGP.
4.	MED	Выбирается маршрут с наименьшим значением этого атрибута. MED (Multi-exit discriminator, дискриминатор нескольких выходов) используется для определения политики передачи трафика между сетями, связанными в нескольких географически распределенных точках.
5.	Откуда получен маршрут	Маршрут, полученный от соседей (eBGP), предпочитается маршруту, полученному от других граничных маршрутизаторов (iBGP).
6.	Минимальный внутренний маршрут	Выбирается маршрут с минимальным путем по внутренней инфраструктуре до узла NEXT_HOP – маршрутизатора, передавшего анонс.
7.	Номер AS	Если маршруты были получены от внешних сетей, выбирается маршрут от автономной системы с наименьшим номером.
8.	ID внутреннего маршрутизатора	Если маршруты были получены от граничных маршрутизаторов своей же сети, выбирается маршрут, переданный маршрутизатором с «наименьшим» IP-адресом.

Последние два правила не имеют никакого практического смысла и используются для предсказуемого выбора маршрута, если все остальные правила привелитаки к ничьей.

Связность: пиры, клиенты и политика маршрутизации

Хотя BGP и определяет стандартный алгоритм выбора маршрута, который мы только что рассмотрели, фактически выбор делается на основе политики маршрутизации, принятой данной автономной системой. Другими словами, администратор сети может явно определить правила выбора маршрута в определенных условиях. Например, изначально политика маршрутизации в NSFNET разделяла коммерческий и некоммерческий трафик. Важно представлять, что этот выбор основан на информации, полученной сетью от ее соседей, или пиров. При этом не существует общей «эталонной» топологии Интернета – каждая автономная система вырабатывает свою собственную картину мира, которую, в свою очередь, транслирует своим пирам.

Другими словами, политика маршрутизации позволяет сети менять стандартный процесс выбора маршрута BGP. Во многих случаях эти изменения касаются собственного выбора (политика импорта), но также используются приемы для влияния на процесс выбора лучшего пути других сетей (политика экспорта). Для определения политики существенное значение имеет тип отношений между взаимодействующими сетями.

Здесь можно выделить два основных типа отношений между сетями: клиент-провайдер (транзит) и пиринг (peering).

Отношения клиент-провайдер предусматривают, что провайдер предоставляет услуги транзита сети клиента. Это означает, что клиент получает доступ не только к сетям провайдера, но и к другим, внешним сетям. Как правило, под внешними сетями подразумевается глобальный Интернет. Обычно оплата клиента за оказание таких услуг зависит от пропускной способности канала и его фактической загрузки.

В случае пиринга две сети (пиры) договариваются об обмене трафиком между собственными сетями, включая своих клиентов. Обычно пиринг не предусматривает взаиморасчетов, поскольку такие отношения заключаются между равноценными сетями, когда оба партнера получают примерно одинаковую выгоду от обмена трафиком. Основными преимуществами пиринга являются, конечно, облегчение нагрузки (и, соответственно, расходов) на провайдера транзита, а также повышение устойчивости за счет более богатой связности.

Это, конечно же, упрощенная картина. Существуют и другие разновидности политик обмена трафиком — платный пиринг, пиринг провайдера контента и т.п. Все они определяются в первую очередь экономическими соображениями.

Глобальная система маршрутизации обладает удивительной способностью к самоорганизации — каждая из сетей формирует свою политику независимо, и тем не менее, в результате обеспечивается глобальная связность. Это еще более удивительно, учитывая, что сегодняшний Интернет имеет не такую иерархическую структуру, как, скажем, NSFNET (см. рис. 36). Правда, хотя в Интернете и нет единой опорной сети, существует дюжина крупнейших сетей, таких как AT&T, Level3, Deutsche Telecom, которые играют ключевую роль в обеспечении глобальной связности. Эти сети еще называют сетями первого уровня, или сетями Tier-1. Все остальные сети Интернета являются непосредственными или опосредованными клиентами сетей Tier-1. У сети Tier-1 нет провайдера транзита — обмен трафиком между собой они осуществляют на основе пиринговых отношений. На этом уровне и формируется глобальная связность. Транзит и пиринг, помимо взаимного соглашения — контракта в случае транзита и чаще всего устной договоренности в случае пиринга, — обеспечиваются техническими средствами. С каждым из типов связана определенная политика маршрутизации и, соответственно, технические средства ее реализации.

Говоря о технических средствах, необходимо упомянуть о специальном языке описания политики маршрутизации — RPSL (Routing Policy Specification Language). Синтаксис этого языка определен в документе RFC 2622[6] с расширениями для поддержки адресации IPv6 и мультикаст, описанными в RFC 4012[7]. Возможность описания и публикации политики нужна, чтобы определять и исправлять проблемы маршрутизации в глобальном масштабе. RPSL также позволяет генерировать конфигурационные файлы для маршрутизаторов, осуществляющих эту политику.

Не будем вдаваться в глубины этого языка, приведем лишь несколько примеров политик в стиле RPSL. Напомним, что политики импорта влияют на потоки исходящего трафика, а политики экспорта — входящего.

Транзит: AS1-провайдер, AS2клиент, AS2-CUSTOMERS — все клиенты AS2.

```
aut-num:    AS2
import:     from AS1 accept ANY
export:     to AS1 announce AS2 AS2-CUSTOMERS

aut-num:    AS1
import:     from AS2 accept AS2 AS2-CUSTOMERS
export:     to AS2 announce ANY
```

Пиринг: AS1 и AS2 являются пирами.

```
aut-num:    AS2
import:     from AS1 accept AS1 AS1-CUSTOMERS
export:     to AS1 announce AS2 AS2-CUSTOMERS

aut-num:    AS1
import:     from AS2 accept AS2 AS2-CUSTOMERS
export:     to AS2 announce AS1 AS1-CUSTOMERS
```

Многие сети одновременно играют все три роли: провайдера — для более мелких сетей, клиента — для получения транзита и глобальной связности и пира — для обмена трафиком с равноценными сетями. Обычно маршруты, полученные от клиентов, предпочитаются маршрутам от пиров, а маршруты от провайдера транзита обычно имеют наименьший приоритет. Это разумно: хотя ваш клиент может также анонсироваться и пиром, все-таки предпочтительнее

[6] RFC 2622: Routing Policy Specification Language (RPSL), URL: https://www.rfc-editor.org/rfc/rfc2622

[7] RFC 4012: Routing Policy Specification Language next generation (RPSLng), URL: https://www.rfc-editor.org/rfc/rfc4012

посылать трафик клиенту напрямую. А для доступа к сетям пира неразумно пользоваться платным транзитом, поэтому транзитные анонсы и стоят последними при выборе наилучшего маршрута.

Однако эти правила не являются стандартными в выборе маршрута BGP: определения «клиент», «пир» или «провайдер» обозначают деловые отношения и не имеют смысла в контексте протокола[8]. Такие правила должны быть реализованы политикой импорта. Помните, мы говорили об атрибуте LOCAL_PREF, позволяющем изменять предпочтение маршрута? Этот атрибут не передается между автономными системами, поэтому он полезен только для принятия собственных решений и оповещения о них других граничных маршрутизаторов сети.

Если AS1 получает транзит от AS4, осуществляет пиринг с AS3 и является провайдером для AS2, то политика импорта этой сети может выглядеть следующим образом:

```
aut-num:    AS1
import:     from AS2 action pref=300; accept AS2 AS2-CUSTOMERS
import:     from AS3 action pref=200; accept AS3 AS3-CUSTOMERS
import:     from AS4 action pref=100; accept ANY
```

Отсутствие контроля за выполнением этих политик на уровне конфигурации фильтров import и export маршрутизатора (см. рис. 36) может привести к аномалиям. Например, вследствие ошибки конфигурации пир может стать провайдером транзита. Это может привести к неоптимальным потокам трафика и потере качества. Гораздо более серьезные последствия могут возникнуть, если вашим провайдером вдруг станет клиент. Получение от него маршрута по умолчанию («default», о/о) приведет именно к такой ситуации, поскольку маршруты, полученные от клиента, всегда имеют приоритет. Наиболее вероятно, что последствием будет отказ в обслуживании вследствие перегрузки инфраструктуры сети клиента. Но «утечка» может быть и менее заметной, например, если речь идет всего о нескольких маршрутах, принадлежащих другим сетям. Такие ситуации получили название «утечка маршрута» (route leak), и они могут иметь серьезные последствия в плане безопасности, поскольку позволяют клиенту «вставить» свою сеть между вашими пользователями и внешними сетями, открывая широкие возможности для атак типа MITM и прослушивания трафика. Представьте, что утечке подвергнется сеть сайта онлайн-платежей

[8] Недавно принятый стандарт RFC9234 "Route Leak Prevention and Detection Using Roles in UPDATE and OPEN Messages" (https://www.rfc-editor.org/rfc/rfc9234) позволяет зафиксировать характер взаимоотношений между сетями на этапе создания BGP-сессии. Впоследствии это может быть использовано для контроля правильности политики маршрутизации. Более подробно мы поговорим об этом в разделе «Безопасность системы маршрутизации».

или сайта социальной сети. Мы вернемся к этой теме в разделе «Безопасность системы маршрутизации».

Соблюдение правил, диктуемых характером взаимоотношений между сетями, обеспечивает маршрут с характеристикой, получившей название valley-free («не пересекая долины»). Маршрут является valley-free, если он состоит из трех последовательных участков, каждый из которых может быть нулевым: движение в направлении клиент-провайдер, максимум один пиринговый линк и, наконец, движение в направлении провайдер-клиент. В этом случае клиенты остаются клиентами, пиры — пирами, а провайдеры предоставляют заранее оговоренный транзит.

Чтобы получить иллюстрацию этого факта, взгляните на рис. 38. Маршруты AS_2-AS_1-AS_C-AS_{10} и AS_5-AS_3-AS_7-AS_6 являются «правильными» с точки зрения соблюдения политик. В то же время маршруты AS_1-AS_5-AS_3, AS_7-AS_8-AS_9 и AS_B-AS_8-AS_9 явно не соответствуют предполагаемым отношениям между сетями. Вы заметите, что все эти маршруты содержат «спуск в долину» или движение в «долине» через два и более пиринговых линка.

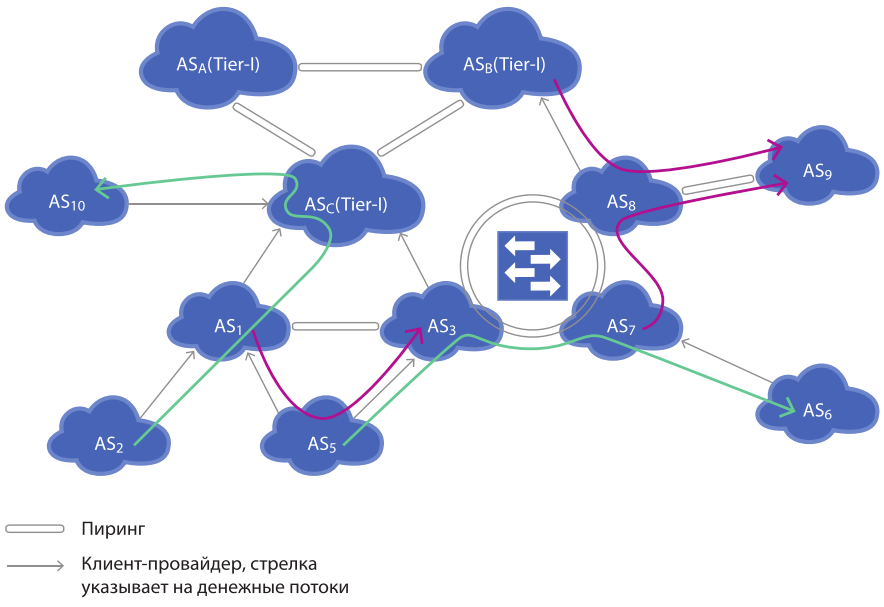

Рис. 38. Структура сегодняшнего Интернета. «Правильные» и «неправильные» маршруты между связанными сетями. Красными линиями обозначены маршруты, нарушающие принцип valley-free (непересечения долины).

Безопасность системы маршрутизации

Протокол маршрутизации BGP по существу является нервной системой Интернета. Несмотря на свою фундаментальную значимость, протокол BGP основан на доверии между соединенными сетями, ведь полученная от них информация принимается за чистую монету. Более того, это доверие обладает транзитивным свойством: пиры доверяют своим соседям, те, в свою очередь, — своим, и в итоге все доверяют всем.

Другими словами, BGP позволяет «лгать». И эта ложь, если не поставить дополнительных заслонов, будет распространяться по всему Интернету. Она может быть следствием ошибок конфигурации или умышленным подлогом. Последствия ее могут быть невинны и незаметны, а могут перерасти в атаку, затрагивающую все системы и сервисы.

Разумеется, сеть вольна не доверять полученной информации и осуществить дополнительную проверку. Но, к сожалению, собственно протокол BGP здесь вряд ли поможет, поэтому необходимо прибегать к дополнительным средствам, о которых мы и поговорим в этом разделе.

Как можно атаковать систему маршрутизации

В информационном документе IETF RFC 4593[9] обсуждаются потенциальные угрозы системы маршрутизации, а RFC 4272[10] подробно указывает на уязвимые места протокола BGP. Вот эти уязвимости:

- отсутствие внутреннего механизма, обеспечивающего сильную защиту целостности, свежести и аутентичности сообщений BGP, которыми обмениваются сети-пиры друг с другом;
- отсутствие механизма для проверки прав автономной системы или сети анонсировать префикс;
- отсутствие механизма для проверки подлинности атрибутов пути, анонсированных сетью-пиром.

Векторы атаки на систему маршрутизации представлены на рис. 39. Первая проблема имеет отношение к защите канала между пирами и часто решается локальными средствами. Рабочая группа IETF KARP[11] предлагает продвинутые решения в этой области. Две остальные группы уязвимости имеют более существенное значение в глобальном масштабе.

---

9 RFC 4593: Generic Threats to Routing Protocols, URL: https://www.rfc-editor.org/rfc/rfc4593

10 RFC 4272: BGP Security Vulnerabilities Analysis, URL: https://www.rfc-editor.org/rfc/rfc4272

11 https://datatracker.ietf.org/wg/karp

Можно выделить несколько общих типов атаки. Несмотря на различие целей и конечного эффекта, механизм атаки принципиально строится на том, что в атакуемой сети создается искаженная картина топологии Интернета. Затем эта искаженная картина транзитивно распространяется по всей Сети.

Создание «черных дыр». Цель этой атаки — отрезать сеть или несколько сетей от всего Интернета или его части. Весь трафик, имеющий отношение к этим сетям, перенаправляется и затем отбрасывается. В результате все сервисы, предлагаемые данными сетями, становятся недоступными для пользователей. Основным результатом атак этого типа является отказ в обслуживании (Denial of Service, DoS).

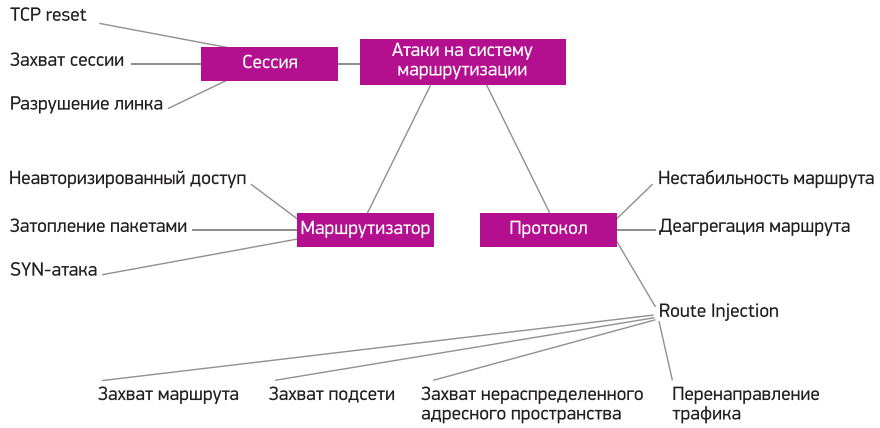

Рис. 39. Атаки на систему маршрутизации.

Перенаправление. В этом случае трафик, предназначенный для одной сети, перенаправляется в другую. Часто эта сеть находится в руках атакующего и маскируется под атакуемую сеть — например, с целью получения секретной информации. Также перенаправление может быть использовано для проведения злоумышленниками определенных краткосрочных акций, например, рассылки спама. После этого такая сеть, или ее фантом, исчезает. Часто злоумышленниками используется нераспределенное или давно не используемое адресное пространство.

Перехват. Эта атака похожа на предыдущую, только после прохождения по сети-перехватчику трафик возвращается в нормальное русло и попадает к получателю. Из-за этого такую атаку труднее обнаружить. Целью обычно является «подслушивание» или модификация передаваемых данных.

Нестабильность. Нестабильность в глобальной системе маршрутизации может быть вызвана частыми изменениями в анонсировании конкретной сети (попеременное анонсирование и аннулирование). Цель — «демпфирование» маршрутов данной сети провайдерами и, как следствие, блокирование связности.

Фабрикация адреса источника трафика. В этом случае непосредственно система маршрутизации не подвергается атаке, но данный метод широко используется в так называемых рефлекторных атаках, о которых мы говорили в главе 2. Обратный трафик (например, ответы на изначальные запросы) направляется не к истинному источнику, а к получателю, чей адрес был сфабрикован. Как правило, такие атаки используют протокол без установления соединения UDP[12] и основаны на эффекте усиления, когда небольшие запросы от многих источников порождают ответы значительно большего размера. Одной из критических систем, в основном использующей UDP и подверженной атакам такого рода, является DNS.

Раз нет механизмов проверки подлинности полученной информации, значит атакующий может повлиять на маршрутизацию трафика, относящегося к той или иной сети, в глобальном масштабе. Наиболее распространенными является захват префикса, или маршрута (prefix hijacking, route hijacking), когда префикс какой-либо сети анонсируется атакующим — и трафик, предназначенный этой сети, перенаправляется в сторону атакующего (см. рис. 40).

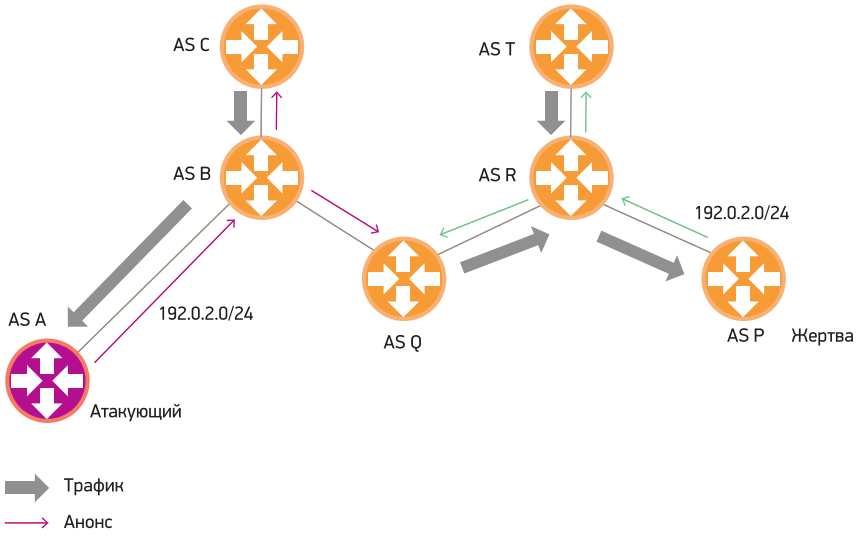

Рис. 40. Захват префикса. Атакующая сеть AS A анонсирует префикс, принадлежащий сети AS P. В результате трафик в части Интернета перенаправляется к атакующей сети.

Эта атака имеет несколько вариантов:

- захват маршрута, когда сеть анонсирует не принадлежащее ей адресное пространство в качестве сети-источника. При выборе маршрута BGP предпочтет более короткий путь, измеряемый числом сетей между источником

[12] User Datagram Protocol, http://ru.wikipedia.org/wiki/Udp

и получателем маршрута. Таким образом, захваченный маршрут будет конкурировать с истинным;

- захват подсетей, когда анонсируются более специфичные префиксы. При выборе маршрута BGP предпочитает тот, который указывается более специфичным префиксом, и таким образом атакующий выигрывает, несмотря на топологическую удаленность. В случае отсутствия конкурирующих префиксов такого же размера захват оказывает глобальный эффект;
- захват нераспределенного или неиспользуемого адресного пространства. В этом случае анонсируемый префикс не встречает конкуренции и имеет высокие шансы распространения по всему Интернету. В то же время очевидные недопустимые префиксы, так называемые bogons, как правило, фильтруются провайдерами.

Классическим примером атаки захвата префикса является захват сайта YouTube в феврале 2008 года, когда оператор Pakistan Telecom (AS17557) начал несанкционированное анонсирование части адресного пространства, используемого YouTube (AS36561), а именно более конкретного (more specific) префикса 208.65.153.0/24. Один из транзитных провайдеров Pakistan Telecom, PCCW Global (AS3491), проанонсировал данный маршрут далее, в глобальный Интернет, что привело к перенаправлению трафика YouTube в глобальном масштабе.

Топология связности YouTube уже спустя две минуты выглядела следующим образом:

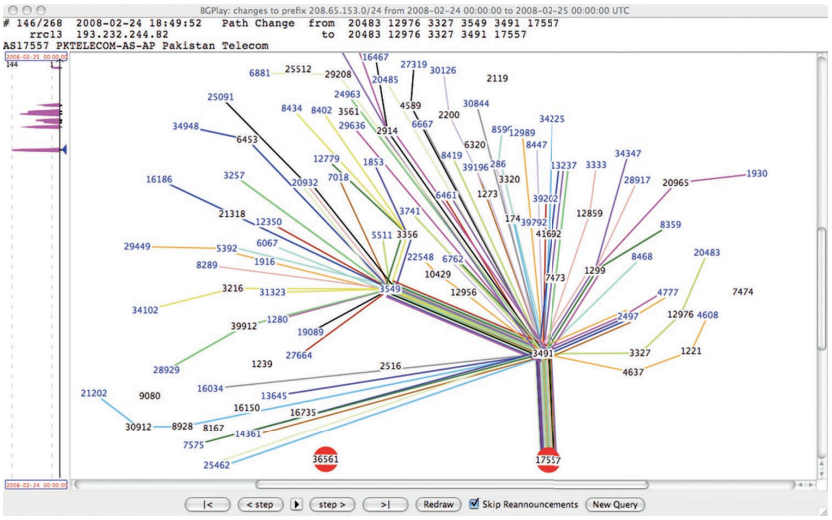

Рис. 41. Топология связности YouTube (AS36561) после начала анонсирования префикса Pakistan Telecom (AS17557).
Источник: «YouTube Hijacking: A RIPE NCC RIS case study» (http://www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study)

Как видно, весь трафик, предназначенный YouTube, был перенаправлен в сеть Pakistan Telecom. Этот трафик, скорее всего, представлял собой обрывки сеансов TCP, начатых с реальным сайтом YouTube, а также попытки начать новые сеансы; вероятно, он попросту отбрасывался сетью Pakistan Telecom. Для пользователей YouTube это выглядело как недоступность ресурса.

Причиной данного эксцесса явилось требование правительства Пакистана заблокировать доступ к враждебному сайту внутри страны. Однако в результате была создана типичная «черная дыра», что и привело к глобальному сбою в работе служб YouTube.

Последствия атак этого типа могут быть различными. Захват маршрута приводит к перетягиванию трафика, предназначенного «захваченной» сети, который, как правило, затем отбрасывается. То есть происходит DoS-атака на все сервисы сети. Такая атака может также быть использована для краткосрочной генерации трафика, например, для рассылки спама. В более изощренном виде захват маршрута может быть направлен на захват некоторого информационного ресурса, например веб-сайта, когда пользователям демонстрируется подложный сайт. В этом случае даже защита DNSSEC будет бессильна. А учитывая относительную простоту получения сертификата TLS, такая атака может иметь серьезные последствия для пользователей — например, кражу данных по кредитным картам.

Атака Пилосова (Pilosov)

Атака на YouTube имела значительные видимые последствия, она получила широкую огласку и резонанс в сетевом сообществе. Однако ряд атак могут проходить почти незаметно — но приводить к не менее серьезным, а то и более значительным последствиям.

Речь идет о перехвате трафика, незаметном как для отправителя и получателя трафика, так и для большинства других участников обмена информацией. Целью может быть, например, перлюстрация данных, которыми обмениваются определенные сети или пользователи. Злоумышленник может также попытаться модифицировать эти данные.

Возможность и простота организации такой атаки были описаны на конференции DEFCON в августе 2008 года Алексом Пилосовым (Alex Pilosov) и Антоном Капелла (Tony Kapella). Они продемонстрировали, что:

- практически любой префикс может быть захвачен без нарушения сквозной связности;
- это можно сделать очень незаметно, замаскировав присутствие атакующего на пути следования трафика (он невидим для утилит типа traceroute, позволяющих получить список узлов, через которые передается трафик к получателю).

Суть атаки сводится к перехвату трафика стандартными методами (например, путем анонсирования атакующим более длинного префикса атакуемой сети, что делает анонс данной подсети более привлекательным с точки зрения BGP, как это произошло в случае с Pakistan Telecom). Далее трафик возвращается в прежнее русло. Для этого атакующая сеть может, например, установить статический маршрут (рис. 41). В результате трафик передается сети, являвшейся частью изначального пути передачи трафика. Далее передача трафика происходит абсолютно законным путем.

Для маскировки движения трафика атакующая сеть производит манипуляцию с параметром TTL (Time To Live) пакетов перехваченного трафика. Согласно протоколу, при передаче пакета от одного маршрутизатора к другому каждый узел сети уменьшает этот параметр на единицу. Не изменяя TTL при прохождении по атакующей сети, злоумышленники могут «замаскировать» этот участок, исключив таким образом атакующего из видимого пути передачи трафика. Для утилит типа traceroute участки пути в сети атакующего просто не попадают в список.

Атака Пилосова проиллюстрирована на рис. 42 и 43. Первый рисунок показывает состояние системы маршрутизации перед началом атаки, когда трафик от пользователя сети AS_{70} доставляется через AS_{60} получателю сети AS_{200}. На втором изображена связность сетей в процессе атаки. Хотя трафик перехватывается атакующей сетью AS_{100}, этот путь не отражается программой traceroute.

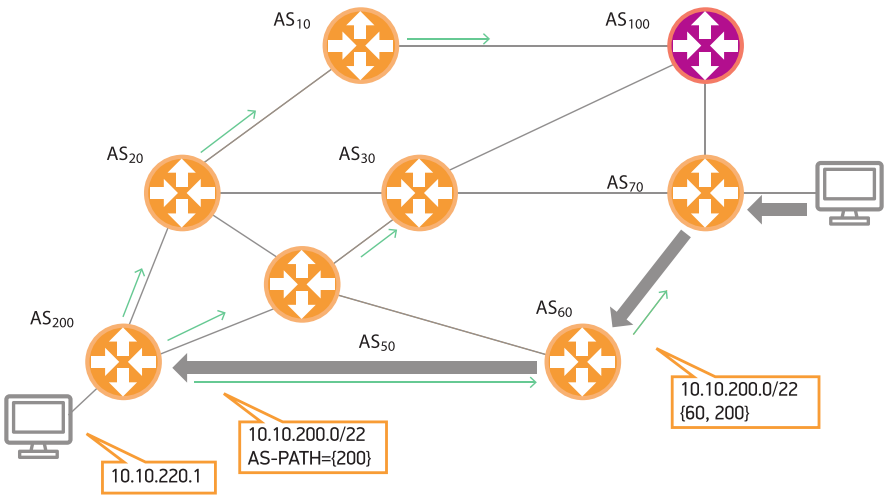

Рис. 42. Состояние системы маршрутизации перед началом атаки Пилосова.

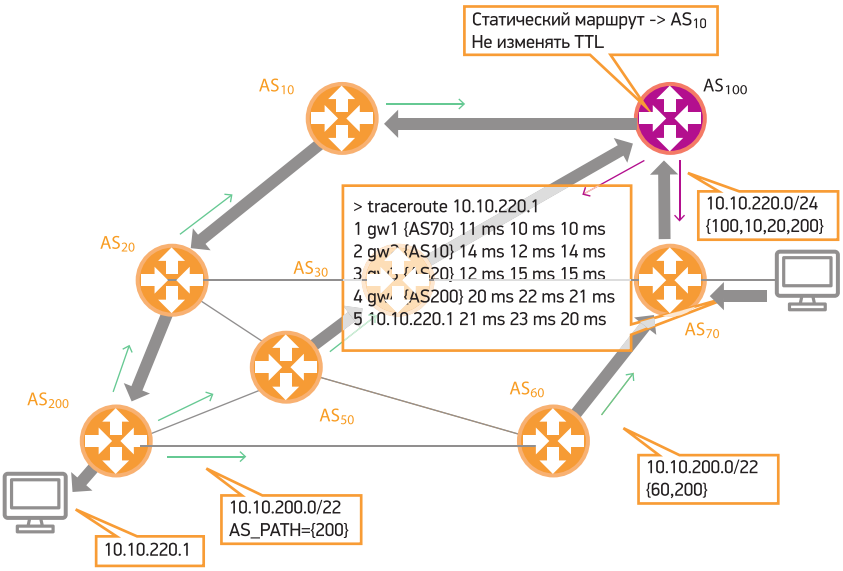

Рис. 43. Перехват трафика атакующей сетью (AS_{100}).

Упрощенной версией атаки Пилосова–Капеллы является простая «утечка маршрута» (route leak), о которой мы говорили в предыдущем разделе. Безусловно, в этом случае аномалия не маскируется, но в случае ограниченной утечки и умеренного трафика может оставаться незамеченной на протяжении долгого времени.

Очевидно, что глобальная система маршрутизации уязвима для всех перечисленных атак. Тем не менее, связанные с ними риски зачастую считаются незначительными и «принимаются» без дополнительных мер защиты. Отчасти это связано с тем, что многие из «атак» являются последствиями ошибок конфигурации, достаточно заметны за счет прозрачности системы и достаточно быстро разрушаются. Опытные операторы рассматривают эту проблему более серьезно; правда, в большинстве случаев учитываются только риски, связанные с непосредственным окружением — сетями, клиентами и пирами. Как правило, это отражается в практике фильтрации маршрутов собственных сетей-клиентов и установки максимального числа принимаемых маршрутов от пиров и сетей, предоставляющих транзит. Реже производится фильтрация маршрутов, полученных от пиров.

Существующая практика безопасности маршрутизации

Как вы, наверное, заметили, рассмотренные выше атаки являются нарушением предполагаемой политики маршрутизации. Одно дело определить политику, и совсем другое — обеспечить ее выполнение техническими средствами. Необходимо получить достоверные данные о правомерных маршрутах, полученных

от клиента, пира или провайдера. Например, если наша политика по отношению клиента AS_2 выглядит так:

aut-num:	AS_1
import:	from AS_2 accept AS_2 AS_2-CUSTOMERS
export:	to AS_2 announce ANY

то как определить легитимные маршруты, которые могут быть анонсированы AS2 и ее клиентами? В самом протоколе BGP эта информация отсутствует, поэтому возникает необходимость во внешних источниках информации.

Итак, безопасность и надежность системы маршрутизации во многом зависят от возможности правильного ответа на вопросы:

- Является ли префикс, полученный в сообщении BGP, правомерным (то есть представляющим законно распределенное адресное пространство и право на его использование)?
- Является ли автономная система-отправитель сообщения BGP правомочным источником префикса?
- Соответствует ли атрибут AS_PATH, полученный в сообщении BGP, действительному пути, который прошло данное сообщение в сети Интернет?

К сожалению, дать правильные ответы на поставленные вопросы трудно ввиду отсутствия надежного источника информации. Но что же все-таки имеется в арсенале сервис-провайдера?

Интернет-регистратуры маршрутизации (IRR)

Частичную помощь в решении данной проблемы оказывают интернет-регистратуры маршрутизации (Internet Routing Registry, IRR). Суть их в следующем: сетевые операторы регистрируют в базе данных свою политику маршрутизации, а именно — с кем и как сеть взаимодействует. Также регистрируются и префиксы, которые сеть использует и анонсирует в Интернете. Для описания политик используется язык RPSL, о котором мы уже говорили. А инструментарий, наиболее известный из которых — IRRToolset[13], позволяет автоматизировать конфигурацию маршрутизации провайдера по данным IRR.

Но IRR отображают весьма неполную картину, так как регистрация данных в этих базах сугубо добровольная. Многие операторы не хотят морочить себе голову какими-то IRR, часть операторов не регистрируется там по причине нежелания разглашать свою политику. Те же, кто все же зарегистрировал свою политику, не всегда поддерживают актуальность данных. Проблема в том, что хотя эта деятельность служит на благо общего дела — безопасной системы маршрутизации, польза для самого провайдера не всегда ощутима.

[13] https://www.isc.org/othersoftware/#IRR

Неполнота и ненадежное качество данных, а также плохая масштабируемость — попробуйте-ка создать фильтры для всех префиксов, зарегистрированных в IRR! — существенно ограничивают применение интернет-регистратур для решения проблем безопасности глобальной маршрутизации.

В результате IRR имеют весьма ограниченное распространение и в основном используются для администрирования провайдером подключенных клиентов.

Whois

Можно воспользоваться более надежными базами данных распределения адресного пространства на уровне региональных интернет-регистратур (Regional Internet Registry, RIR). Эту информацию можно получить через соответствующий whois-сервер регистратуры, но иногда более практично использовать так называемые файлы статистики, доступные на ftp-сайте всех РИРов[14]. Например, интернет-ресурсы, распределенные RIPE NCC, представлены в отдельном файле[15].

Как вы можете заметить, число записей весьма внушительно. Также внушительным будет и список префиксов в конфигурации ваших граничных маршрутизаторов.

База данных IANA (Internet Assigned Number Authority, iana.org), например, для ресурсов IPv4[16], является более компактной, хотя и не содержит деталей — каждая запись имеет размер /8 в случае IPv4, а детализация для адресного пространства IPv6 и того меньше. Однако данный подход позволяет по крайней мере блокировать сети, использующие нераспределенные адресные ресурсы.

Сертификация номерных ресурсов и безопасность маршрутизации

Как мы уже говорили, существенные препятствия на пути обеспечения безопасной маршрутизации — недостаточная доступность и надежность данных, позволяющих оператору принять решение о достоверности анонсируемых маршрутов.

Система, построенная на основе RPKI, призвана решить эти проблемы. Ее фундамент — система открытых ключей (Public Key Infrastructure, PKI), элементами которой являются сертификаты интернет-ресурсов (СР). На основе этих сертификатов держатели ресурсов могут создавать криптографически заверенные объекты, например, ROA (Route Origin Authorization), указывающие на список автономных систем, которые могут являться источником определенного маршрута.

Во-первых, данные о распределенных номерных ресурсах предоставляются в стандартной форме цифровых сертификатов со стандартными расширениями (расширения X.509, о которых чуть позже, как раз содержат список ресурсов,

[14] Например, ftp://ftp.ripe.net/pub/stats
[15] ftp://ftp.ripe.net/pub/stats/ripencc/delegated-ripencc-latest
[16] http://iana.org/assignments/ipv4-address-space/ipv4-address-space.xml

привязанных к открытому ключу сертификата). Во-вторых, достоверность и свежесть данных может быть проверена с использованием криптографических средств третьими лицами. Как и в стандартном PKI, для проверки необходима конфигурация доверия только к одному сертификату — так называемые точки доверия (Trust Anchor, TA). В-третьих, сертификаты ресурсов могут использоваться их владельцами (держателями адресного пространства), например, для электронной авторизации определенных автономных систем для анонсирования данного адресного пространства. Таким образом, сертификаты выполняют функцию объектов route традиционных IRR.

Структура RPKI

Как и в случае традиционной PKI, в состав RPKI входят следующие компоненты (см. RFC 6480[17]):

Удостоверяющий центр сертификации (УЦС), задачи которого — выдача сертификатов и их отзыв. УЦС содержит две важные службы.

- Служба выдачи сертификатов (Certificate Authority, CA). Основной функцией CA является генерация и публикация сертификатов и списков аннулированных сертификатов. Эта функция, по существу, не меняется в RPKI, за исключением того, что CP содержат расширения, документирующие распределенные АИР.
- Служба регистрации (Registration Authority, RA) отвечает за проверку подлинности связи между субъектом сертификата и его ключом. В случае RPKI также удостоверяется, что субъект сертификата имеет права на использование номерных интернет-ресурсов (НИР), перечисленных в расширении. По существу, эта функция неотличима от функции регистрационных услуг, оказываемых сегодня РИР. Хотя предполагается, что УЦС удостоверяет субъекта CP, данное условие не является необходимым, и информация о субъекте в CP может не быть неявной (например, субъект может быть представлен цифровым идентификатором, имеющим значение только во внутренней структуре УЦС).

Репозитории — открытые базы данных, в которых публикуются выданные сертификаты, списки аннулированных сертификатов и в случае RPKI — вторичные объекты.

Сертификаты.
Система RPKI основана на сертификатах стандарта X.509. Они содержат критические расширения для документации АИР, стандартизованные в RFC 3779[18]. Такое расширение содержит список всех АИР (адресов IPv4 и IPv6, а также номера автономных систем), полученных субъектом сертификата. Важно

[17] RFC 6480: An Infrastructure to Support Secure Internet Routing, URL: https://www.rfc-editor.org/rfc/rfc6480

[18] RFC 3779: X.509 Extensions for IP Addresses and AS Identifiers, URL: https://www.rfc-editor.org/rfc/rfc3779

отметить, что в задачу CP не входит идентификация субъекта, в отличие от стандартной системы PKI. CP удостоверяет, что УЦС распределил определенные ресурсы субъекту сертификата и данные ресурсы перечислены в расширении сертификата. УЦС удостоверяет, что любой документ, подписанный секретным ключом, соответствующим открытому ключу сертификата, подписан законным обладателем прав на использование ИР, перечисленных в сертификате. Данная концепция схематично представлена на рис. 44.

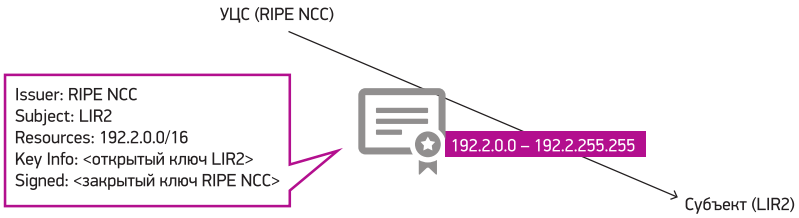

Рис. 44. X.509 Сертификат ресурсов с расширениями.

Вторичные объекты

Вторичные объекты специфичны для системы RPKI и, строго говоря, не являются ее частью, а предоставляются из соображений удобства практического применения RPKI. Они являются данными, подписанными владельцем сертификата. Один из наиболее проработанных вторичных объектов — так называемое разрешение на создание маршрута, или ROA (Route Origin Authorisation), определенное в RFC 6482[19]. Как следует из названия, ROA является разрешением, выданным сетью-владельцем прав на использование АИР.

Это разрешение на анонсирование данных ресурсов автономной системой (АС), указанной в ROA. В соответствии со спецификацией ROA содержит номер авторизованной АС и список IP-префиксов, которые эта АС имеет разрешение анонсировать. К этому «заявлению» прилагается сертификат, описывающий соответствующие АИР, и весь объект подписан с использованием ключа, указанного в сертификате. Также отметим, что наличие ROA не означает «согласие» авторизованной АС на то, что указанные префиксы непременно будут анонсированы данной автономной системой.

Как и любая система PKI, RPKI имеет иерархическую структуру с корневым сертификатом во главе. Поскольку для корневого сертификата не существует родительского УЦС, данный сертификат представляет собой самоподписанный корневой ключ. Организация, которой принадлежит корневой CP, является так называемой точкой доверия (Trust Anchor). Важно отметить, что, как и в любой системе PKI, вопрос доверия данной PKI остается за третьими лицами — пользователями системы.

[19] RFC 6482: A Profile for Route Origin Authorizations (ROAs), URL: https://www.rfc-editor.org/rfc/rfc6482

Корневой СР в качестве списка АИР охватывает все адресное пространство IPv4, IPv6 и автономных систем. С помощью этого сертификата могут быть сгенерированы сертификаты РИР в соответствии с фактически распределенным адресным пространством.

В свою очередь РИРы могут сертифицировать АИР, которые они распределяют локальным регистратурам (Local Internet Registry — LIR) или конечным пользователям, которые получают АИР непосредственно от РИР. Локальные регистратуры могут осуществлять последующее распределение, а также соответствующую сертификацию. Наконец, конечные пользователи — сети, фактически использующие адресное пространство, — также должны иметь возможность генерирования временных сертификатов для подписания вторичных объектов RPKI, например, ROA. Дело в том, что срок жизни этих объектов обычно короче, чем право на использование АИР, а их аннулирование реализуется путем отзыва сертификатов, использованных при создании соответствующих объектов. Во избежание аннулирования всех вторичных объектов и последующего их воссоздания для каждого объекта генерируется свой СР. Таким образом, аннулирование вторичных объектов может быть осуществлено независимо.

Важное следствие такого подхода — все участники RPKI являются органами выдачи сертификатов, УЦС, а значит, нуждаются в соответствующей защищенной инфраструктуре, организации процессов управления ключами и т.п.

Общая схема RPKI проиллюстрирована на рис. 45.

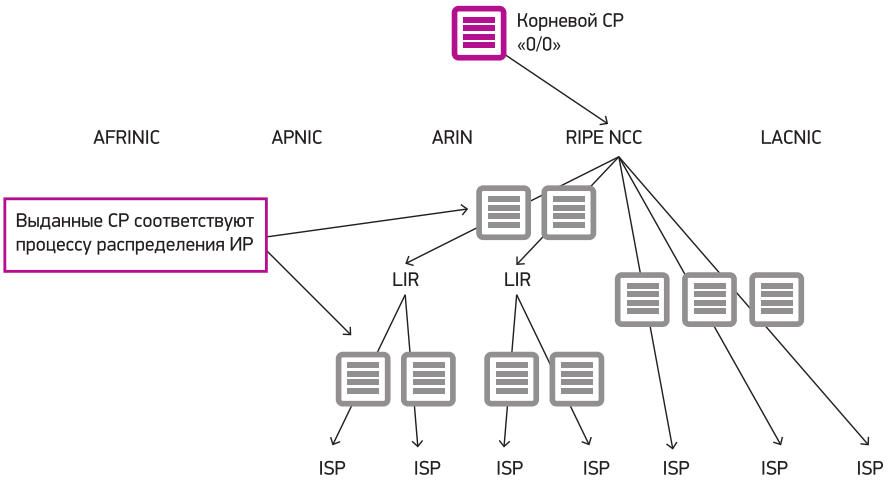

Рис. 45. Общая структура RPKI.

Использование RPKI

Одна из задач внедрения системы RPKI — предоставить возможность сетевым операторам независимо проверять достоверность сертификатов и ROA. Это важный процесс с точки зрения безопасности системы маршрутизации, и он состоит из нескольких этапов, которые мы кратко рассмотрим.

После проверки подлинности подписи ROA пользователь должен удостовериться, что ресурсы, описанные сертификатом ROA, содержат все IP-префиксы, указанные в ROA.

Следующим шагом является проверка так называемой цепи доверия, означающей, что начиная от корневого сертификата каждый последующий сертификат на пути к ROA выдан предыдущим и все сертификаты пути являются достоверными. Проверка достоверности в RPKI отличается от аналогичного процесса в PKI общего назначения: в RPKI, помимо проверки криптографической правильности сертификата, удостоверяется, что ресурсы, описанные родительским сертификатом, включают все ресурсы дочернего сертификата. Вы, наверное, заметили, что в отличие от системы PKI, используемой для веб-сайтов, валидация в RPKI производится, начиная от корневого сертификата, и предполагает проверку всего дерева. Это связано с громадным числом путей, которые требуется проверить: в случае 100-процентного внедрения системы RPKI число путей будет соответствовать числу маршрутов в глобальной таблице маршрутизации — и проверка снизу вверх просто непрактична.

Этот процесс описан в RFC 6487[20] и представлен на рис. 46.

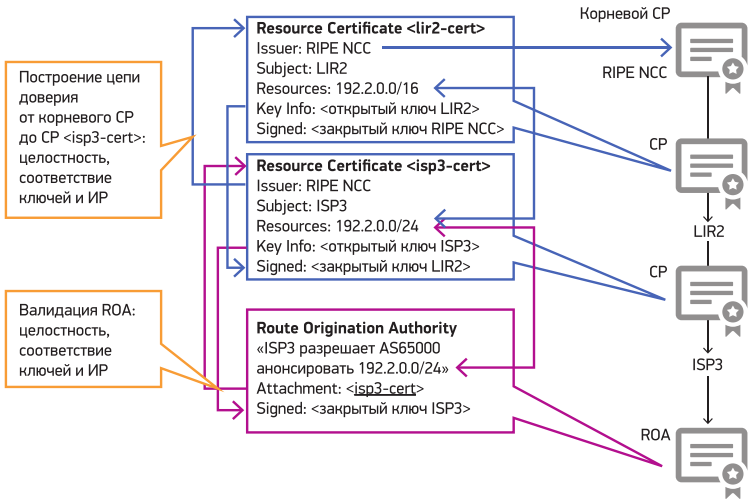

Рис. 46. Построение цепи доверия для проверки подлинности вторичных объектов и СР.

[20] RFC 6487: A Profile for X.509 PKIX Resource Certificates, URL: https://www.rfc-editor.org/rfc/rfc6487

Использование ROA возможно как для построения фильтров, так и в качестве дополнительного правила в процессе выбора пути BGP. Логично предположить, что интеграция в процесс BGP информации, полученной от системы RPKI, является более масштабируемым решением.

Архитектура такого решения схематично представлена на рис. 47. Предполагается, что сервис-провайдер хранит собственную копию всех объектов глобальной системы RPKI, проверяет их достоверность и периодически обновляет. Результирующая база данных содержит только достоверную информацию (достоверный кеш, validated cache) и может быть непосредственно использована процессом BGP маршрутизатора.

При получении очередного сообщения BGP маршрутизатор запрашивает базу данных на предмет наличия префиксов, указанных в поле NLRI сообщения BGP. Достоверность маршрута проверяется на предмет соответствия префиксов NLRI префиксам, указанным в ROA, а также номера первой AC пути (атрибут AS_PATH) номеру автономной системы, указанной в ROA как источник анонса. Если эти условия выполняются, маршрут маркируется как «valid». Если же база данных не содержит указанных префиксов, значит, система RPKI не содержит ни одного правомерного объекта ROA для этих префиксов. Причин может быть несколько. Например, просроченный сертификат в цепочке проверки подлинности ROA или отсутствие ROA как такового. Учитывая, что внедрение данной системы будет происходить постепенно, данная ситуация скорее свидетельствует, что сеть еще не использует преимущества RPKI, и такой маршрут может быть принят при отсутствии более надежного. В этом случае результатом определения достоверности маршрута будет «not-found».

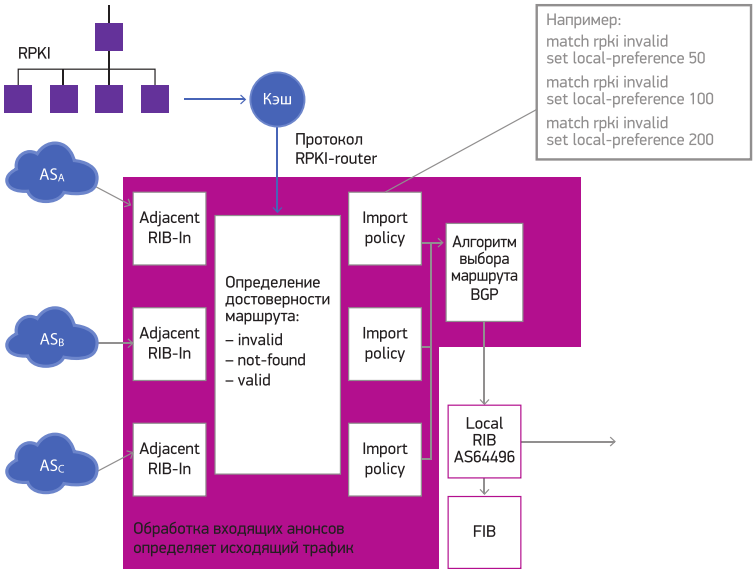

Рис. 47. Интеграция RPKI в процесс принятия решения BGP.

Другой случай — база данных содержит правомерные ROA, которые описывают префиксы, указанные в анонсе BGP, но не соответствующие автономным системам-отправителям, указанным в атрибуте AS_PATH. Возможно, это является свидетельством захвата префикса (prefix hijacking), и такой маршрут следует использовать с большой осторожностью. Скорее всего, его следует отбросить, даже если он единственный. Результат определения достоверности такого маршрута — «invalid».

Этот процесс определения достоверности маршрута получил название «валидация источника маршрута» (route origin validation, ROV).

В любом случае результат удостоверения маршрута с использованием RPKI является одним из критериев выбора маршрута в BGP наряду с другими атрибутами BGP — AS_PATH, ORIGIN, MED. В конечном счете интерпретация этой информации является частью политики маршрутизации данной сети. Например, на начальном этапе внедрения RPKI более разумной может быть политика занижения приоритета маршрутов «invalid» с помощью параметра LOCAL_PREF, как показано на рис. 45.

На сегодняшний день RPKI — наиболее технологичный способ обеспечения безопасности маршрутизации. Но даже если RPKI внедрят все операторы, это не обеспечит безопасность в полной мере. Возможность обмана в Сети останется, пока не будет поддержки функции установления подлинности пути передачи сообщения BGP — AS_PATH.

Например, злоумышленник может утверждать, что автономная система, указанная в ROA, является его клиентом: это реализуется путем присоединения номера данной АС в атрибут AS_PATH своих анонсов BGP. Поэтому RPKI не сможет полностью защитить глобальный Интернет от фальсификации адреса отправителя, атак типа Pilosov и YouTube. Но внедрение системы и, главное, применение сопутствующих технологий сервис-провайдерами позволят ограничить вред, наносимый такими атаками.

Поэтому в IETF были разработаны стандарты, обеспечивающие возможность проверки криптографической подлинности анонса маршрута и достоверности пути, по которому этот анонс был передан. Это расширение BGP получило название BGPsec.

Например, представим анонс маршрута 192.168/16 сетью AS_1 сети AS_2 и затем AS_3. По получении этого анонса сеть AS_4 сможет удостовериться, что AS_1 является правомочным «владельцем» маршрута 192.168/16, который она анонсировала сети AS_2; что сеть AS_2 получила этот маршрут от сети AS_1 и передала его сети AS_3; наконец, что сеть AS_3 получила этот маршрут от сети AS_2 и передала его сети AS_4.

BGPsec

Чтобы устранить описанные выше проблемы безопасности, требуется расширить возможности самого протокола BGP. Эти расширения были стандартизованы в 2017 году. Основной спецификацией протокола BGPsec является RFC 8205[21].

Расширение функциональности BGP реализовано с помощью нового атрибута: BGPSEC_Path_Signatures. Данный атрибут содержит последовательность цифровых подписей для каждой сети (точнее, автономной системы), через которую был передан соответствующий анонс маршрута.

Чтобы лучше понять, как это работает, представим три сети: AS_1, AS_2 и AS_3 (рис. 48). Допустим, что AS_1 анонсирует маршрут 192.168/16 сети AS_2. При использовании BGPsec этот анонс будет содержать атрибут BGPSEC_Path_Signatures, состоящий из префикса 192.168/16, сети-источника AS_1 и сети-пира, которой этот маршрут передан, — AS_2. Вся эта информация заверена подписью AS_1. В свою очередь, когда сеть AS_1 передаст этот анонс сети AS_2, та также заверит своей подписью информацию, полученную от AS_1, плюс номер автономной системы-пира, которой передается анонс, — AS_3. Заметим, что с точки зрения передачи анонсов от граничных маршрутизаторов требуется только наличие секретного ключа своей автономной системы для подписания атрибута BGPSEC_Path_Signatures.

Если AS_3 захочет удостовериться в подлинности полученного анонса, ей придется сделать несколько проверок. Сначала она должна будет убедиться, что AS_1 действительно авторизована для анонсирования этого маршрута держателем соответствующего адресного пространства. Для этого AS_3 проверит наличие и достоверность соответствующего объекта ROA системы. Затем эта сеть последовательно проверит подписи, содержащиеся в атрибуте BGPSEC_Path_Signatures полученного анонса, чтобы убедиться в их подлинности и соответствии пути прохождения анонса.

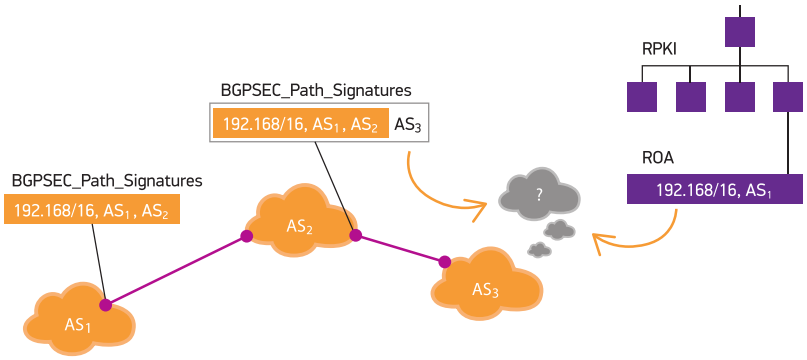

Рис. 48. Схема работы BGPsec.

[21] RFC 8205: BGPsec Protocol Specification, URL: https://www.rfc-editor.org/rfc/rfc8205

Чтобы обеспечить возможность проверки подписей граничных маршрутизаторов, BGPsec определяет дополнительный тип сертификата, связывающего открытый ключ маршрутизатора с номером его автономной системы. Этот сертификат является дочерним по отношению к сертификату RPKI данной АС. Таким образом, при верификации анонса проверяющая сторона сможет создать цепочку доверия, используя глобальную систему RPKI.

Как и в случае с RPKI, решение, которое примет сеть AS_3 по результату проверки, зависит от политики безопасности данной сети. Например, при наличии строгой политики рассматриваемый путь будет удален из таблицы маршрутизации, а более гибкая политика лишь уменьшит предпочтительность этого маршрута.

Пока работа над протоколом BGPsec находится на стадии стандартизации. Основная спецификация протокола «BGPsec Protocol Specification»[22] уже опубликована как стандарт RFC. Но пройдет немало времени, прежде чем этот стандарт и связанные с ним спецификации будут воплощены в оборудовании и затем внедрены сетевыми операторами. А пока операторы могут сфокусироваться на использовании результатов первой фазы работы — RPKI.

Более прагматичные решения

По мнению многих операторов, широкое внедрение BGPsec не является реалистичным в обозримом будущем. Производители сетевого оборудования тоже не спешат реализовать эту функциональность.

Во-первых, постепенное внедрение проблематично с экономической точки зрения, так как любая AS, которая оказывается на пути и не поддерживает BGPsec, сводит на нет все усилия и преимущества, делая валидацию невозможной. А таких AS на начальном этапе будет очень много. Во-вторых, BGPsec не защищает от другого вида атак – так называемых утечек маршрута (route leak).

Мы уже упоминали этот тип атак, давайте рассмотрим «утечку маршрута» более подробно. Суть этой атаки можно проиллюстрировать на простом примере. Допустим, сеть AS_1 (рис. 49) является клиентом двух провайдеров транзита – AS A и AS B. В случае, если этот клиент реанонсирует маршрут к Google Public DNS (8.8.8.0/24), полученный от AS B, другому провайдеру – сети AS A, а последняя примет его, то произойдет утечка маршрута. В результате весь трафик клиентов AS A к Google будет направлен через клиентскую сеть AS_1, вероятнее всего, с катастрофическими последствиями для последней.

[22] RFC 8205: BGPsec Protocol Specification,
URL: https://www.rfc-editor.org/rfc/rfc8205

Рис. 49. Утечка маршрута сетью-клиентом.

Без дополнительных проверок со стороны сети AS A так и произойдет, поскольку типичная политика маршрутизации предполагает, что маршруты, полученные от клиентов, имеют предпочтение. При этом, конечно, политика предполагает, что клиент анонсирует только свои собственные сети, а не чужие сети где-то в Интернете, о которых он узнал от другого провайдера. Другими словами, в данном случае мы имеем дело с нарушением политики, а не манипуляцию пути. BGPsec в данной ситуации бессилен.

Но если на BGPsec рассчитывать не приходится, какие возможности улучшения защищенности маршрутизации есть в распоряжении сетевых операторов?

Peer-lock

Peer-lock (peer-locking) – это механизм, предложенный Джобом Снейдерсом (Job Snijders), в то время работавшим в NTT, обеспечивающий определенную защиту от утечек маршрутов, поступающих от пиров. В отличие от утечки, вызванной сетью-клиентом, как показано на рис. 47, утечки такого типа происходят в результате нарушения пиринговой политики – пиры должны анонсировать только собственные сети и сети своих клиентов. Если пир анонсирует сети, полученные от других пиров, это приводит к утечке маршрута.

Механизм Peer-lock основан на активной координации и требует кооперации от сети-пира, которая желает стать «защищенной» (“protected ASN”). В самом

простом варианте анонсы, включающие эту AS, позволены только через прямое пиринговое соединение. Анонсы, содержащие эту AS, но полученные от других пиров, будут отброшены.

Схема работы peer-lock показана на рисунке 50. Сеть NTT в этом примере является провайдером, обеспечивающим механизм. Защищенная сеть «peer A» указывает на возможность альтернативного транзита через сеть «peer B». В этом случае анонсы, содержащие B в пути будут приняты NTT только непосредственно от A или от B. Анонс с сетью B, полученный от сети C, будет отброшен. Таким образом, даже если сети A и C обмениваются маршрутами в режиме пиринга, peer-lock предотвратит возможную утечку маршрута через C.

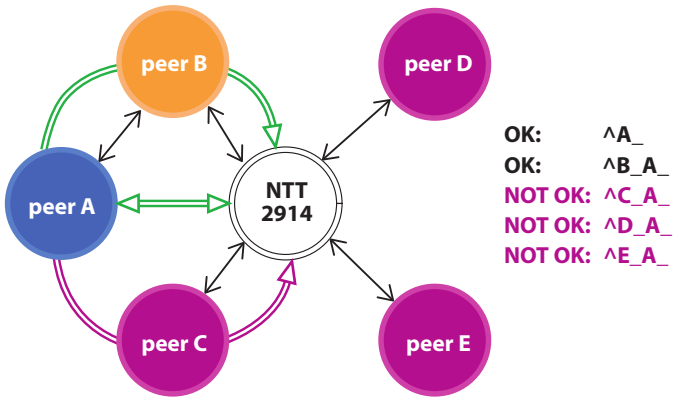

Рис. 50. Механизм работы peer-lock.
Источник: Job Snijders, https://archive.nanog.org/sites/default/files/Snijders_Everyday_Practical_Bgp.pdf

Механизм может быть адаптирован с учетом географической распределённости сетей[23].

Была также предпринята попытка расширить и автоматизировать механизм с использованием атрибута community BGP[24], но дальше изначального предложения дело не пошло. Некоторые из этих идей были использованы в другом предложении[25], работа над которым также не привела к конечному результату.

[23] https://instituut.net/~job/peerlock_manual.pdf
[24] https://datatracker.ietf.org/doc/draft-heitz-idr-route-leak-community
[25] https://datatracker.ietf.org/doc/draft-ietf-grow-route-leak-detection-mitigation

ASPA

ASPA, сокращенное от Autonomous System Provider Authorization, или Авторизация провайдеров автономной системы, позволяет автономной системе указать ее непосредственных провайдеров транзита. Подход основан на регистрации объектов ASPA, криптографически сертифицированных с помощью системы RPKI.

Согласно интернет-драфту «Verification of AS_PATH Using the Resource Certificate Public Key Infrastructure and Autonomous System Provider Authorization»[26], ASPA – это объект с цифровой подписью, который связывает для выбранного адресного пространства AFI (Address Family Identifier – идентификатор семейства адресов) (например, IPv4 или IPv6) набор номеров AS провайдеров с номером AS клиента (с точки зрения анонсов BGP, а не бизнеса) и подписывается владельцем AS клиента. ASPA подтверждает, что владелец клиентской AS (CAS) авторизовал набор провайдеров AS (Set of Provider ASes, SPAS) для дальнейшего анонсирования сетей клиента, например, провайдерам выше по потоку или пирам.

В этом смысле механизм похож на только что рассмотренный нами peer-lock. Однако peer-lock трудно автоматизировать, и информация о возможных транзитных провайдеров предназначена для конкретной сети, реализующей peer-lock, а не является частью глобального репозитория, как это предполагается в ASPA.

В отличие от BGPsec, который позволяет проверить всю цепочку пути анонса, ASPA позволяет определить валидность фрагментов пути. Процедура проверки сводится к нескольким шагам, описанным ниже. При этом предполагается, что проверяющая сторона, например, сеть, осуществляющая фильтрацию неверных анонсов, имеет доступ к кешу всех криптографически правильных объектов ASPA. Допустим, проверяющей стороне требуется проверить валидность фрагмента пути AS_PATH (AS_1, AS_2, AFI), а именно, предположение, что в случае, если AS_1 является клиентской AS, AS_2 является законным провайдером транзита для адресного пространства AFI.

1. Проверяющая сторона запрашивает из кеша все объекты ASPA, у которых CAS имеет значение AS1. Все SPAS, входящие в объединенное множество, являются потенциальными провайдерами.
2. В случае, если это множество пусто, проверка заканчивается с результатом «No Attestation» (Неизвестно).
3. Если AS2 является членом этого множества, проверка заканчивается с результатом «Provider+» (Верно).
4. В противном случае проверка заканчивается с результатом «Not Provider+» (Неверно).

При проверке полного пути все сегменты (hops) пути должны иметь значение «Provider+».

[26] https://datatracker.ietf.org/doc/draft-ietf-sidrops-aspa-verification/

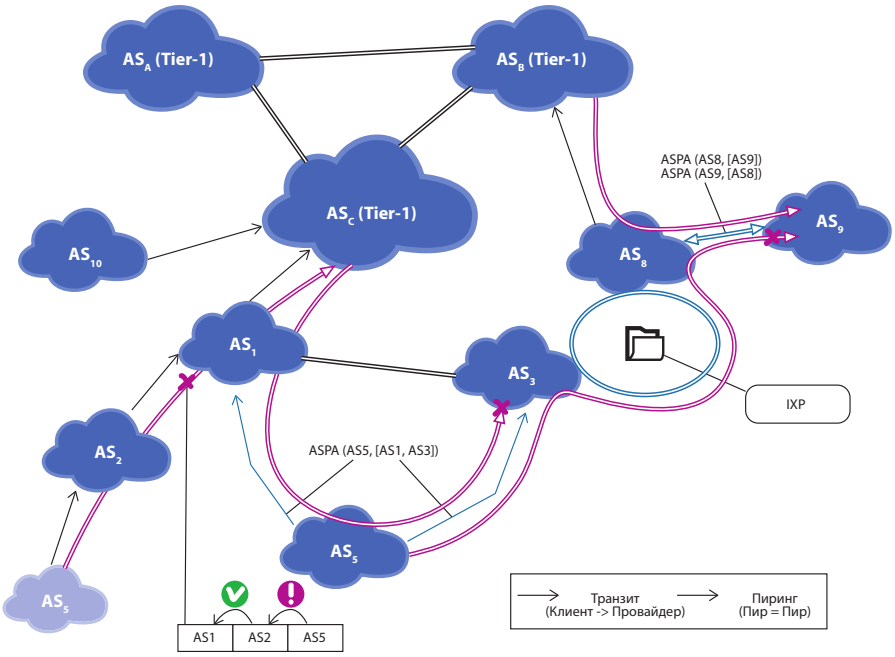

Рис. 51. ASPA позволяет предотвратить замаскированные захваты и утечки маршрута.

Как уже говорилось ранее, одной из задач защиты пути является предотвращение так называемых замаскированных захватов маршрута, когда атакующий представляется провайдером атакуемой системы. Так, на рис. 51 AS_2 может модифицировать AS_PATH анонсов сетей AS_5 провайдеру AS_1, добавив AS_5 в качестве клиента. Проверка с помощью ROA (ROV) в этом случае не поможет, поскольку формально AS_5 является источником анонсированных маршрутов. Хотя такой маршрут является менее «конкурентоспособным» в силу увеличенной длины, для многих сетей он может все же являться лучшим маршрутом, что приведет к успешной атаке.

Если же AS5 зарегистрирует своих транзитных провайдеров с помощью ASPA (AS_5, [AS_1, AS_3]), атакующему будет гораздо сложнее представиться провайдером AS_5 – фрагмент пути (AS_2, AS_5) будет отмечен как неверный в процессе проверки ASPA (см. рис. 51). Чем больше фрагментов пути будут зарегистрированы с помощью ASPA, тем меньше шансов у атакующего провести успешный захват маршрута.

Поскольку ASPA указывает на отношения между сетями (клиент-провайдер) и связанную с ними стандартную политику маршрутизации, этот механизм можно использовать для обнаружения утечек маршрутов. Так, AS_3 сможет установить, что AS_5 «протекла» с маршрутами, полученными от другого провайдера транзита AS_1, а AS_9 сможет остановить утечку маршрутов через последовательных пиров. Эти примеры приведены на рис. 51.

Важным свойством ASPA является то, что выгоды от его использования растут пропорционально масштабам внедрения этой технологии – в отличие от BGPsec, который требует значительного, если не тотального внедрения, прежде чем будет получен ощутимый эффект.

Роли участников сессии BGP

Как мы уже упомянули, атаки типа «утечки маршрута» являются результатом нарушения стандартной политики маршрутизации и потому защита BGP в этих случаях не работает.

Однако если BGP указать на тип взаимоотношений между сетями, участвующими в сессии, это может служить предохранителем против нарушения стандартной политики. Эта идея была стандартизована в IETF в 2022 году в спецификации RFC 9234[27].

Идея основана на том, что на начальном этапе при создании BGP-сессии между двумя сетями стороны должны договориться о возможностях, которые они обе поддерживают. К таким возможностям относится поддержка 4-октетных номеров автономных систем, BGPsec и т.п. Стандарт предлагает добавить еще одну возможность – определение роли каждой из сторон. В качестве таких ролей предложены следующие: Провайдер (Provider), Сервер маршрутов (Route Server, RS), Клиент сервера маршрутов (RS Client), Клиент (Customer), Пир (Peer).

С каждой ролью связана стандартная политика, так, например, стандартная политика в отношении Провайдера с Клиентом предусматривает, что Провайдер принимает все анонсы Клиента (и сетей его собственных Клиентов), а сам анонсирует глобальную связность (маршрут default или полную таблицу маршрутизации). Соответственно, Клиент зеркалирует эту политику.

Определение роли позволяет, во-первых, предотвратить ситуацию, когда стороны играют несовместимые роли (например, обе сети считают себя Провайдерами, типичный случай появления утечек маршрута), а во-вторых, предотвратить нарушение стандартных политик, соответствующих определенным ролям.

[27] RFC 9234: Route Leak Prevention and Detection Using Roles in UPDATE and OPEN Messages, URL: https://www.rfc-editor.org/rfc/rfc9234

Совместимыми являются следующие роли:

Местная AC	Удаленная AC
Provider	Customer
Customer	Provider
RS	RS-Client
RS-Client	RS
Peer	Peer

При обнаружении несоответствия ролей процесс создания BGP-сессии прерывается с ошибкой «Role Mismatch».

Стандарт также предлагает использование нового атрибута – «Only to Customer» или OTC. Этот атрибут представляет собой номер AC, который устанавливается в соответствии с процедурой, описанной ниже.

После того как роли сетей определены, этот атрибут может предотвратить анонсы, противоречащие стандартной политике маршрутизации для этой роли. Так, при получении маршрута от удаленной AC, сеть должна выполнить следующие проверки:

- Если маршрут с атрибутом OTC получен от Клиента или RS-Клиента, то это утечка маршрута, и такой анонс считается неприемлемым и должен быть отброшен.
- Если маршрут с атрибутом OTC получен от Пира, и атрибут имеет значение, не равное номеру AC этого Пира, то это утечка маршрута, и такой анонс считается неприемлемым и должен быть отброшен.
- Если маршрут получен от Провайдера, Пира или Сервера маршрутов RS, а атрибут OTC отсутствует, то он должен быть добавлен со значением, равным номеру удаленной AC.

Следующая процедура применяется к обработке атрибута OTC при анонсировании маршрута:

- Если маршрут должен быть анонсирован Клиенту, Пиру или RS-Клиенту (когда отправителем является RS), а атрибут OTC отсутствует, то при анонсировании маршрута атрибут OTC должен быть добавлен со значением, равным номеру местной AC.
- Если маршрут уже содержит атрибут OTC, он не должен анонсироваться Провайдерам, Пирам или Серверам маршрутов RS.

Нужно отметить, что, как и многие атрибуты BGP, OTC не защищен от модификаций (BGPsec защищает только атрибут AS_PATH). Поэтому описанный подход позволяет избежать ошибок конфигурации, но не умышленных атак. В то же время, по оценке специалистов, большинство инцидентов возникают как раз вследствие ошибок конфигурации, и в этом смысле роли и OTC являются эффективным средством.

Вопросы внедрения RPKI

Решения безопасности на базе RPKI зависят от уровня внедрения базовых компонентов – сертификации номерных ресурсов и создания ROA, соответствующих анонсам маршрутов.

Как видно из графиков на рисунках 52-55, наблюдается значительный рост числа ROA во всех регионах, как по количеству зарегистрированных префиксов, так и по размеру адресного пространства, которое они покрывают. По всем параметрам лидирует регистратура RIPE NCC, хотя ARIN не уступает по размеру адресного пространства IPv4 с зарегистрированными ROA.

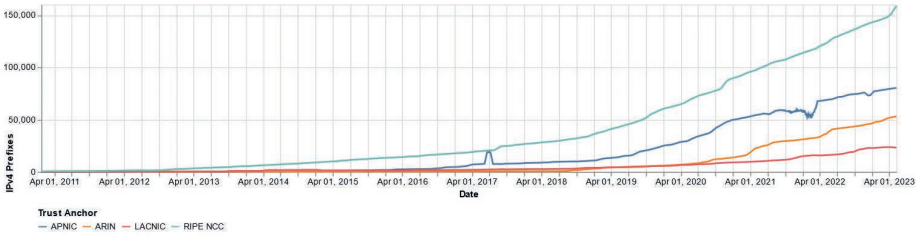

Рис. 52. Число префиксов IPv4 с зарегистрированными ROA.
Источник: https://certification-stats.ripe.net

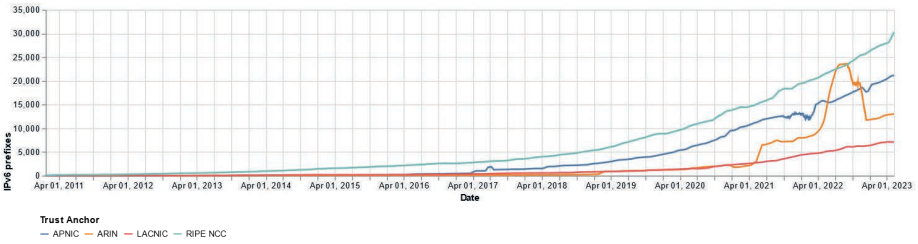

Рис. 53. Число префиксов IPv6 с зарегистрированными ROA.
Источник: https://certification-stats.ripe.net

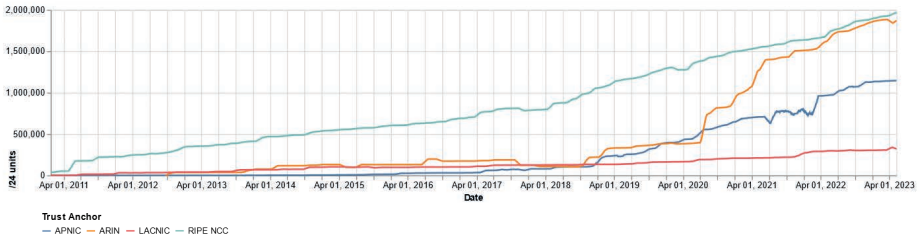

Рис. 54. Адресное пространство IPv4 (в единицах /24) с зарегистрированными ROA.
Источник: https://certification-stats.ripe.net

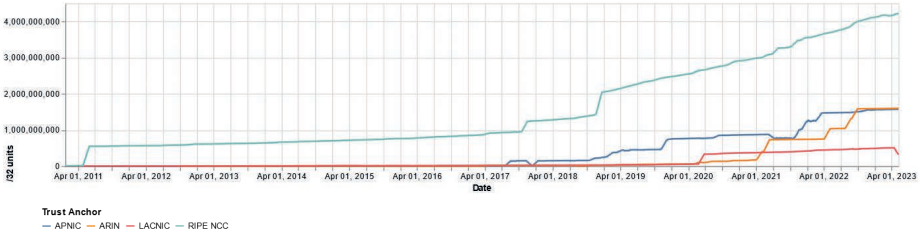

Рис. 55. Адресное пространство IPv6 (в единицах /24) с зарегистрированными ROA.
Источник: https://certification-stats.ripe.net

По состоянию на май 2023 года большая часть всех анонсов (67%) имела соответствующую ROA. Это означает, что 67% всех анонсов могут быть защищены с помощью RPKI в случае, если сети используют фильтрацию на основе этой технологии.

Здесь мы подходим ко второму параметру внедрения – уровню валидации источника маршрута (т.н. ROV) на основе сравнения полученных маршрутов с соответствующими ROA, о чем мы говорили в предыдущих разделах. Ведь для усиления защищенности системы маршрутизации требуются оба фактора – регистрация ROA владельцем адресного пространства и внедрение ROV операторами сетей.

Согласно исследованиям APNIC Labs, средний мировой уровень фильтрации на основе ROV не превышает 24% (по состоянию на май 2023 года). В этом смысле интересно сравнить карту внедрения ROA и использования ROV в мире.

Будем надеяться, что по мере более широкого покрытия анонсируемых префиксов ROA также будет расширяться использование этой технологии для фильтрации неверных маршрутов, делая общую систему более защищенной.

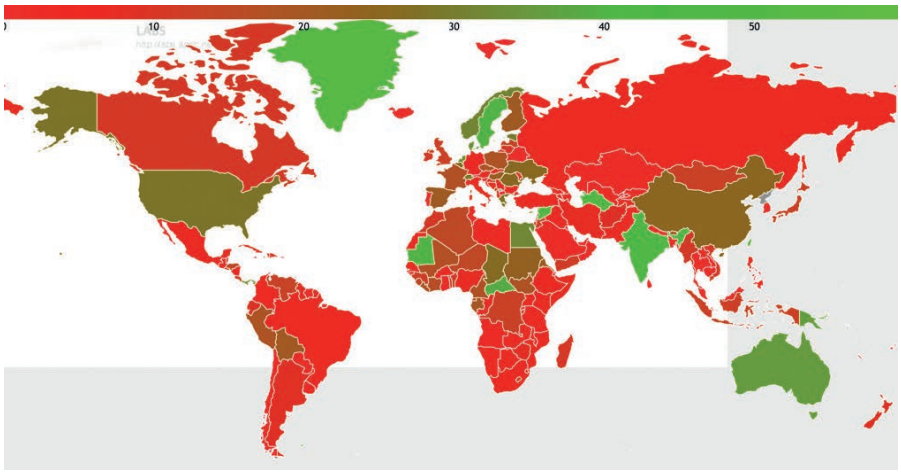

Рис. 56. Степень фильтрации маршрутов на основе ROV по странам.
Источник: https://stats.labs.apnic.net/rpki/

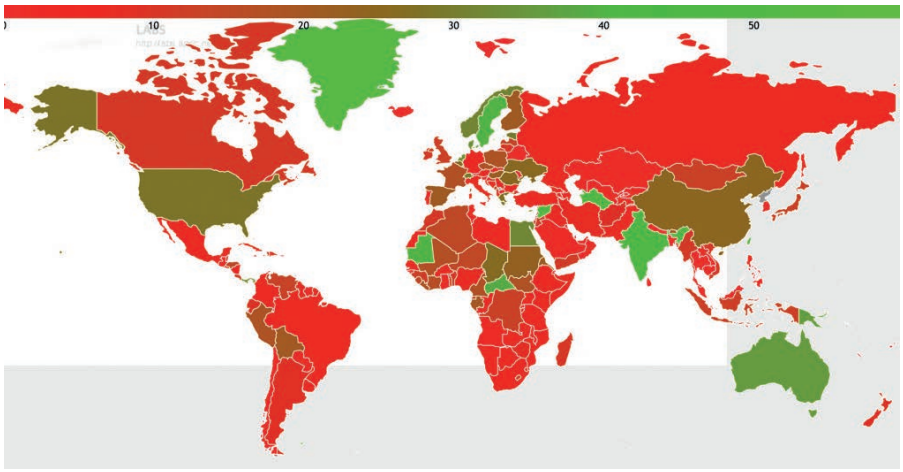

Рис. 57. Процент префиксов с зарегистрированными ROA по странам.
Источник: https://stats.labs.apnic.net/roa/

Вопросы обеспечения качества передачи в Интернете

Качество сеанса связи в Интернете непредсказуемо, но можно определенно говорить о его постоянном улучшении. Телеконференция Skype не уступает, а подчас и превосходит по качеству традиционную телефонную связь. Широкое распространение онлайн-игр и других интерактивных приложений говорит скорее о колоссальных возможностях Интернета, чем о его ограничениях. И все

же отсутствие базовой инфраструктуры, гарантирующей качество, подчас вселяет некоторый дискомфорт.

Во многом это ложное чувство. Хотя качество в Интернете и не обеспечивается на уровне базовой пакетной передачи, эта задача решается на многих уровнях, усиливая положительный эффект. Ключ к решению проблемы качества лежит в методах более эффективного использования имеющейся пропускной способности, о которых мы и поговорим в этом разделе.

Традиционная телефония и Интернет

Если взять стандартный спектр человеческого голоса и оцифровать его, используя простейший алгоритм импульсно-кодовой модуляции[28] (англ. PCM), то для передачи этой информации потребуется канал с пропускной способностью 64 кбит/с. Первые цифровые голосовые каналы обладали именно такой емкостью. Конечно, 64 кбит/с — это роскошь, и при использовании сегодняшних эффективных кодеков требуемую полосу можно значительно уменьшить. Например, кодек EFR (Enhanced Full Rate)[29], широко используемый в мобильной связи, позволяет «сжать» голос до 12,2 кбит/с. А один из базовых кодеков для приложений IP-телефонии, G.723.1[30], и вовсе использует полосу в 5,3 кбит/с!

В традиционной телефонии каждая сеть, через которую проходит вызов, должна сначала зарезервировать канал, а затем предоставить эту емкость для соединения или разговора, если вызываемый абонент снял трубку. В результате соединение между двумя говорящими имеет гарантированные пропускную способность и другие параметры качества, например, задержку. Задачей маршрутизации вызова и создания соединения в телефонии занимается система сигнализации ОКС-7 (англ. SS7)[31].

С другой стороны, если хотя бы одна из сетей перегружена и не может обеспечить канальную емкость, все предыдущие резервирования должны быть отыграны назад, и звонок не состоится. Однако достаточно простая топология и связность телефонных сетей, а также ограниченный спектр предлагаемых услуг — преимущественно голосовая связь или соединения одинаковой пропускной способности — позволяют обеспечить достаточно точное планирование емкостей и минимизировать число отказов.

Другой особенностью является то, что телефонные сети являются синхронными; по существу, они не используют буферизацию. Это означает, что задержка передачи между абонентами определяется в основном скоростью распространения сигнала или, другими словами, расстоянием между абонентами.

---

[28] https://ru.wikipedia.org/wiki/Импульсно-кодовая_модуляция
[29] http://ru.wikipedia.org/wiki/GSM-EFR
[30] http://ru.wikipedia.org/wiki/G.723.1
[31] https://ru.wikipedia.org/wiki/ОКС-7

Интернет же отличается коренным образом. Топология и связность сетей, как правило, весьма разветвленная и нерегулярная.

«Стандартной услуги» как таковой нет — различные приложения имеют различные требования к пропускной способности и качеству сети. Также Интернет является сетью пакетной передачи, где IP-дейтаграммы асинхронно передаются узлами-маршрутизаторами по каналам со значительной вариацией пропускной способности. Это, в свою очередь, выражается в нерегулярности трафика и требует использования буферов для сглаживания «всплесков». Как мы увидим дальше, буферизация и управление очередями пакетов являются важными факторами качества связи.

Качественные решения

Фундаментальные технологии и протоколы Интернета, такие как IP, TCP или BGP[32], были разработаны в соответствии с требованиями пакетной передачи данных и высокой стойкости в отношении отказа отдельных узлов или целых сетей. Эти технологии обеспечивали простую, но универсальную услугу — передачу данных в режиме «best effort», не гарантирующем никаких параметров качества, а иногда даже и того, что пакеты будут доставлены получателю. Сеть не давала предпочтения какому-либо приложению — все пакеты подвергались одинаковой обработке.

Предоставить такую услугу было относительно просто: не имели значения ни пропускная способность и производительность сети, ни инфраструктура нижнего уровня. Не требовалась синхронизация между отдельными сетями — основным критерием являлась поддержка протокола IP. Поэтому и «подключиться» к Интернету, стать частью сети сетей было так же просто — что и явилось катализатором его быстрого роста. Приложениям также не требовалось спрашивать никакого специального «разрешения» у Сети — то, что происходило выше уровня IP, было делом договоренности между отправителем и получателем, клиентом и сервером и т.п. Это и сегодня является основой громадного инновационного потенциала Интернета.

На первый взгляд, услуга best effort была довольно ограниченной — особенно во времена раннего Интернета, когда «традиционные» сети гарантировали достоверную доставку и высокие параметры качества передачи. Однако большинство тогдашних приложений Интернета были весьма «эластичными», а именно малочувствительными к параметрам передачи. Например, обмен файлами с помощью протокола FTP или отображение веб-страницы могло занять секунды, минуты или часы, но фундаментального значения для приложения это не имело.

Другое дело — приложения реального времени: голосовая связь и видеоконференции. Они требуют определенных параметров качества, ниже которых

[32] http://ru.wikipedia.org/wiki/TCP/IP

эти приложения работать просто не могут. Но для этих приложений существовали свои сети — традиционные телефонные и ISDN[33].

Однако дешевизна и растущее распространение Интернета заставило задуматься о возможности его использования также и для «качественных» приложений — видео и голоса. Это, в свою очередь, привело к разработке «качественных» решений.

IntServ, или Интеграция служб

В середине 90-х гг. прошлого века в IETF[34] была разработана концепция, получившая название Integrated Services, или IntServ (интегрированные службы). Как было указано в документе RFC 1633[35], описывавшем архитектуру IntServ, «это расширение [архитектуры Интернета] необходимо для удовлетворения растущих потребностей в услугах реального времени для широкого диапазона приложений, включая телеконференции, удаленные семинары и распределенное моделирование». Также IntServ должны были обеспечить мультиплексирование различных классов трафика в сети, что позволило бы операторам предоставлять различные изолированные друг от друга услуги, используя общую сетевую инфраструктуру.

IntServ определяет два основных класса приложений: приложения реального времени, чувствительные к задержке, и «эластичные» приложения, для которых не так важно, когда именно будут получены данные, а точнее, дейтаграммы.

Чувствительность к задержке у приложений реального времени также различается. Например, для передачи голосового или видеопотока приложению необходимо знать максимально возможную задержку для обеспечения адекватной буферизации. В противном случае поток будет прерываться и его качество деградирует. Для таких приложений IntServ предлагает так называемую гарантированную услугу, при которой задержка не может превысить заданную величину.

Менее чувствительные приложения смогут довольствоваться так называемым предсказуемыми услугами, дающими среднестатистическую задержку. Предполагается, что стоимость таких услуг будет значительно дешевле, так как этот подход позволяет достичь гораздо большей утилизации сети.

Наконец, «эластичные» приложения могут продолжать пользоваться «обычным» Интернетом с его услугой best effort.

---

33 http://ru.wikipedia.org/wiki/ISDN

34 http://www.ietf.org

35 RFC 1633: Integrated Services in the Internet Architecture: an Overview, URL: https://www.rfc-editor.org/rfc/rfc1633

Хотя сам подход казался привлекательным, он требовал двух существенных изменений в архитектуре и функционировании Интернета. Появилась необходимость требования контроля доступа и резервирования.

Действительно, для предоставления гарантий по задержке и пропускной способности сетевые ресурсы должны быть первоначально зарезервированы по всему пути передачи данных от источника к получателю (и обратно, поскольку большинство потоков являются дуплексными). Такой процесс очень напоминает резервирование канала в телефонии, только вариация параметров этих каналов значительно больше. Это, в свою очередь, потребовало внедрения дополнительного сигнального протокола резервирования (такой протокол был разработан — Resource ReSerVation Protocol (RSVP)[36]) и модификации протоколов внутри межсетевой маршрутизации.

Далее: прежде чем сеанс связи сможет состояться, приложение должно «заключить контракт» с сетью, в соответствии с которым сеть будет обеспечивать передачу данных. Этот «контракт» предусматривает, что приложение не выйдет за рамки установленных параметров, а сеть обеспечит входной контроль.

Схематично архитектура IntServ показана на рис. 58.

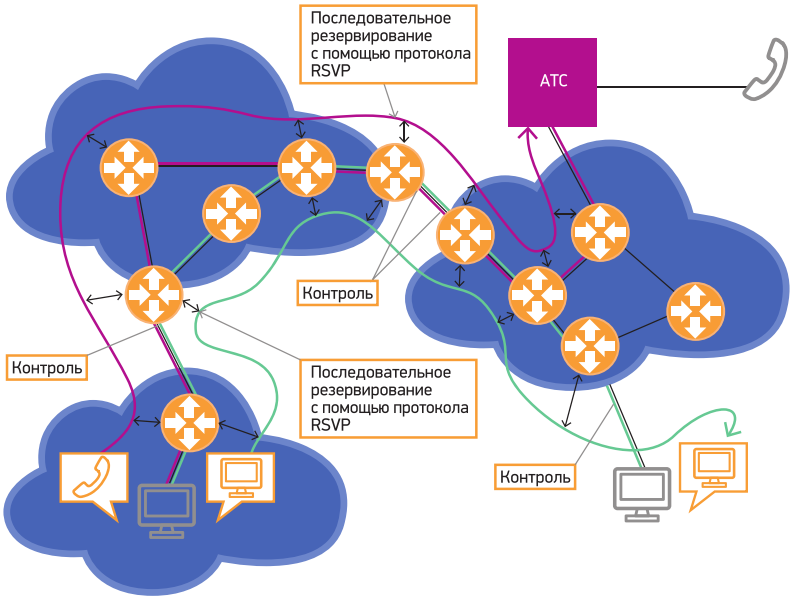

Рис. 58. Архитектура IntServ. На схеме представлено резервирование двух каналов с различными параметрами качества: для видеоконференции (зеленый цвет) и голосовой связи (красный цвет).

[36] RFC 2205: Resource ReSerVation Protocol (RSVP), URL: https://www.rfc-editor.org/rfc/rfc2205

Даже не вдаваясь в подробности, можно заметить, что уже на техническом уровне внедрение IntServ означало существенное усложнение архитектуры Сети. В экономическом смысле это влекло за собой существенное удорожание инфраструктуры для поддержки новой функциональности. Бизнес-отношения и система взаиморасчетов между сетевыми операторами должны были подвергнуться коренному пересмотру.

Наконец, технологии IntServ следовало внедрять и поддерживать глобально, во всем Интернете. Иначе — чего стоят островки качества в океане услуг «best effort»? Как и в случае многих других глобальных технологий, преимущества от их внедрения становятся ощутимыми, только когда они получают значительное распространение, — это своего рода замкнутый круг, который нелегко разорвать. Другими словами, даже одной из перечисленных проблем было достаточно, чтобы поставить под сомнение будущее предлагаемого решения. В случае с IntServ решение осталось в основном на бумаге.

DiffServ: разделяй и властвуй

Однако идея поддержки качества в глобальной инфраструктуре Интернета была слишком заманчивой, и IETF сделал вторую попытку. Началась разработка другого подхода, цель которого — обеспечение относительного, а не абсолютного, как в случае с IntServ, качества передачи. Другими словами, приложениям реального времени гарантируется определенная емкость, в рамках которой различные потоки конкурируют между собой. Эта архитектура была названа Differentiated Services, или DiffServ.

Вместо резервирования на уровне потока/приложения в DiffServ контроль производится на уровне достаточно статичных агрегированных «профилей» трафика. Классификация пакетов и их принадлежность к тому или иному профилю определяется по полю IP Type of Service (TOS), исторически оставшемуся в заголовке IP-пакета.

Соответственно, соглашение между различными сетями должно включать и договоренность о параметрах ограниченного числа профилей. При этом на входе в сеть производится контроль и возможное кондиционирование трафика для каждого профиля, которое включает буферизацию или даже отброс пакетов, если агрегированный поток превышает договоренные параметры.

Масштабируемость данного подхода, безусловно, выше по сравнению с IntServ. Еще важнее, что DifServ не требует динамического резервирования и сохранения состояния для каждого узла сети и каждого потока, проходящего через нее. Однако неразрешимым вопросом остается проблема предоставления определенного качества индивидуальному приложению. Трудно представить, что динамические требования многообразных приложений Интернета можно уложить в прокрустово ложе статической конфигурации нескольких профилей.

В результате и это решение не вызвало большого энтузиазма среди операторов. Ресурсы и инвестиции, требуемые для внедрения «качественных» решений, нашли лучшее применение в увеличении канальной емкости, благо и стоимость этих каналов к концу 1990-х значительно упала.

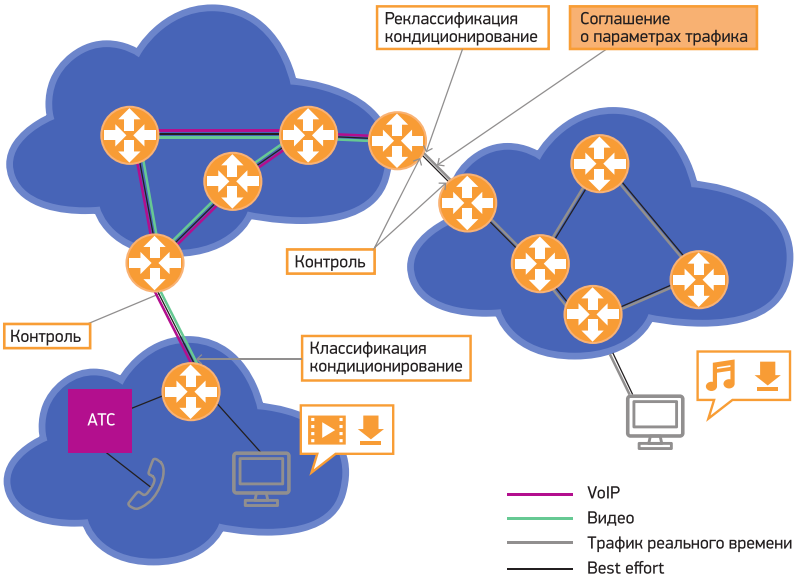

Рис. 59. Система DiffServ, позволяющая управлять качеством ограниченного числа «профилей» трафика. На схеме: голосовой трафик, видео и обычный трафик best effort.

Заметим, что концепция DiffServ используется для оптимизации трафика во внутренней инфраструктуре сети оператора, и здесь производители оборудования всегда рады помочь с широким набором возможностей, однако задача обеспечения сквозного гарантированного качества в Интернете по-прежнему остается мечтой. О полезных применениях DiffServ мы поговорим чуть позже.

Скрытые резервы

Колоссальный рост доступной канальной емкости и производительности сетей отодвинул проблему качества на задний план. Приложения реального времени также стали «умнее» и адаптивнее. Но по мере того, как опорные сети освобождались от заторов, узкое горлышко начало проявляться ближе к самому пользователю. И проблема оказалась в самом неожиданном месте — в неоправданно больших буферах устройств передачи, собственно и призванных бороться с перегрузками в сети. Феномен этот получил название bufferbloat, или «раздутый буфер».

Bufferbloat

Чтобы понять, в чем же тут дело, необходимо посмотреть, как работает фундаментальный протокол передачи данных в Интернете — транспортный протокол TCP[37].

Поскольку TCP — протокол достоверной доставки, то передача данных отправителем основана на получении подтверждения от получателя. Отсутствие подтверждения позволяет отправителю заключить, что произошла потеря пакетов, и произвести повторную передачу. Однако, если отправитель будет вынужден ожидать подтверждения каждого пакета перед передачей очередного, передача данных станет неэффективной и не сможет достичь максимальной скорости — ведь пока пакет достигнет получателя, а затем подтверждение проделает обратный путь, канал будет простаивать.

Современный дизайн TCP оптимизирован для максимальной производительности. Отправитель пытается заполнить канал данными, чтобы минимизировать простои. В то же время протокол должен эффективно работать и в случае возможной потери данных, когда требуется повторная передача.

Для этого в TCP используется понятие «окна», определяющего объем данных, которые могут быть переданы без получения подтверждения. Размер окна выбирается из следующих соображений. Для получения подтверждения на переданный пакет потребуется как минимум время для путешествия данных туда и обратно, так называемые round-trip time (RTT). Обозначим пропускную способность канала как BW (bandwidth), и тогда объем данных, определяемый произведением RTT x BW, целиком заполнит канал в интервале между подтверждениями. В этом случае отправитель будет передавать данные без остановки и канал будет до предела заполнен пакетами. Например, если RTT = 100 мс, а скорость передачи 100 Мбит/с, то окно равняется 10 Мбит. Это означает в идеальном случае, что в тот самый момент, когда отправитель передаст последний бит, будет получено подтверждение и передачу можно будет продолжать.

Проблема заключается в том, что в реальности отправитель не знает ни эффективной пропускной способности пути между ним и получателем, ни RTT. Для поиска правильных параметров в TCP существует режим так называемого медленного старта (slow start), когда изначальное окно задается размером один пакет, а затем экспоненциально растет с получением каждого последующего подтверждения. Это происходит до тех пор, пока не обнаруживается потеря пакетов, после чего отправитель уменьшает скорость передачи и переходит в режим «избежания перегрузки» (congestion avoidance).

Если мы взглянем на путь передачи данных между отправителем и получателем, то увидим, что, скорее всего, он проходит через несколько физических сетей,

[37] http://ru.wikipedia.org/wiki/TCP

каждая — со своей канальной емкостью и уровнем загрузки. Очевидно, что эффективная пропускная способность этого пути будет определяться участком с самой низкой пропускной способностью. Для сегодняшнего пользователя таким «узким горлышком» часто является домашний маршрутизатор или кабельный модем, где высокоскоростная беспроводная домашняя сеть (скажем, 54 Мбит/с) встречается с аплинком сервис-провайдера (например, 2 Мбит/с). Из-за разницы в скоростях пакеты, полученные из беспроводной сети, должны ожидать, пока очередной пакет будет передан через линк 2 Мбит/с. Для этого и используются буферы памяти. Буферы являются необходимым компонентом любой пакетной сети асинхронной передачи, они позволяют сгладить всплески трафика и улучшить общую производительность сети.

Но, к сожалению, это достигается не всегда. Наоборот: оказывается, буферы являются частью проблемы!

Логично предположить, что размер буфера должен быть сопоставим с размером максимального окна TCP — в этом случае буфер сможет удержать все пакеты максимального «всплеска», если таковой произойдет. Рекомендованный размер буфера = RTT × BW, где BW — пропускная способность исходящего канала (поскольку неизвестно, где находится истинное узкое горлышко), а в качестве RTT принято выбирать значение 100 мс — величину задержки трансатлантической передачи.

Удешевление компьютерной памяти и неполное понимание работы очередей привело к тому, что производители оборудования начали щедро начинять свои устройства буферами, иногда даже превышая рекомендованные размеры. И буферы стали частью канальной емкости, которую так хорошо умеет заполнять TCP!

И что же произошло? В случае появления затора буферы довольно быстро наполняются и переполняются, однако сигнал о потере (и, соответственно, о снижении скорости передачи) задерживается, поскольку пакет теперь проводит некоторое время в буфере, прежде чем продолжить свой путь к получателю и в виде подтверждения вернуться обратно. В качестве справки: для «очистки» буфера в 128 КБ при 3 Мбит/с аплинке (одна из типичных конфигураций в кабельных сетях, см. рис. 60) требуется 340 мс, в случае 1 Мбит/с аплинка — около 1 с.

В результате TCP не способен своевременно адаптировать скорость передачи к доступной канальной емкости, что вызывает задержки, потери и повторные передачи, способные на порядок уменьшить реальную пропускную способность.

Проблема усугубляется тем, что выбрать «правильный» размер буфера невозможно. Вариация параметров отдельных потоков данных, мультиплексируемых в канале, слишком велика и динамична для статически заданного

параметра. Более того, параметры самой физической среды могут меняться. Например, изменение местоположения ноутбука или планшета способно на порядок изменить пропускную способность в беспроводной сети. Вследствие этого и само «узкое горло» может меняться, перемещаясь от домашнего маршрутизатора в пользовательский компьютер.

Эффект bufferbloat может существенно снизить производительность сети, особенно для приложений, чувствительных к задержкам: ведь интерактивные приложения, VoIP и видео требуют, чтобы задержка не превышала 50–100 мс.

Данную проблему иллюстрируют данные, полученные с помощью приложения Netalyzr[38] для более чем 130 000 тестов, проведенных пользователями по всему миру. Каждой точке на графике соответствует отдельный тест. В каждом тесте скорость передачи постоянно увеличивалась, пока все буферы не оказывались заполненными. Из анализа распределения полученных пакетов исследователи делали вывод о пропускной способности канала (узкого горла) и возможном размере буфера[39]. Горизонтальные группы соответствуют наиболее типичным скоростям доступа, а вертикальные группы указывают на типичные размеры буферов. Наконец, диагональные линии показывают дополнительную задержку, порожденную буферизацией.

Как управлять очередями

В ряде случаев архитектура DiffServ может помочь решению проблемы. Поскольку DiffServ позволяет изолировать друг от друга потоки различных классов, то, например, для веб- и голосового трафика могут использоваться различные буферы. Но в то же время каждый из этих буферов подвержен проблеме bufferbloat.

Более основательное решение — использование алгоритмов активного управления очередями, или AQM (Active Queue Management). Первый из таких алгоритмов, RED[40], был разработан еще в 1993 году, когда проблема переполненных буферов впервые встала на повестку дня. Суть алгоритмов семейства RED заключается в постоянном мониторинге динамики уровня заполненности буфера и своевременной сигнализации различным TCP-потокам о необходимости снижения скорости путем маркировки или просто отброса пакета. К сожалению, уровень внедрения этих технологий остался невысок ввиду сложности настройки и негативного результата при неправильно выбранных параметрах (а во многих случаях и ввиду невозможности выбора оптимальных параметров для всех ситуаций).

---

38 http://netalyzr.icsi.berkeley.edu

39 Более подробно см. «Netalyzr: Illuminating The Edge Network», http://icir.org/ christian/publications/2010-imc-netalyzr.pdf

40 Random Early Detection, случайное раннее обнаружение, http://ru.wikipedia.org/ wiki/Random_early_detection

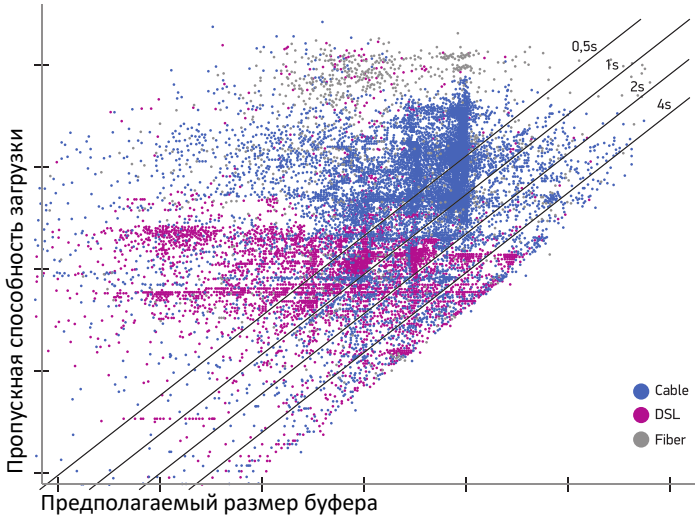

Рис. 60. Данные измерений пропускной полосы и возможных размеров буферов, полученные из тестов Netalyzr; вертикальные группы указывают на типичные размеры буферов в сети.
Источник: блог Jim Gettys (bufferbloat)
https://gettys.wordpress.com/2012/02/20/diagnosing-bufferbloat/

Контроль задержки: CoDel

Основываясь на опыте RED, Кати Николс (Kathie Nichols) и Ван Якобсон (Van Jacobson) предложили новый алгоритм. В отличие от предшественников, он не требовал ручной настройки. Алгоритм был назван CoDel от Controlled Delay, или контролируемая задержка. Ведь единственным целевым параметром является не размер очереди или ее статистические данные и не утилизация канала или степень отброса пакетов, а минимальная задержка, которой подвергаются пакеты в буфере.

Цель алгоритма CoDel — поддержание минимальной задержки ниже установленного параметра (5 мс). Если за определенный интервал, который изначально равен 100 мс, минимальная задержка превышает 5 мс, то последний пакет, покинувший очередь, отбрасывается, а интервал уменьшается в соответствии с формулой, отражающей число последовательных отбросов пакетов[41]. Как только минимальная задержка опять достигает заданного значения, отбрасывание пакетов прекращается и интервал принимает первоначальное значение в 100 мс.

Тестирование алгоритма показало хорошие результаты. ColDel позволяет удерживать задержку в районе заданного параметра для широкого спектра скоростей — от 3 до 100 Мбит/с. При этом утилизация канала составляет почти 100%.

[41] Более подробно см. Controlling Queue Delay, http://dl.acm.org/citation.cfm?id=2209336

Внедрение CoDel только начинается. В настоящее время этот алгоритм доступен для ОС Linux и OpenWrt[42]. К сожалению, в реальной жизни скорость внедрения соответствует циклу обновления пользовательского оконечного оборудования, который может составлять четыре–пять лет, а иногда и больше. Многое из сегодняшнего оконечного оборудования серьезно устарело и не позволяет внедрить новую функциональность путем простого апгрейда программного обеспечения. Тем не менее, хотя рассчитывать на быстрое решение проблемы не приходится, операторы должны иметь эту ситуацию в виду и не упустить очередного шанса улучшить качество предоставляемых услуг.

Управление потоками

Сразу оговоримся, что перегрузки в сети и связанные с ними заторы — неизбежное следствие вариации канальных емкостей по пути следования трафика и принципа работы протокола TCP, который старается максимально «заполнить» канал. Поэтому даже увеличение канальных емкостей не решает проблемы полностью. И если механизмы AQM призваны минимизировать излишнюю буферизацию в сети, тем самым уменьшая задержки и улучшая обратную связь, необходимую для контроля потоками, то новые разработки позволяют улучшить само управление потоками.

И здесь, как ни странно, может существенно помочь архитектура DiffServ, о которой мы уже говорили. Однако для этого требуется дополнительный инструментарий и другой взгляд на классификацию трафика.

В изначальной концепции DiffServ предполагалось, что трафик классифицируется в зависимости от приложения, которое его генерирует. Как мы обсуждали, небольшой набор стандартных профилей вряд ли обеспечит долгосрочное решение проблемы качества. Однако если мы посмотрим на совместное использование ресурсов сети не с точки зрения «бюджета» пропускной способности, а с точки зрения «бюджета» заторов, которые пользователь или приложение вызвали в сети, то ситуация станет намного проще.

Такой подход был применен крупным оператором широкополосного доступа в США — Comcast. Собственно, новая система была внедрена в ответ на претензии со стороны пользователей и регулятора (Федерального агентства по связи США) в отношении старой системы, которая для борьбы с перегрузками в сети осуществляла фильтрацию избыточного P2P-трафика, в частности, BitTorrent.

Новая система, внедренная в 2008 году и подробно описанная в RFC 6057[43], классифицирует трафик пользователя в зависимости от его вклада в перегрузку в сети (когда таковая случается). Работает система следующим образом:

---

42 https://openwrt.org

43 RFC 6057: Comcast's Protocol-Agnostic Congestion Management System, URL: https://www.rfc-editor.org/rfc/rfc6057

- по достижении определенного уровня загрузки сети (точнее — 70–80%) выявляются пользователи, на индивидуальную долю которых приходятся непропорционально высокие объемы трафика. Объем трафика считается высоким, если за последние 15 минут трафик пользователя превысил 70% предлагаемой полосы;
- трафик таких пользователей маркируется как трафик низкого приоритета. В маркетинговых терминах Comcast это означает изменение классификации от PBE (Priority Best Effort) на просто BE (Best Effort). Это не значит, что такой трафик обязательно будет отброшен. Однако, если сеть будет бороться с перегрузкой, применяя тот или иной алгоритм обработки очередей, пакеты BE будут обработаны в последнюю очередь (и если степень затора велика, то, возможно, частично отброшены);
- при последующем измерении маркировка трафика пользователя возвращается к PBE, если уровень его трафика опустился ниже 50%.

Данный подход, вкупе с оптимальной буферизацией и обработкой очередей, о которых мы говорили выше, хорошо зарекомендовал себя на практике. При его применении «штрафуются» не отдельные приложения, а пользователи, перегружающие сеть на протяжении длительного времени. При этом в условиях низкой загрузки сети доступна вся ее пропускная способность.

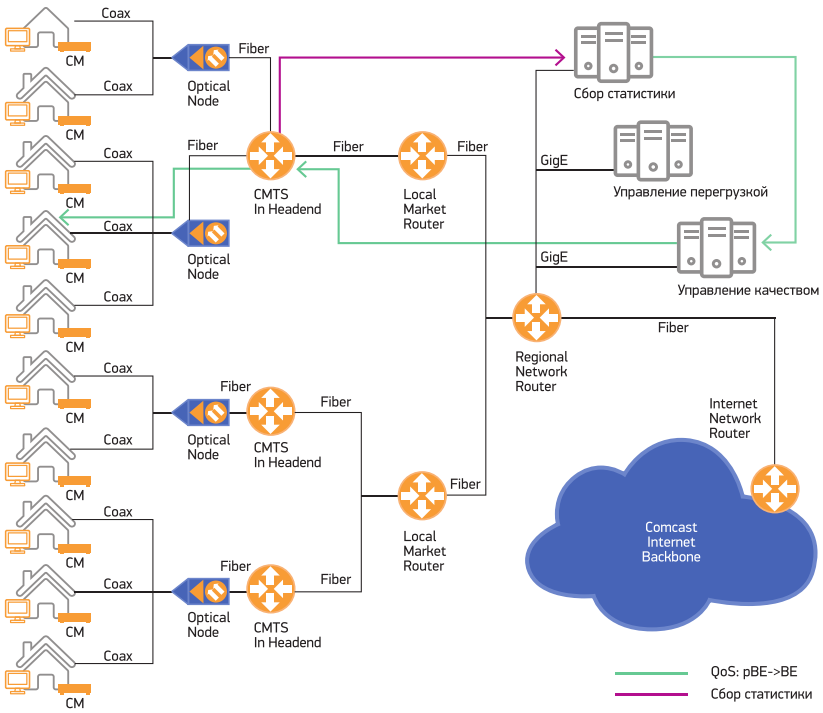

Рис. 61. Упрощенная схема сети широкополосного доступа компании Comcast и системы управления заторами.
Источник: http://downloads.comcast.net/docs/Attachment_B_Future_Practices.pdf

ECN и Conex

RED и CoDel позволяют оптимизировать буферизацию пакетов и тем самым увеличить эффективную пропускную способность пути передачи данных. Однако естественным способом управления потоком все же является отбрасывание пакета, поскольку это сигнал отправителю данных о перегрузке и о необходимости снизить скорость передачи. С другой стороны, отбрасывание пакета — это потеря данных, требующая повторной передачи и, соответственно, приводящая к снижению эффективной пропускной способности.

Поэтому и встает вопрос: нельзя ли сигнализировать о перегрузке каким-либо другим способом, без потери данных?

Для этого в IETF был стандартизован механизм под названием Explicit Congestion Notification[44] (ECN, или «явное уведомление о перегрузке»). ECN основан на маркировке пакетов специальным флагом вместо отбрасывания пакета в режиме перегрузки. Маркировка пакетов происходит на уровне IP путем установки соответствующих битов в IP-заголовке пакета, ведь перегрузка обслуживается маршрутизаторами именно на этом уровне.

Применение ECN не является обязательным. Но отправитель и получатель могут договориться о его использовании в момент установления TCP-соединения. В этом случае все пакеты отправителя для данного соединения будут содержать бит ECT (ECN Capable Transport) в IP-заголовке. В случае перегрузки в сети для пакетов ECN промежуточные маршрутизаторы вместо отбрасывания пакета могут установить флаг CE (Congestion Encountered).

Получатель, в свою очередь, должен отправить уведомление о перегрузке обратно отправителю, но в данном случае — в TCP-заголовке сегмента. При получении такого уведомления отправитель снизит скорость передачи точно так же, как он сделал бы в случае обнаружения потери пакета.

Участники рабочей группы IETF Conex[45] пошли еще дальше, поставив перед собой задачу сделать информацию о сквозной перегрузке всего пути видимой не только отправителю и получателю, но и сетям, через которые этот трафик проходит.

Согласно Conex, отправитель данных указывает уровень перегрузки для данного потока по всему пути, основываясь на информации от получателя. А получатель определяет уровень перегрузки на основе полученных пакетов с маркировкой ECN (CE), а также потерянных пакетов. Хотя эта информация относится к недавнему прошлому (поскольку поступает с задержкой в RTT), это все же лучше, чем ничего.

---

44 RFC 3168: The Addition of Explicit Congestion Notification (ECN) to IP, URL: https://www.rfc-editor.org/rfc/rfc3168

45 Congestion Exposure, http://datatracker.ietf.org/wg/conex/charter

Теперь любой маршрутизатор по пути следования трафика может определить не только уровень перегрузки на собственном участке, но и на других отрезках пути — как вверх, так и вниз по потоку. Для определения перегрузки вверх по потоку маршрутизатору достаточно взглянуть на уровень пакетов с маркировкой CE/ECN, а если вычесть это значение из предполагаемой перегрузки для всего пути (как указано отправителем), то получится уровень перегрузки по потоку вниз.

Эта информация имеет различное применение. Например, на ней может быть основана система управления потоками Comcast, использующая более точные данные о фактической перегрузке, вызванной тем или иным потоком. И тогда кондиционированию будет подвергаться не весь трафик пользователя, а отдельные потоки, вносящие существенный вклад в перегрузку. Другой пример применения — получение оператором детальной информации о текущей загрузке различных участков сети.

Топологическая эволюция

Многие из описанных технологий и подходов позволяют оптимизировать загрузку сети, порой высвобождая значительные ресурсы. Сквозная оптимизация, однако, гораздо более сложная задача, поскольку требует дополнительной кооперации между независимыми сетями и усложняет сетевое взаимодействие большим числом параметров. Поэтому сквозные решения имеют ограниченное применение.

Реальному прорыву в обеспечении качества передачи и сквозной производительности мы должны в значительной степени быть благодарными топологической эволюции Интернета, а именно появлению и развитию сетей доставки контента (content delivery networks, CDNs). Но об этом мы поговорим в главе 5 «Будущее начинается сегодня».

Эволюция системы маршрутизации: программируемый Интернет

Интернет все глубже проникает в нашу жизнь и продолжает удивлять новыми возможностями. Но в основном развитие сегодняшнего Интернета происходит на уровне приложений. Каждый день приносит нам сотни, если не тысячи новых apps. Спускаясь ниже, к сетевому уровню, мы обнаружим, что здесь развитие происходит существенно медленнее. Безусловно, «количественно» базовая инфраструктура Интернета продолжает стремительно развиваться — растет общая пропускная способность, внедряются новые технологии канального уровня. Но архитектурно изменения до последнего времени были весьма незначительны. В частности, потому, что сетевая архитектура представляет собой достаточно монолитный блок, включающий множество функций, в том числе и сетевые «приложения», как, например, DNS или BGP. Внедрение новой функ-

циональности требует модернизации всего сетевого стека в миллионе устройств. Представьте, что вам пришлось бы делать апгрейд операционной системы компьютера каждый раз, когда вы устанавливаете новое приложение!

Другими словами, инновация глобальных инфраструктурных протоколов и приложений в рамках сегодняшней архитектуры весьма затруднительна. Новые функции и возможности увеличивают сложность системы, их тестирование трудоемко, а внедрение сложно реализуемо, поскольку требует глобального масштаба. Мы обсудим эти вопросы подробнее в главе 4 «Экосистема Интернета». Не обошли эти проблемы и систему маршрутизации, и используемые протоколы, такие как BGP и OSPF. В рамках существующей парадигмы, когда каждый узел принимает независимое решение на основе информации, полученной от соседей, очень трудно решать глобальные задачи оптимизации трафика и реализации соответствующей политики маршрутизации.

Поэтому многие связывают большие надежды с новой сетевой моделью, получившей название SDN (Software Defined Networking), или программно-конфигурируемая сеть. В чем же новизна подхода, и действительно ли это новое архитектурное знамение?

Пакеты, коммутация и маршрутизация

Чтобы лучше представить архитектурный сдвиг, предлагаемый SDN, вернемся к общим вопросам маршрутизации в IP-сетях.

Различают внутреннюю (по отношению к сети) и внешнюю, или межсетевую, маршрутизацию. Эти две системы используют различные протоколы. Например, для внутренней маршрутизации широко используется протокол OSPF[46], а стандартом межсетевой маршрутизации, как мы говорили, является BGP. Ключевым элементом сетевой инфраструктуры является маршрутизатор, который выполняет две функции: собственно маршрутизацию — определение лучшего маршрута к получателю данных — и пересылку пакетов с одного интерфейса на другой. Иногда под маршрутизацией понимают обе функции, но на деле они существенно различаются.

В алгоритмическом плане пересылка пакетов достаточно проста, и основной задачей является производительность. Отправителями и получателями пакетов являются интерфейсы маршрутизатора, а определение адресата осуществляется с помощью так называемой таблицы передачи — Forwarding Information Base (FIB).

Задачей же маршрутизации является построение собственной таблицы — таблицы маршрутизации (Routing Information Base, RIB), которая потом транслируется в таблицу FIB. Для построения этой таблицы и используются протоколы маршрутизации, которые на основе информации, полученной от соседних

[46] http://ru.wikipedia.org/wiki/OSPF

маршрутизаторов (например, о связности и доступности тех или иных маршрутов), и собственной конфигурации (например, статических маршрутов и ограничений, наложенных сетевой политикой маршрутизации) формируют собственное представление о сети и ее топологии.

Важным в этой модели является то, что каждый маршрутизатор принимает решения самостоятельно и относительно независимо.

Такая модель работает замечательно в простых сетях, особенно когда основной задачей является обеспечение связности. Она проста и надежна. Однако по мере усложнения сетевой архитектуры и политики внутренней и внешней маршрутизации ограничения модели становятся все более заметными.

Возьмем, к примеру, необходимость включения в политику маршрутизации требований обеспечения определенных параметров качества, производительности и стоимости. Эта задача зачастую невыполнима из-за ограничений существующих протоколов маршрутизации. А попытки удовлетворить такие требования в рамках протоколов уже привели к их значительному усложнению и созданию закрытых расширений.

Аналогичные проблемы присутствуют и в сетях Ethernet или MPLS, когда автономность элементов и связанная с этим необходимость распределенной конфигурации существенно усложняют управление сетью.

Программно-конфигурируемая сеть

Концепция SDN позволяет разрабатывать и конфигурировать сети при помощи программирования.

В традиционной архитектуре в одном устройстве сосуществуют уровень управления (так называемый control plane, к которому относятся, например, процессы маршрутизации) и уровень передачи данных (так называемый data plane, отвечающий за пересылку пакетов с одного интерфейса на другой). Концепция SDN предусматривает передачу управляющих функций центральному серверу — так называемому контроллеру, таким образом заменяя традиционную распределенную модель маршрутизации на централизованную. Соответственно, и процесс управления сетью, включающий создание маршрутов, является не чем иным, как программированием сети в целом. Для сравнения на рис. 6о приведены традиционная сетевая архитектура и сеть SDN.

Такой подход обладает рядом существенных преимуществ.

Во-первых, существенно упрощается сам процесс создания маршрутов. В сегодняшней сети маршрутизация — это распределенный итеративный процесс, при котором рабочая топология сети «вычисляется» совместно всеми устройствами. А в SDN это не что иное, как программа моделирования сети с заданными параметрами. Использование этой модели открывает новые воз-

можности по созданию сети с требованиями, немыслимыми в рамках традиционного инжиниринга трафика и с использованием стандартных протоколов маршрутизации.

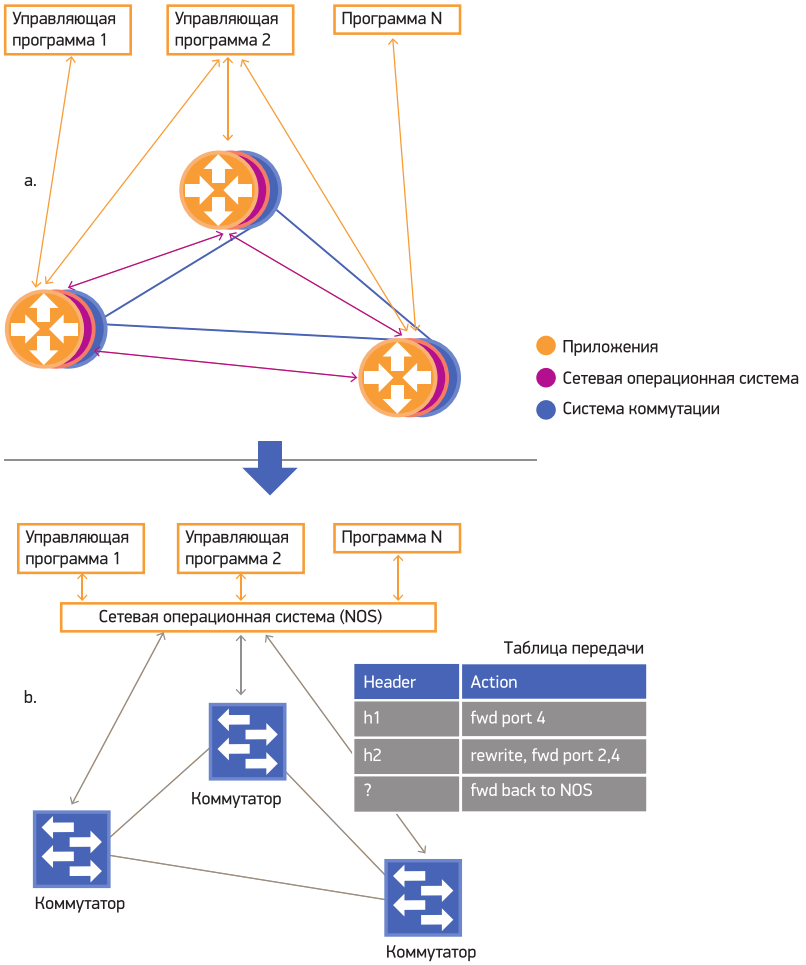

Рис. 62. Разделение системы управления и передачи в архитектуре SDN:
a — традиционная архитектура с автономными сетевыми элементами;
b — архитектура SDN с централизованной управляющей функцией).

Рассмотрим в качестве примера реконфигурацию сети в случае падения канала между какими-либо узлами сети. В традиционной модели маршрутизатор, подключенный к данному каналу, сообщит своим соседям о разрыве связи — и все они независимо займутся разработкой новых маршрутов. При этом они будут распространять информацию об изменяющейся топологии своим соседям и т.д.

В конце концов итерационный процесс закончится и сеть перейдет в новое состояние. В зависимости от сложности сети и используемых протоколов маршрутизации, этот процесс займет больше или меньше времени, но, учитывая задержки при передаче информации на каждой итерации, всё произойдет совсем не мгновенно. Более того, этот процесс не является детерминированным: другими словами, если повторится падение того же канала — не факт, что сеть перейдет в то же состояние.

В случае использования управляющего центра расчет новой топологии производится исходя из знания обо всей сети в целом. Мы также можем задать необходимую топологию следующего состояния. Наконец, поскольку создание новой топологии — это чисто вычислительный процесс, он может быть выполнен значительно быстрее. Во-вторых, значительно увеличиваются возможности для инноваций. В традиционной распределенной модели необходимо привести функциональность к общему знаменателю — для возможности взаимодействия между независимыми устройствами. Это определяет существенную консервативность системы по отношению к новшествам и приводит к «технологической окостенелости», что мы во многом наблюдаем в сегодняшней глобальной инфраструктуре Интернета. В SDN же инновация — лишь вопрос написания нового приложения. Наконец, в-третьих, вместо сложных и дорогостоящих маршрутизаторов можно использовать более простые устройства.

OpenFlow

SDN и OpenFlow — не одно и то же. SDN означает более общую архитектурную концепцию отделения уровня передачи данных от уровня управления. OpenFlow — протокол, наиболее проработанный и стандартизованный, который реализует эту концепцию. Далее мы также рассмотрим несколько другой подход к реализации SDN, основанный на программировании сети с использованием существующих протоколов маршрутизации.

Разработка OpenFlow началась с исследовательского проекта, который должен был дать возможность разработчикам новых сетевых архитектур и протоколов опробовать свои идеи в более или менее реальной среде — скажем, в рамках университетской сети, причем таким образом, чтобы внедрять новую архитектуру изолированно от существующих услуг и параметров сети, исключив негативное влияние на ее текущую функциональность.

Идея OpenFlow проста и основана на наблюдении, что, несмотря на существенные различия между сотнями моделей коммутаторов и маршрутизаторов, все они содержат таблицу передачи. А она определяет базовую функцию передачи данных — как можно быстрее переправить каждый входящий пакет на определенный исходящий интерфейс. Более того, хотя формат этих таблиц различен, можно идентифицировать стандартный набор функций данного уровня.

Каждая запись абстрактной таблицы передачи OpenFlow является «правилом» и связана с так называемым потоком данных (flow). Поток определяется заголовком

пакета — например, комбинацией адресов MAC, IP и номеров портов источника и получателя данных, хотя в принципе поток может состоять из пакетов с нестандартным заголовком — например, для поддержки внедрения новых протоколов. Не все элементы этой комбинации должны быть определены — например, поток может быть определен как весь трафик к некоторому хосту. В этом случае определенным является только один элемент — IP-адрес получателя данных.

Другим элементом записи таблицы является «действие» (action), которое определяет требуемую обработку пакетов потока. Основных действий четыре:

1. Передать пакет на определенный порт (или определенные порты) коммутатора.
2. Передать пакет контроллеру через «защищенный» канал. Контроллер — это управляющий центр сети, включающий центральную сетевую операционную систему и управляющие приложения, рассчитывающий топологию и маршруты, а также осуществляющий другие функции управления. Поэтому, как правило, первый пакет неизвестного потока отправляется контроллеру для определения правила и создания новой записи таблицы передачи.
3. Отбросить пакет. Это действие может быть необходимым, например, в борьбе с компьютерными атаками.
4. Пакет может быть направлен на «стандартную» обработку — например, если OpenFlow-коммутатор также является стандартным коммутатором или маршрутизатором. Данная функциональность позволяет отделить друг от друга потоки данных, управляемые OpenFlow, и потоки, управляемые другими механизмами, например, с помощью существующих протоколов маршрутизации. Благодаря этому исследователи могут изолировать экспериментальный трафик от нормального, используя общую инфраструктуру.

Наконец, последний элемент записи таблицы Open Flow содержит различную статистику — продолжительность потока, число полученных и переданных пакетов и т.п.

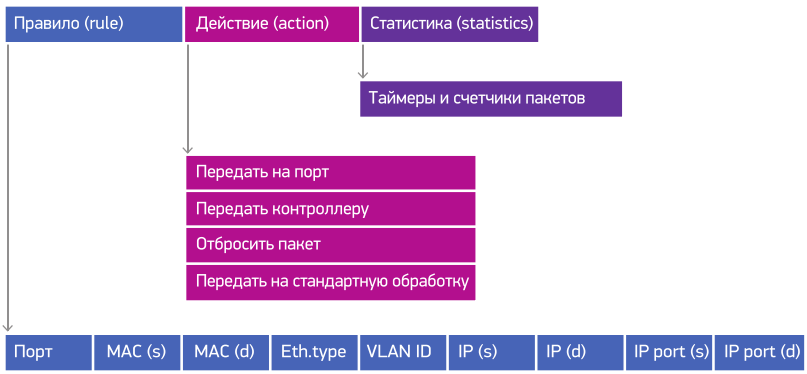

Рис. 63. Упрощенная структура таблицы передачи, OpenFlow 1.0.

На деле сегодняшняя архитектура OpenFlow[47] немного сложнее. В частности, она предусматривает наличие нескольких таблиц правил с возможностью каскадирования (см. рис. 64).

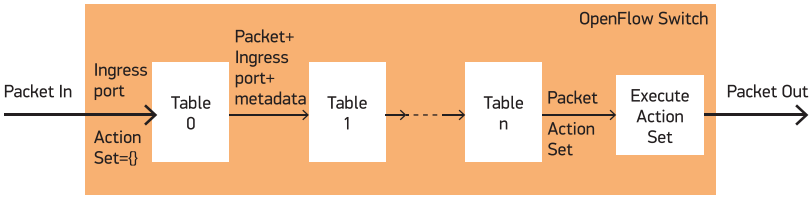

Рис. 64. «Каскадирование» правил и действий позволяет сделать обработку более гибкой.
Источник: спецификация коммутатора OpenFlow – OpenFlow Switch (Specification 1.5.1., https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf)

Такая модель открывает уникальные возможности. Коммутатор OpenFlow может быть чем угодно — от простого коммутатора до сетевого экрана и маршрутизатора. В конечном итоге все определяется таблицей передачи, пример которой представлен на рис. 65.

Контроллер обменивается информацией с «подконтрольными» ему коммутаторами OpenFlow с помощью одноименного протокола. В рамках стандартной модели SDN задача коммутаторов — без задержек передавать пакеты с одного порта на другой, осуществляя некоторую обработку в соответствии с правилами. В ответ на запрос коммутатор может сообщить контроллеру о своих возможностях и конфигурации, а также сигнализировать об изменениях в своем состоянии, например, о потере канала или возникновении ошибки. Но в остальном коммутатор полностью полагается на контроллер. Коммутатор не знает ни о топологии сети, ни даже о своих непосредственных соседях.

	Порт комм.	MAC src	MAC dst	Eth.type	VLAN ID	IP src	IP dst	Proto/ sport	Proto/ dport	Действие
Коммутатор	*	*	00:1F:...	*	*	*	*	*	*	Port6
Сетевой экран	*	*	*	*	*	*	*	*	22	Отбросить
Маршрутизатор	*	*	*	*	*	*	198.51.100.1	*	*	Port6
Коммутация потоков	Port3	00:20:...	00:1F:...	0800	Vlan1	192.0.2.10	198.51.100.1	6/27089	6/80	Port6

Рис. 65. Примеры реализации различных специализированных функций коммутатора с помощью правил таблицы передачи (* обозначает любое допустимое значение соответствующего поля).

[47] см. https://www.opennetworking.org/sdn-resources/onf-specifications

Задача определения «общей картины» и конфигурирование коммутаторов возлагается на контроллер, а точнее, на приложения, его использующие. Это они задают топологию сети и рассчитывают оптимальные маршруты. С помощью контроллера они устанавливают нужные правила, следят за состоянием устройств, осуществляют мониторинг трафика и сбор статистики.

Другими словами, OpenFlow дает базовые функции управления любым аппаратным обеспечением — коммутаторами. А вот с программной начинкой SDN ситуация более фрагментарна. Хотя многие ведущие производители сетевого оборудования, и в первую очередь — производители коммутаторов, заявляют о своей поддержке OpenFlow, программный интерфейс от контроллера к приложениям либо вообще недоступен, либо является собственной разработкой производителя.

Это, безусловно, не позволяет архитектуре SDN полностью раскрыть свой инновационный потенциал, поскольку делает невозможной разработку «сетевых мозгов», независимых от производителя контроллера. На рынке коммутаторов присутствие OpenFLow наиболее заметно. Стратегия ведущих производителей, в том числе Brocade, BigSwitch, IBM, HP и NEC, включает развитие SDN на основе OpenFlow. Правда, OpenFlow пока что реально поддерживают лишь отдельные модели.

Наиболее впечатляющий пример использования OpenFlow на практике — распределенная внутренняя сеть Google, обеспечивающая обмен данными между датацентрами (так называемая сеть G-scale), показанная на рис. 66. Для построения этой сети компания была вынуждена разработать собственные коммутаторы, но обмен данными между ними и контроллерами основан на OpenFlow.

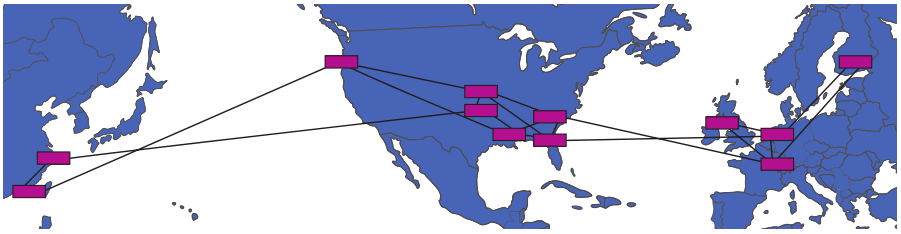

Рис. 66. Распределенная сеть Google (G-scale), использующая протокол OpenFlow.
Источник: доклад Urs Hoelzle, Google на конференции Open Networking Summit, апрель 2012 г. (https://www.segment-routing.net/images/hoelzle-tue-openflow.pdf)

Программирование системы маршрутизации — I2RS

Внедрению технологии SDN также препятствует отсутствие стандартного подхода и программного интерфейса верхних уровней. Задача становится еще сложнее, если мы говорим о миграции традиционной сети в SDN. Особенно если эта сеть обеспечивает услугу третьего уровня — маршрутизацию пакетов.

Подобно многим новым архитектурным решениям, концепция SDN (и особенно в исполнении OpenFlow) вызывает некоторое неприятие и у сетевых инженеров, и у администраторов, и у производителей сетевого оборудования, особенно маршрутизаторов.

Для первых SDN означает необходимость переключения на новый стиль управления сетью, если не переквалификацию в программиста. При этом неочевидно, что накопленный опыт разработки, управления и отладки сложных сетей может быть эффективно перенесен на новую платформу. И хотя не все задачи могут быть решены через систему управления сетью, или NMS[48], SDN представляется лишь как одна из подобных метасистем.

Производители маршрутизационного оборудования видят в SDN/OpenFlow угрозу собственной конкурентоспособности. Найдется хотя бы несколько компаний, оборудование которых переключает пакеты быстрее, чем, скажем, маршрутизаторы Juniper. Наверное, немало компаний смогут разработать программное обеспечение лучше, чем, например, Cisco. Эффективное объединение этих компонентов в одном продукте и громадная внедренная база — явное преимущество, которое может быть утрачено с приходом SDN, когда потребитель получит возможность сам выбирать и комбинировать лучшее сетевое программное обеспечение с лучшим аппаратным решением. Кстати, именно поэтому и Cisco, и Juniper предлагают SDN в собственной закрытой упаковке, означающей, по существу, улучшенный (по сравнению с пользовательским интерфейсом командной строки!) программный интерфейс для своих маршрутизаторов. Возвращаясь к вопросу миграции к новой технологии и архитектуре, отметим: лучшая программируемость сети — это уже существенный шаг вперед. Особенно если он позволяет решить некоторые насущные проблемы сетевых администраторов.

Попытки решить эту задачу предпринимались давно. Помимо повсеместно присутствующего интерфейса командной строки, сегодня существует ряд механизмов, позволяющих программно взаимодействовать с маршрутизатором.

Наиболее распространенным является протокол SNMP[49], который позволяет получать от устройства информацию о его состоянии и конфигурации, а также различную статистику. Для этого каждое устройство содержит так называемую базу управляющей информации, или MIB[50], где хранятся параметры, доступные по SNMP. Обычно SNMP используется для сбора статистики и мониторинга за состоянием устройств и их конфигурацией. Однако эта система

[48] https://ru.wikipedia.org/wiki/Система_управления_элементами_сети

[49] Simple Network Management Protocol, простой протокол управления сетью, http://ru.wikipedia.org/wiki/SNMP

[50] Management Information Base, http://ru.wikipedia.org/wiki/Management_Information_Base

крайне редко применяется для конфигурации устройства. Отсутствие необходимых функций, таких как возможность «отката», понятия тестовой конфигурации и т.п., не позволяет использовать ее в этом режиме в производственных условиях.

Другой сетевой протокол конфигурации, NetConf[51], также разработанный в IETF, снимает большинство ограничений SNMP, однако его применение ограничено ввиду отсутствия стандартной модели данных.

Система ForCES[52] предлагает принципиально другой метод конфигурирования маршрутизатора. Эта архитектура, разработанная в IETF еще в 2004 году, так же как и в случае OpenFlow, предусматривает разделение монолитного устройства на управляющий и передающий элементы[53]. Однако программный интерфейс относится к уровню передачи, тем самым не позволяя использовать «интеллектуальную начинку» маршрутизатора — логику протоколов маршрутизации и построения маршрутизационной таблицы.

Чтобы решить задачу обеспечения программного интерфейса к системе маршрутизации устройств, в IETF была создана Рабочая группа I2RS[54].

Подход I2RS направлен на использование существующей информации, собранной самим маршрутизатором о топологии сети, ее состоянии, а также о собственном состоянии и конфигурации. Ограничиваясь интерфейсами уровня системы маршрутизации, I2RS также использует существующие средства преобразования маршрутизационной информации в таблицы передачи FIB и RIB. Общая архитектура I2RS приведена на рис. 67.

Для иллюстрации возможностей I2RS рассмотрим один из примеров возможного применения — установку статического маршрута. Это часто встречающаяся задача конфигурации сети, особенно при инжиниринге трафика. Традиционные способы включают использование интерфейса командной строки и MIB, но они оба не поддерживают программного интерфейса. Обеспечивая доступ к RIB и FIB, I2RS позволяет решить эту задачу на программном уровне.

Другим примером является возможность программного доступа к функциям «policy-based routing»[55], а именно к правилам обработки определенного трафика.

---

51 http://ru.wikipedia.org/wiki/NETCONF

52 RFC 7409: Forwarding and Control Element Separation (ForCES) Packet Parallelization, URL: https://www.rfc-editor.org/rfc/rfc5155

53 Более подробно о различиях между OpenFlow и ForCES см. https://datatracker.ietf.org/doc/draft-wang-forces-compare-openflow-forces

54 Interface to the Routing System, http://datatracker.ietf.org/wg/i2rs/

55 PBR, http://en.wikipedia.org/wiki/Policy-based_routing

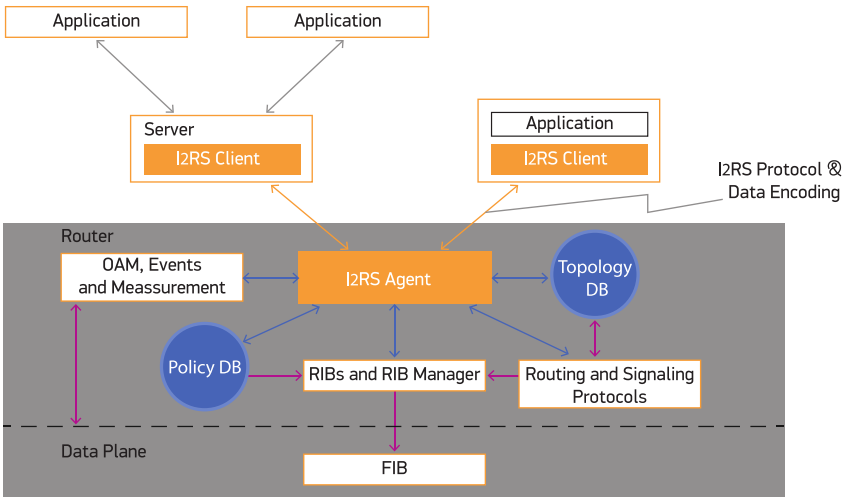

Рис. 67. Архитектура системы I2RS; желтым цветом обозначены элементы и протоколы I2RS (с разрешения A. Farrel, Old Dog Consulting).

При этом речь идет не только о доступе, например, к записям установки маршрута или изменениям правила. I2RS дает возможность получать информацию от системы маршрутизации — в частности, данные RIB или текущую конфигурацию и состояние отдельных элементов. Это не менее важно, особенно в контексте моделирования топологии сети.

Будущее SDN

На сегодняшний день SDN — это архитектурная концепция и маркетинговый термин, объединяющий множество закрытых решений построения и управления сетью. Как архитектура, SDN привлекает своей концептуальной простотой, открывая громадный потенциал для построения более сложных систем на ее основе. Кроме того, декомпозиция управляющих и передающих элементов сети открывает новые горизонты для инноваций. Стремительному развитию этой технологии может способствовать стандартизация всех интерфейсов SDN и, как следствие, создание открытой архитектуры, когда потребитель сможет самостоятельно комбинировать продукты различных производителей: коммутаторы, контроллеры и программное обеспечение управления сетью.

Сегодня стандарты — протоколы и модели данных — существуют только для «южного» интерфейса: между контроллером и коммутаторами. Стандартизация «северного» интерфейса обеспечит доступ на рынок независимым разработчикам программного обеспечения. Определение стандартного интерфейса между контроллерами позволит различным сетям взаимодействовать между собой, открывая возможности построения SDN, охватывающих несколько независимых провайдеров.

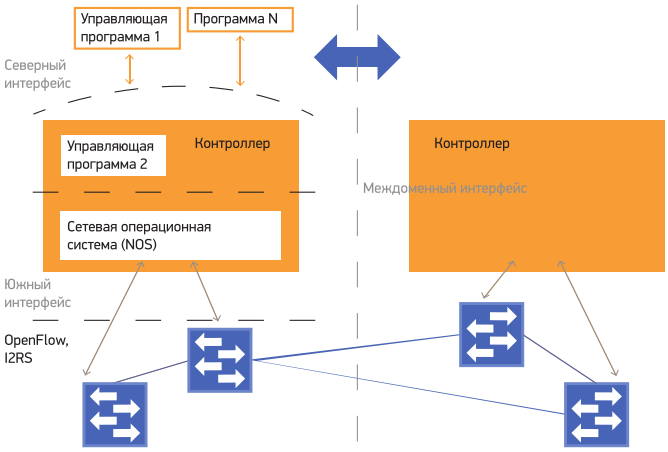

Рис. 68. Интерфейсы SDN, требующие стандартизации.

Достаточно посмотреть на число организаций, участвующих в исследованиях и разработке стандартов, связанных с SDN (рис. 69), чтобы понять — работы здесь непочатый край. Этот рисунок также свидетельство тому, насколько сильна потребность в стандартных компонентах, строительных блоках SDN, способных взаимодействовать между собой. Работы ведутся во многих направлениях, и, возможно, недалек тот день, когда SDN из архитектурной концепции и маркетингового термина превратится в стандартные протоколы и технологии.

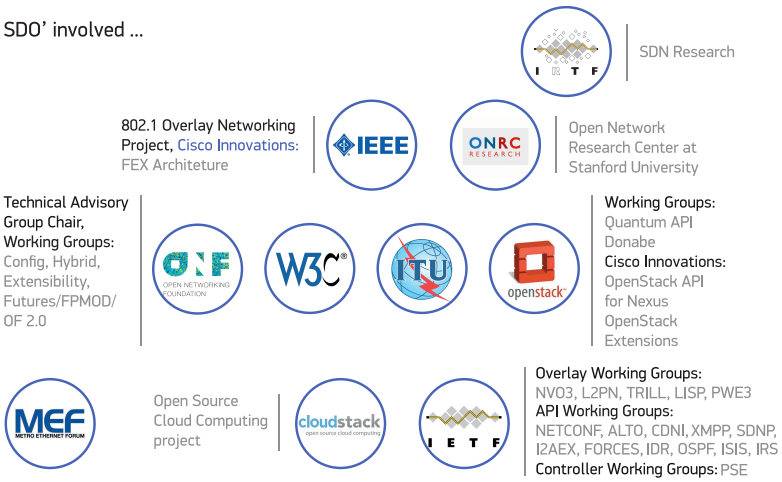

Рис. 69. Организации, участвующие в исследованиях и разработке стандартов, связанных с SDN.
Источник: доклад David Ward, Cisco, на пленарном заседании IETF84, июль 2012 г. (http://www.ietf.org/proceedings/84/slides/slides-84-iab-techplenary-3.pdf)

И кто знает — может быть, Интернет действительно станет программируемым.

Заключение

Ограниченный объем данных, называемый пакетом, в порядке эстафеты передается от отправителя к следующему узлу-маршрутизатору, затем к следующему — пока наконец не достигнет получателя. Каждый из узлов принимает решение о дальнейшем пути пакета самостоятельно, в зависимости от текущей топологии сети. Это значит, что потоки данных могут быть оперативно перенаправлены, скажем, при выходе из строя одного из узлов или «падении» канала. Передачу данных не нужно синхронизировать между всеми сетями, через которые проходит поток. Такие сети пакетной коммутации могут поддерживать любые технологии передачи данных и любые приложения. Эта новая парадигма в конце 1960-х гг. определила и продолжает определять инфраструктуру передачи данных Интернета. Управляет этой структурой протокол маршрутизации BGP, который позволяет узлам, а точнее, сетям, или автономным системам, обмениваться их представлением о топологии Интернета с другими сетями. Эта информация все время меняется, в среднем в день в Интернете происходит около 50 000 изменений топологии: подключаются новые подсети, выходят из строя каналы и маршрутизаторы, меняется топология сетей. Но эти изменения распространяются достаточно быстро, и в целом система является стабильной. Со времени внедрения протокола BGP прошло уже тридцать лет, и размер Интернета в терминах количества маршрутов вырос на три порядка. Но, как это ни странно, основные протоколы и архитектура маршрутизации в Интернете почти не изменились. Узлы по-прежнему коммутируют пакеты IP, обмен и определение маршрутов происходит с помощью протокола BGP. И удивительно, что, несмотря на столь стремительный рост, устойчивость и производительность системы по-прежнему соответствуют потребностям ее пользователей.

Однако мы уже отчетливо видим и контуры новой архитектуры: это облачные услуги, централизованные, глобально распределенные данные, доступ к которым не отличается от доступа к локальному диску компьютера. Это контент, видео, демонстрирующее высокое качество, неожиданное для сети с отсутствием поддержки качества предоставления услуг. Такая инфраструктура требует новых решений, по крайней мере во внутренней архитектуре сетей, составляющих сегодняшний Интернет.

Глобальные изменения в Интернете нелегки — будь то система адресации, имен или маршрутизации. И все-таки виртуализация Сети открывает новые возможности, в корне меняющие наши представления о топологии, производительности и местоположении ресурсов.