Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

«Аппаратное обеспечение компьютерных сетей»

на тему

«Подсистема маршрутизатора»

БГУИР КР 1-40 02 01 01 019 ПЗ

Выполнил: Руководитель:

Студент гр. 550502 Глецевич И. И.

Кессо П. И.

Минск 2018

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ………………………………………………………………………………...3](#_8702g290vnnk)

[1. ОБЗОР ЛИТЕРАТУРЫ……………………………………………………………….....6](#_kr7ul97aoeqs)

[ЗАКЛЮЧЕНИЕ…………………………………………………………………………...14](#_vyludyuhp4t)

[СПИСОК ЛИТЕРАТУРЫ………………….……………………………………………..15](#_wuu0aqr59vz8)

# 

# ВВЕДЕНИЕ

Сетевой уровень занимается разработкой маршрутов доставки пакетов от отправителя до получателя. Чтобы добраться до пункта назначения, пакету может потребоваться преодолеть несколько транзитных участков между маршрутизаторами. Функции, выполняемые на сетевом уровне, резко контрастируют с деятельностью канального уровня, цель которого была более скромной — просто переместить кадры с одного конца провода на другой. Таким образом, сетевой уровень оказывается самым нижним уровнем, который имеет дело с передачей данных по всему пути от одного конца до другого. Для достижения этих целей сетевой уровень должен обладать информацией о топологии сети (то есть о множестве всех маршрутизаторов и связей) и выбирать нужный путь по этой сети, даже если она достаточно крупная. При выборе маршрутизаторов он должен также заботиться о том, чтобы нагрузка на маршрутизаторы и линии связи была, по возможности, более равномерной. Наконец, если источник и приемник находятся в различных сетях, именно сетевой уровень должен уметь решать проблемы, связанные с различиями в сетях. В данной главе мы рассмотрим все эти аспекты и проиллюстрируем их в основном на примере Интернета и его протокола сетевого уровня — IP.

Сегодня в мире компьютерных сетей используется только одна сетевая модель: TCP/IP (Transmission Control Protocol/Intemet Protocol - протокол управления передачей/протокол Интернета).

Модель TCP/IP описывает множество протоколов, позволяющих взаимодействовать компьютерам. Подробное описание протоколов, входящих в стандартный набор TCP/IP, представлено в документах, которые называются запросами на комментарии (Requests for Comments - RFC). (Вы можете найти документы RFC, используя любой сетевой поисковый механизм.) Модель TCP/IP не дублирует работу, уже проделанную другими организациями по стандартизации или консорциумом производителей, просто ссылаясь на соответствующий стандарт или протокол, созданный этими группами. Например, Институт инженеров по электротехнике и электронике ( Institute of Electrical and Electronic Engineers - IEEE) определяет локальные сети Ethemet; поэтому модель TCP/IP не определяет сети Ethemet в своих запросах на комментарии, а ссылается на документы IEEE Ethernet.

Компьютер, использующий протоколы TCP/IP, можно сравнить с обычным телефоном. Можно пойти в магазин, торгуюший бытовой техникой, и купить телефонный аппарат какой угодно модели и производителя. Тем не менее, если принести его домой и включить в телефонную розетку тем же самым кабелем, каким был подключен старый аппарат, новый телефон будет работать. Производители телефонов знают стандарты телефонии для своей страны и производят телефоны в соответствии с ними.

Аналогично, когда вы покупаете новый компьютер, он, по сути, уже реализует модель TCP/IP, чтобы вы могли взять соответствующие кабели и подключить компьютер к сети. Теперь вы можете использовать веб-браузер для просмотра своего любимого веб-сайта. Почему? Операционная система на компьютере реализует части модели TCP/IP. Встроенная в компьютер плата Ethernet или плата беспроводной сети реализует некоторые стандарты LAN, используемые моделью TCP/IP. Короче говоря, производители, которые создали аппаратные средства и программное обеспечение, реализовали модель TCP/IP.

Чтобы упростить изучение сетевых моделей, каждая из них разделена на несколько функциональных разделов, называемых уровнями (layer). Каждый уровень включает протоколы и стандарты, относящиеся к данному функциональному разделу. Фактически есть две альтернативные модели ТСР /IP, как показано на рис. 1 .1.



Рис. 1.1. Две сетевые модели ТСР//Р

Слева представлена первоначальная модель TCP/IP, описанная в документе RFC1.1.22. Она подразумевает четыре уровня. Два верхних уровня сосредоточены больше на приложениях, которые должны передавать и получить данные. Нижние уровни сосредоточены на передаче битов по каналу связи, где уровень Интернета осуществляет передачу данных по всему пути от отправляющего компьютера на компьютер конечного получателя.

Справа представлена используемая в настоящее время общепринятая модель, дополнительные уровни которой сформированы за счет разделения канального уровня первоначальной модели на два отдельных уровня: канала связи и физического (подобно двум нижним уровням модели OSI). Обратите внимание: модель справа сейчас используется чаще.

# 1. ОБЗОР ЛИТЕРАТУРЫ

Протокол IP отвечает за маршрутизацию данных в форме пакетов IP от хоста отправителя к хосту получателя. Он не участвует в физической передаче данных, -

это задача более низких уровней модели TCP/IP. Протокол IP определяет логические детали передачи данных, а не физические. В частности, сетевой уровень определяет правила следования пакетов по сети TCP/IP, даже когда пакет пересекает множество каналов связи WAN и LAN разных типов.

Для осуществления маршрутизации IP маршрутизаторы и компьютеры конечного пользователя (хосты (host) сети TCP/IP) должны взаимодействовать. В операционной системе хоста выполняется программное обеспечение, реализующее сетевой уровень модели TCP/IP. Хосты используют это программное обеспечение для выбора соседнего маршрутизатора, на который следует отсылать пакеты IP. Этот маршрутизатор в свою очередь выбирает направление дальнейшей передачи пакета IP. Пример логики маршрутизации представлена на рисунке 2.1.

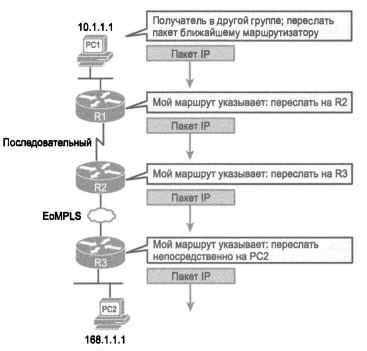


Рис. 2. 1. Логика маршрутизации

Основная функция сетевого уровня заключается в выборе маршрута для пакетов от начальной до конечной точки. В большинстве сетей пакетам приходится проходить через несколько маршрутизаторов. Единственным исключением являются широковещательные сети, но даже в них маршрутизация является важным вопросом, если отправитель и получатель находятся в разных сегментах сети. Алгоритмы выбора маршрутов и используемые ими структуры данных являются значительной областью при проектировании сетевого уровня.

Алгоритм маршрутизации реализуется той частью программного обеспечения сетевого уровня, которая отвечает за выбор выходной линии для отправки пришедшего пакета. Если сеть использует дейтаграммную службу, выбор маршрута для каждого пакета должен производиться заново, так как оптимальный маршрут мог измениться. Если сеть использует виртуальные каналы, маршрут выбирается только при создании нового виртуального канала. После этого все информационные пакеты следуют по установленному маршруту. Последний случай иногда называют сеансовой маршрутизацией (session routing), так как маршрут остается в силе на протяжении всего сеанса связи (например, все время, пока вы подключены к сети VPN).

Полезно понимать разницу между маршрутизацией, при которой системе приходится делать выбор определенного маршрута следования, и пересылкой — действием, происходящим при получении пакета. Можно представить себе маршрутизатор как устройство, в котором функционируют два процесса. Один из них обрабатывает приходящие пакеты и выбирает для них по таблице маршрутизации исходящую линию. Такой процесс называется пересылкой (forwarding). Второй процесс отвечает за заполнение и обновление таблиц маршрутизации. Именно здесь в игру вступает алгоритм маршрутизации. Вне зависимости от того, отдельно ли выбираются маршруты для каждого отправляемого пакета или же только один раз для соединения, желательно, чтобы алгоритм выбора маршрута обладал определенными свойствами — корректностью, простотой, надежностью, устойчивостью, справедливостью и эффективностью. Корректность и простота вряд ли требуют комментариев, а вот потребность в надежности не столь очевидна с первого взгляда. Во время работы большой сети постоянно происходят какие-то отказы аппаратуры и изменения топологии. Алгоритм маршрутизации должен уметь справляться с изменениями топологии и трафика без необходимости прекращения всех задач на всех хостах.

Алгоритм маршрутизации должен также обладать устойчивостью. Существуют алгоритмы выбора маршрута, никогда не сходящиеся к фиксированному набору путей, независимо от того, как долго они работают. Устойчивый алгоритм должен достигать состояния равновесия и оставаться в нем. Но он также должен быстро находить этот набор путей, так как соединение может быть прервано до того, как будет достигнуто равновесие. Такие цели, как справедливость и эффективность, могут показаться очевидными - вряд ли кто-нибудь станет возражать против них, - однако они зачастую оказываются взаимоисключающими. Для примера рассмотрим ситуацию, показанную на рис. 2.2.

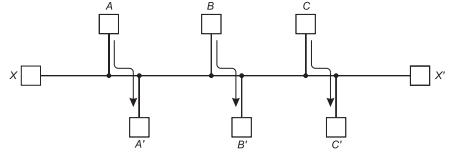


Рис. 2.2. Конфликт справедливости и оптимальности

Предположим, что трафик между станциями A и A', B и B', а также C и C' настолько интенсивный, что горизонтальные линии связи оказываются полностью насыщенными. Чтобы максимизировать общий поток данных, трафик между станциями X и X' следовало бы совсем отключить. Однако станции X и X', скорее всего, имеют другую точку зрения по данному вопросу. Очевидно, необходим компромисс между справедливым выделением трафика всем станциям и оптимальным использованием канала в глобальном смысле. Конфликт справедливости и оптимальности Прежде чем пытаться искать приемлемое соотношение справедливости и эффективности, следует решить, что именно мы будем стремиться оптимизировать. Для увеличения эффективности передачи данных по сети можно попробовать минимизировать среднее время задержки или увеличить общую пропускную способность сети.

Однако эти цели также противоречат друг другу, поскольку работа любой системы с очередями вблизи максимума производительности предполагает долгое стояние в очередях. В качестве компромисса многие сети стараются минимизировать расстояние, которое должен пройти пакет, или просто снизить количество пересылок для каждого пакета. В обоих случаях время прохождения каждого пакета по сети снижается, в результате чего улучшается пропускная способность всей сети. Алгоритмы выбора маршрута можно разбить на два основных класса: неадаптивные и адаптивные. Неадаптивные алгоритмы не учитывают при выборе маршрута топологию и текущее состояние сети и не измеряют трафик на линиях. Вместо этого выбор маршрута для каждой пары станций производится заранее, в автономном режиме, и список маршрутов загружается в маршрутизаторы во время загрузки сети. Такая процедура иногда называется статической маршрутизацией (static routing). Поскольку статическая маршрутизация не реагирует на сбои, она, как правило, используется в тех случаях, когда выбор маршрута очевиден.

Адаптивные алгоритмы, напротив, изменяют решение о выборе маршрутов при изменении топологии и также иногда в зависимости от загруженности линий. Эти динамические алгоритмы маршрутизации (dynamic routing algorithms) отличаются источниками получения информации (такие источники могут быть, например, локальными, если это соседние маршрутизаторы, либо глобальными, если это вообще все маршрутизаторы сети), моментами изменения маршрутов (например, при изменении топологии или через определенные равные интервалы времени при изменении нагрузки) и данными, использующимися для оптимизации (расстояние, количество транзитных участков или ожидаемое время пересылки).

Статическая маршрутизация относится к неадаптивным алгоритмам выбора маршрута. Неадаптивные алгоритмы не учитывают при выборе маршрута топологию и текущее состояние сети и не измеряют трафик на линиях. Вместо этого выбор маршрута для каждой пары станций производится заранее, в автономном режиме, и список маршрутов загружается в маршрутизаторы во время загрузки сети. Поскольку статическая маршрутизация не реагирует на сбои, она, как правило, используется в тех случаях, когда выбор маршрута очевиден.

К преимуществам статической маршрутизации можно отнести: простоту настройки (в небольших сетях), отсутствие дополнительной нагрузки на сеть (в отличии от динамических протоколов маршрутизации).

К недостаткам относится: сложность масштабирования, при возникновении каких-либо изменений в сети, как правило потребуется вмешательство администратора и настройка новых, актуальных статических маршрутов, если возникают проблемы на канальном уровне, но интерфейс по-прежнему в статусе up, то статический маршрут остается активным, хотя фактически данные передаваться не могут.

Адаптивные алгоритмы, напротив, изменяют решение о выборе маршрутов при изменении топологии и также иногда в зависимости от загруженности линий. Динамические алгоритмы маршрутизации отличаются источниками получения информации (такие источники могут быть, например, локальными, если это соседние маршрутизаторы, либо глобальными, если это вообще все маршрутизаторы сети), моментами изменения маршрутов (например, при изменении топологии или через определенные равные интервалы времени при изменении нагрузки) и данными, использующимися для оптимизации (расстояние, количество транзитных участков или ожидаемое время пересылки).

Протоколы динамической маршрутизации могут автоматически отслеживать изменения в топологии сети. Успешное функционирование динамической маршрутизации зависит от выполнения маршрутизатором двух основных функций: поддержка своих таблиц маршрутизации в актуальном состоянии, своевременное распространение информации об известных им сетях и маршрутах среди остальных маршрутизаторов.

Протоколы динамической маршрутизации используются для решения следующих задач: обнаружение удаленных сетей, обновление данных маршрутизации, выбор оптимального пути к сетям назначения, поиск нового оптимального пути в случае, если текущий путь недоступен.

По сравнению со статической маршрутизацией протоколы динамической маршрутизации требуют меньшего вмешательства со стороны администратора. Тем не менее, к издержкам использования протоколов динамической маршрутизации можно отнести тот факт, что часть ресурсов маршрутизатора выделяется для работы протокола.

Основополагающей идеей оптимальных маршрутов является принцип оптимальности (Беллман, 1957). В соответствии с этим принципом, если маршрутизатор J располагается на оптимальном маршруте от маршрутизатора I к маршрутизатору K, то оптимальный маршрут от маршрутизатора J к маршрутизатору K совпадет с частью первого маршрута. Чтобы убедиться в этом, обозначим часть маршрута от маршрутизатора I к маршрутизатору J как r1, а остальную часть маршрута — r2. Если бы существовал более оптимальный маршрут от маршрутизатора J к маршрутизатору K, чем r2, то его можно было объединить с r1, чтобы улучшить маршрут от маршрутизатора I к маршрутизатору K, что противоречит первоначальному утверждению о том, что маршрут r1r2 является оптимальным. Прямым следствием принципа оптимальности является возможность рассмотрения множества оптимальных маршрутов от всех источников к конкретным приемникам в виде дерева, имеющего приемник в качестве корня. Такое дерево называется входным деревом (sink tree). Оно изображено на рис. 2.3, б. Расстояния измеряются количеством транзитных участков. Цель всех алгоритмов выбора маршрутов заключается в вычислении и использовании входных деревьев для всех маршрутизаторов.

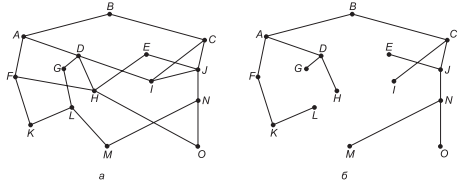


Рис. 2.3. Сеть (а); входное дерево для маршрутизатора B (б)

Если мы будем считать все возможные пути допустимыми, мы получим более общую структуру, и наше дерево будет подходить под определение направленного ациклического графа (directed acyclic graph, DAG). В таких графах нет циклов. Мы будем использовать понятие входного дерева для обозначения обоих вариантов. Оба они также основываются на предположении, что пути не мешают друг другу (из этого следует, в частности, что появление затора на одном пути не приводит к изменению другого пути).

Поскольку входное дерево действительно является деревом, оно не содержит петель, поэтому каждый пакет будет доставлен за конечное и ограниченное число пересылок. На практике все это не так просто. Линии связи и маршрутизаторы могут выходить из строя и снова появляться в сети во время выполнения операции, поэтому у разных маршрутизаторов могут оказаться различные представления о текущей топологии сети. Кроме того, мы обошли вопрос о том, собирает ли маршрутизатор сам информацию для вычисления входного дерева, или эта информация каким-то другим образом поступает к нему. Мы вскоре рассмотрим этот вопрос. Тем не менее принцип оптимальности и входное дерево — это те точки отсчета, относительно которых можно измерять эффективность различных алгоритмов маршрутизации

Сбросом нагрузки называется простое игнорирование маршрутизаторами пакетов, которые они не могут обработать. Своим происхождением этот термин обязан системам электроснабжения, где он означает отключение в случае перегрузок отдельных участков во избежание выхода из строя всей системы. Обычно такое происходит в жаркие летние дни, когда спрос на электроэнергию многократно превосходит предложение.

Ключевой проблемой для маршрутизатора, заваленного пакетами, является выбор пакета, который будет отвергнут. Выбор может зависеть от типа приложений, использующих данную сеть. Для передачи файла более старый пакет ценится выше нового, так как отвержение пакета номер 6 и сохранение пакетов с номерами, например, с 7 по 10 лишь заставит получателя выполнить лишнюю работу: поместить в буфер данные, которые он еще не может использовать. Для мультимедийных приложений, работающих в реальном времени, напротив, новый пакет важнее старого. Причина в том, что пакеты становятся ненужными, если они задерживаются и не приходят вовремя. Первую стратегию (старое лучше нового) часто называют винной стратегией, а вторую (новое лучше старого) — молочной стратегией, так как большинство людей предпочтут пить свежее молоко и выдержанное вино, а не наоборот.

Более разумные алгоритмы сброса нагрузки требуют участия отправителей. В качестве примера можно привести пакеты, содержащие сведения о маршрутизации. Эти пакеты значительно важнее обычных пакетов с данными, поскольку они устанавливают маршруты; если они будут утеряны, может пострадать связность сети. Другой пример — алгоритмы сжатия видеосигнала (например, MPEG), которые периодически посылают полный кадр, а последующие кадры представляют собой карты изменений относительно последнего полного кадра. В таком случае потеря пакета, содержащего разностный сигнал, не так страшна, как потеря полного кадра, так как от этого полного кадра зависят последующие пакеты.

Для реализации интеллектуальной стратегии выбрасывания части информации приложения должны помечать свои пакеты, сообщая сети об их важности. Тогда маршрутизаторы смогут сначала выбросить пакеты наименее важного класса, затем следующего за ним и т.д.

Конечно, при отсутствии стимула все будут помечать свои пакеты не иначе как ОЧЕНЬ ВАЖНО — НИ В КОЕМ СЛУЧАЕ НЕ ВЫБРАСЫВАТЬ. Предотвращение неоправданного использования таких отметок часто достигается за счет сетевых ресурсов и денежных средств. Например, сеть может разрешить отправителям пересылать пакеты с большей скоростью, чем указано в договоре на предоставление услуг, если пакет будет помечаться как низкоприоритетный. Такая стратегия весьма удачна, поскольку более эффективно использует свободные ресурсы, разрешая хостам пользоваться ими, пока это никому не мешает, но не закрепляя за ними этого права.

# 

# ЗАКЛЮЧЕНИЕ

Сетевой уровень предоставляет услуги транспортному уровню. В его основе могут лежать либо виртуальные каналы, либо дейтаграммы. В обоих случаях его основная работа состоит в выборе маршрута пакетов от источника до адресата. В сетях с виртуальными каналами решение о выборе маршрута осуществляется при установке виртуального канала. В дейтаграммных сетях оно принимается для каждого пакета.

В компьютерных сетях применяется большое количество алгоритмов маршрутизации. Заливка — это простой алгоритм передачи пакетов по всем путям. Большинство алгоритмов определяют кратчайший путь и адаптируются к изменениям топологии сети. Основными алгоритмами являются маршрутизация по векторам расстояний и маршрутизация с учетом состояния линий. В большинстве имеющихся сетей применяется один из этих алгоритмов. К другим важным методам маршрутизации относятся использование иерархии в больших сетях, маршрутизация для мобильных хостов, широковещание, многоадресная маршрутизация и произвольная маршрутизация.

Сети подвержены перегрузкам, приводящим к увеличению задержки и утере

пакетов. Разработчики сетей пытаются предотвратить перегрузку разнообразными

способами, включающими создание сетей с достаточной пропускной способностью, выбор не перегруженных маршрутов, временный запрет входящего трафика, отправку сообщения источнику с просьбой замедлить передачу трафика и сброс нагрузки.

# СПИСОК ЛИТЕРАТУРЫ

1. Академия cisco [Электронный ресурс]. – Режим доступа: <https://www.netacad.com/>

2. Глецевич, И. И. Администрирование компьютерных систем и сетей / Глецевич И. И. [Электронный ресурс]. – Режим доступа : https://erud.bsuir.by/index.jsp?PageID=84010&menuItemID=118936&resID=116608&lang=ru&resourceID=116608&versionid=-1&templateID=116641&pagenum=1

3. Одом, У. Официальное руководство Cisco по подготовке к сертификационным экзаменам CCENT/CCNA ICNDl 100-101, акад. изд. : Пер. с англ. - М. : ООО "И.Д. Вильяме", 2015. - 912 с. : ил. - Парал. тит. Англ

4. Одом, У. Официальное руководство Cisco по подготовке к сертификационным экзаменам CCENT/CCNA ICND2 200-101, акад. изд. : Пер. с англ. - М. : ООО "И.Д. Вильяме", 2015. - 736 с. : ил. - Парал. тит. англ