**Лекция за 04.04.2012, часть 1**

***Маршрутизация в сетях TCP IP.***

Изначально, структура адресации в сетях TCP IP была так называемой ***классовой***. Границы, по которым ***адрес*** делился на ***адрес подсети и адрес хоста***, проходили по границам октетов. Это привело к двум очень неприятным последствиям, одно из которых: исчерпание пространства ip-адресов, связанное с тем, что в сетях классов “А” и “В” было очень много не задействованных ip-пулов, соответственно сети класса “С” исчерпывались очень быстро. С этим поборолись следующим образом: ввели, так называемую, ***бесклассовую адресацию***, по которой границы адреса подсети и адреса хоста могли проходить по любому биту. Это потребовало введение нового параметра ***маски подсети***. Которое в бинарном формате представляет собой двоичное число, в котором ***единицы*** стоят везде в полях, соответствующих адресу подсети, а ***нули*** в полях соответствующих адресу хоста. Проведя побитовую конъюнкцию между маской подсети и ip адресом, получим адрес сети, а если провести побитовую конъюнкцию между ip-адресом и инверсией маски подсети, то получим адрес хоста.

***ip-адрес & mask = адрес сети***

***Ip-адрес & inversion (mask) = адрес хоста***

***Представление маски (примеры записей маски):***

* 255.255.255.0 (тетрадный формат)
* 11111111111111111111111100000000 (двоичный формат)
* Ip-address/24 (сокращенный формат)

Была еще одна не менее важная проблема, которая тоже связана с классовой адресацией, при этом резко возрастал объем трафика необходимый на передачу таблиц маршрутизации.

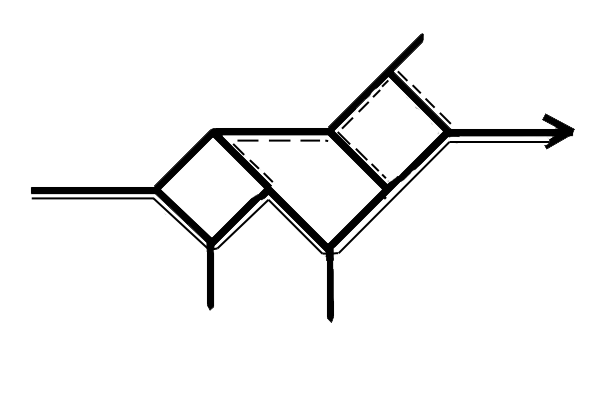
***Маршрутизация*** бывает двух основных типов:

* ***Прямая***
* ***Не прямая*** (***косвенная***)

В случае прямой маршрутизации, отправляя какое-либо сообщение, вы сразу знаете весь список узлов, по которым должно пройти ваше сообщение. В случае не прямой маршрутизации каждый узел, по которому проходит ваше сообщение, сам определяет следующего адресата, которому он должен отправить сообщение.

Если вы используете прямую маршрутизацию, вы не накладываете дополнительной нагрузки на узлы. Вы сразу передаете с вашим сообщением список узлов, через которое должно пройти ваше сообщение, и каждый узел работает как, грубо говоря, “почтовое отделение”, которое переправляет сообщение следующему узлу (по цепочке). С другой стороны, есть и очень существенный недостаток, если во время передачи сообщения происходит сбой, какой-то из узлов является недоступным или не может по каким-то другим причинам обработать сообщение, передача мгновенно прекращается. В другом случае, непрямой маршрутизации, когда каждый узел сам решает, куда передать дальнейшее сообщение, в случае любых ошибок с настройками узлов, возможно зацикливание, при котором сообщение, раз за разом, будет проходить одну и ту же последовательность узлов, ни разу из этого круга не выходя. В TCP-IP применяется непрямая маршрутизация. На самом деле существует еще один тип маршрутизации, который называется ***комбинированный***, или так называемый ***source-routing***. ***Комбинированный*** – это тоже форма непрямой маршрутизации, при которой выбор следующего узла определяется не только политикой и логикой работы каждого узла маршрутизации, но и некоторыми дополнительными параметрами. Которые могут быть описаны в том сообщении, которое вы собираетесь маршрутизировать. В частности для IP, таким параметром может быть исходящий ip-адрес, а не только ip-адрес конечного получателя.

Непрямая маршрутизация описывается через так называемые таблицы маршрутизации, каждой узел которой может отправлять и принимать сообщения. Такие узлы называются ***маршрутизаторами***, или ***роутерами***. Имеется специальная таблица, которая связывает ip-адрес получателя (точнее адрес подсети получателя) с адресом следующего в цепочки маршрутизатора, который должен обработать это сообщение.

Представление графа: рисунок

------- - пути в графе по выбранной метрике.

*Комментарии*: Есть чрезвычайно разветвленный граф, который связывает отправителя сообщения и получателя, и на каждом из узлов этого графа, вы должны решить, куда перенаправить ваше сообщение.

***Таблица маршрутизации*** выглядит следующем образом:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| network destination | | next router | metric | [interface] |
| адрес подсети | маска |  |  |  |
| 192.168.100.0 | 255.255.255.0 | 192.168.14.0 | 10 |  |
| 192.168.100.0 | 255.255.255.0 | 192.168.14.0 | 20 |  |
| 0.0.0.0 |  | 192.168.1.0 | 1 |  |

* ***Network destination*** - ip-адрес сети получателя, состоит из 2х полей, адреса подсети и маски получателя.
* ***Next router*** – маршрутизатор, который обрабатывает это сообщение.
* ***Metric*** – при поддержки нескольких маршрутов к одному получателю, метрика является показателем того, какой из маршрутов является предпочтительным, причем предпочтительный здесь не определяется каким бы то ни было параметром.

Это может быть параметр, который означает физическую стоимость передачи. Например, если у нас есть маршруты с тарификацией по трафику, то соответственно, нужно выбрать лучшую метрику (чем меньше, тем лучше). Так вот, наименьшую метрику будет иметь маршрут с более низкой стоимостью трафика, а может быть наоборот, в зависимости от того, какие приоритеты мы ставим перед собой. Метрика является некой весовой характеристикой этого маршрутного графа (рис), можно представить, что у нас имеется взвешенный граф, граф с весами ребер, и мы ищем оптимальный маршрут с учетом этих весов.

* ***Interface*** – необязательное поле, физический интерфейс, если на один маршрут указывает несколько физических интерфейсов, то мы можем выбирать, тот который нам наиболее предпочтителен. Кроме того, физ. интерфейс играет определенную роль в сетях получателей, но там мы рассмотрим ARP таблицу.

Всегда присутствует адрес по умолчанию, это адрес, куда перенаправляются все сообщения, которые не имеют прямой подсети получателя, традиционно записывается как четыре нуля и не имеет маски подсети.

Таким образом, ***алгоритм маршрутизации*** действует так: по адресу получателя который указывается в заголовке в любой IP-datagram’e (frame ip), мы определяем наилучшее соответствие между записями в таблице маршрутизации и ip-адресом, причем наилучшее соответствие понимается, таким образом, если у нас существует несколько подсетей, то выбирается сеть с максимальной маской под сети. Причина проста: если у нас в сети получателя меньше хостов, то по теоретическим соображениям, скорее всего маршрутизация туда займет наименьшее кол-во хопов (шагов). Потому что, чем меньше размер сети, тем меньше там подсетей…

Есть протоколы маршрутизации, которые позволяют строить эти таблицы автоматически. Сейчас они построены таким образом, чтобы сократить объем информации, передаваемый по сети. ***Минимизация трафика*** маршрутизации является более важной задачей, чем сокращение кол-ва шагов при доставке датаграммы**!**

В ранние времена, еще в начале 90х годов, основным протоколом маршрутизации служил, так называемый протокол RIP (Routing Internet Protocol). Этот протокол обладал очень неприятным свойством, по этому протоколу маршрутизаторы обменивались целиком маршрутными таблицами, поэтому этот протокол создавал очень большую нагрузку на сеть, потому что они обменивались этими таблицами несколько минут. Было решено сделать агрегирование или объединение сетей в маршрутных таблицах. Если у вас есть 2 сети, которые можно записать в одну с более короткой маской, то это необходимо сделать.

**Пример:** первая сеть 192.168.0.0/24, вторая сеть 192.168.1.0/24 можно записать: 192.168.0.0/23 (маска 255.255.254.0)

Это позволяет сократить объем маршрутных таблиц на маршрутизаторах и снизить объем трафика для передачи этих маршрутных таблиц.

**Какие сети можно агрегировать?**

На самом деле, нужно выписать адреса этих сетей в двоичном формате и посмотреть, какая часть двоичного адреса совпадает (сделать побитовую операцию “и”). Результат выдаст некое двоичное число, и если из этого числа выбрать последовательность непрерывно идущих единиц, то мы получим маску агрегирования сети.

**Пример:**

1. 192.168.0.0/24
2. 192.168.3.0/24

(рассмотрим 3 тетраду т.к. только в ней отличия, представим её в двоичном формате)

1. .0. 🡪 00000000
2. .3. 🡪 00000011

Делаем побитовую операцию &:

11111100 = x

Получили маску 1111111111111111x00000000, т.е. 255.255.252.0 или /22

Эта маска может еще агрегировать сети 192.168.1.0 и 192.168.2.0

**Пусть мы доставили ip-датаграмму в локальную сеть, а как, собственно говоря, доставить датаграмму до получателя в локальной сети?**

*(Как определяется получатель?)*

Дальше маршрутизация методами IP быть обеспечена не может, вам уже необходимо разбирать на уровне сети физического уровня, чаще всего это, безусловно, Ethernet. Собственно, каким образов в этой сети физического уровня найти получателя? Для этого есть протокол ***ARP (address resolution protocol)*.** Строго говоря, ARP, с точки зрения модели ISO-OSI к семейству IP относиться не должен, он должен относиться к категории канальных протоколов, которые работают на втором уровне ISO-OSI, которые соответственно связывают протоколы физического и сетевого уровней, но по устоявшейся традиции протоколы ARP принято изучать вместе с протоколами семейства IP.

**Структура ARP пакета**

|  |  |
| --- | --- |
| **0** **7** | **8 15** |
| **Hardware type** | |
| **Proto type** | |
| **Hardware address length** | **Proto address length** |
| **Operation** | |
| **Sender Hardware address** | |
| **Sender Proto address** | |
| **Target Hardware address** | |
| **Target Proto address** | |

Поля протокола ARP всегда выравниваются по 16-битной границе, последние 4 поля могут иметь произвольную длину, их длина определяется полями приведенными выше.

* ***Hardware type* –** поле определяет протокол физического уровня (Ethernet имеет номер 1)
* ***Proto type*** – протокол сетевого уровня, по устоявшейся традиции, для ipv4 значение этого поля 0x800.
* **Hardware address length** – длина физического адреса
* **Proto address length** – длина адреса сетевого уровня
* **Operation** – коды операций

(Для Ethernet длина физического адреса 6 октет длина сетевого адреса 4 октет)

Последние 4 поля очевидны (адреса как они есть).

**Как протокол ARP работает?**

Изначально протокол ARP работал только на уровне широковещательных запросов. Когда вы осуществляли ***ARP-запрос***, в поле кода операции ставилась единица (1) и отправлялся широковещательный запрос. Если в такой сети есть получатель, то он отвечает уже сугубо индивидуально, и в коде операции ставится двойка (2), это ***ARP-ответ***. Поле sender, он заполнял своими параметрами, а поле target он писал все параметры, которые содержались в запросе, т.е. он отвечал тому отправителю, который запрашивал ip-адрес.

Поле Sender отправитель заполняет своими адресами, в поле Target Hardware address пишем шесть нулей (широковещательный запрос) и в поле Target Proto address пишем ip-адрес получателя с целью определения его физического адреса.

Существует разновидность ARP - это reverse ARP, который по физическому адресу возвращает ip-адрес.

**На самом деле, как это трансформировалось и во что?**

Появился специально выделенный ***ARP host*** , который носит название ***ARP proxy***, во всех реально действующих сетях, обычно, маршрутизатор выступает в качестве APR proxy данной сети. В принципе можно настроить так, что любой из хостов может быть ARP proxy. ARP proxy рассылает по времени (например, 5 минут, можно настроить) ARP запросы, (код операции другой), он просит все хосты сети сообщить ему соответствие физического и логического сетевого адреса, при этом он сообщает свои адреса, после истечения времени он сокращает период запросов до какого-то разумного времени.

Подразумевается, что все включенные хосты должны откликнуться. Если вы включаете после этого хост, то он отправляет ARP-запрос (код операции другой), ARP-proxy должен ему ответить, и сообщить свои адреса. По такой схеме сводится до минимума кол-во широковещательных запросов.

Существуют системы автоматического назначения ip-адресов. Наиболее распространенным является протокол DHCP (dynamic host configuration protocol). Если у вас выдаются адреса таким образом, то самое разумное назначить DHCP сервером тот же самый хост, который является ARP-proxy, даже если он не является маршрутизатором.