1. Внутренний протокол маршрутизации RIP

1.1. История возникновения RIP

Этот протокол маршрутизации предназначен для сравнительно небольших и относительно однородных сетей. Протокол разработан в университете Калифорнии (Беркли), базируется на разработках фирмы Ксерокс. Маршрут здесь характеризуется вектором расстояния до места назначения. Предполагается, что каждый маршрутизатор является отправной точкой нескольких маршрутов до сетей, с которыми он связан. Описания этих маршрутов хранится в специальной маршрутной таблице. Таблица маршрутизации RIP содержит записи на каждый маршрут и должна включать в себя:

§ IP–адрес места назначения.

§ Метрика маршрута (от 1 до 15; число шагов до места назначения).

§ IP–адрес ближайшего маршрутизатора (gateway) по пути к месту назначения.

§ Таймеры маршрута.

Протокол базируется на векторе расстояния (место назначение–направление; метрика–модуль вектора). Периодически (раз в 30 сек) каждый маршрутизатор посылает широковещательно копию своей маршрутной таблицы всем соседям–маршрутизаторам (регулярные коррекции), с которыми связан непосредственно. Маршрутизатор–получатель просматривает таблицу. Если в таблице присутствует новый путь или сообщение о более коротком маршруте, или произошли изменения длин пути, эти изменения фиксируются получателем в своей маршрутной таблице. Протокол RIP должен обрабатывать три типа ошибок:

1. Циклические маршруты. Так как в протоколе нет механизмов выявления замкнутых маршрутов, необходимо либо слепо верить партнерам, либо принимать меры для блокировки такой возможности.

2. Для подавления нестабильностей RIP должен использовать малое значение максимально возможного числа шагов (<16).

3. Медленное распространение маршрутной информации по сети создает проблемы при динамичном изменении маршрутной ситуации (система не поспевает за изменениями). Малое предельное значение метрики улучшает сходимость, но не устраняет проблему.

Несоответствие маршрутной таблицы реальной ситуации типично не только для RIP, но характерно для всех протоколов, базирующихся на векторе расстояния, где информационные сообщения актуализации несут в себе только пары кодов: адрес места назначение и расстояние до него.

Проблема может быть решена следующим образом. Маршрутизатор запоминает, через какой интерфейс получена маршрутная информация, и через этот интерфейс эту информацию уже не передает.

Существуют и другие пути преодоления медленных переходных процессов. Если производится оповещение о коротком пути, все узлы-получатели воспринимают эти данные немедленно. Если же маршрутизатор закрывает какой–то путь, его отмена фиксируется остальными лишь по таймауту. Универсальным методом исключения ошибок при маршрутизации является использование достаточно большой выдержки, перед тем как использовать информацию об изменении маршрутов. В этом случае к моменту изменения маршрута эта информация станет доступной всем участникам процесса маршрутизации. Но многие из этих методов могут при определенных условиях вызвать лавину широковещательных сообщений, что также дезорганизует сеть. Именно малая скорость установления маршрутов в RIP и др. является причиной их постепенного вытеснения новыми протоколами.

В RIP сообщения инкапсулируются в udp–дейтограммы, при этом передача осуществляется через порт 520. В качестве метрики RIP использует число шагов до цели. Если между отправителем и приемником расположено три маршрутизатора (gateway), считается, что между ними 4 шага. Такой вид метрики не учитывает различий в пропускной способности или загруженности отдельных сегментов сети. Применение вектора расстояния не может гарантировать оптимальность выбора маршрута.

Маршрут по умолчанию имеет адрес 0.0.0.0. Каждому маршруту ставится в соответствие таймер тайм–аута и «сборщика мусора». Тайм–аут–таймер сбрасывается каждый раз, когда маршрут инициализируется или корректируется. Если со времени последней коррекции прошло 3 минуты или получено сообщение о том, что вектор расстояния равен 16, маршрут считается закрытым. Но запись о нем не стирается, пока не истечет время «уборки мусора» (2мин).

1.2. Режимы реализации RIP

При реализации RIP можно выделить следующие режимы:

Инициализация, определение всех «живых» интерфейсов путем посылки запросов, получение таблиц маршрутизации от других маршрутизаторов. Часто используются широковещательные запросы.

Получен запрос. В зависимости от типа запроса высылается адресату полная таблица маршрутизации, или проводится индивидуальная обработка.

Получен отклик. Проводится коррекция таблицы маршрутизации (удаление, исправление, добавление).

1.3. Недостатки протокола RIP

a. RIP не работает с адресами субсетей. Если нормальный 16–бит идентификатор ЭВМ класса B не равен 0, RIP не может определить является ли не нулевая часть cубсетевым ID, или полным IP–адресом.

b. RIP требует много времени для восстановления связи после сбоя в маршрутизаторе (минуты). В процессе установления режима возможны циклы.

c. Число шагов важный, но не единственный параметр маршрута, да и 15 шагов не предел для современных сетей.

1.4. Конфигурация RIP

Router2#**configure terminal**

Router2(config)#**interface *Ethernet0***

Router2(config-if)#**ip address *192.168.30.1 255.255.255.0***

Router2(config-if)#**interface *Serial0.1***

Router2(config-subif)#**ip address *172.25.2.2 255.255.255.0***

Router2(config-subif)#**exit**

Router2(config)#**router rip**

Router2(config-router)#**network *172.25.0.0***

Router2(config-router)#**network *192.168.30.0***

Router2(config-router)#**exit**

Router2(config)#**end**

Router2#

2. Протокол OSPF (алгоритм Дикстры)

2.1 Историческая справка

Протокол OSPF (алгоритмы предложены Дикстрой) является альтернативой RIP в качестве внутреннего протокола маршрутизации. OSPF представляет собой протокол состояния маршрута (в качестве метрики используется – коэффициент качества обслуживания). Каждый маршрутизатор обладает полной информацией о состоянии всех интерфейсов всех маршрутизаторов автономной системы. Протокол OSPF реализован в damone маршрутизации gated, который поддерживает также RIP и внешний протокол маршрутизации BGP.

Автономная система может быть разделена на несколько областей, куда могут входить как отдельные ЭВМ, так и целые сети. В этом случае внутренние маршрутизаторы области могут и не иметь информации о топологии остальной части автономной системы. Сеть обычно имеет выделенный (designated) маршрутизатор, который является источником маршрутной информации для остальных маршрутизаторов. Каждый маршрутизатор самостоятельно решает задачу оптимизации маршрутов. Если к месту назначения ведут два или более эквивалентных маршрута, информационный поток будет поделен между ними поровну. Переходные процессы в OSPF завершаются быстрее, чем в RIP. В процессе выбора оптимального маршрута анализируется ориентированный граф сети. Алгоритм Дикстры по выбору оптимального пути: пути с наименьшим суммарным значением метрики считаются наилучшими.

2.2. Качество сервиса

Качество сервиса (QoS) может характеризоваться следующими параметрами:

* пропускной способностью канала;
* задержкой (время распространения пакета);
* числом дейтограмм, стоящих в очереди для передачи;
* загрузкой канала;
* требованиями безопасности;
* типом трафика;
* числом шагов до цели;
* возможностями промежуточных связей (например, многовариантность достижения адресата).

Определяющими являются три характеристики: задержка, пропускная способность и надежность. Для транспортных целей OSPF использует IP непосредственно, т.е. не привлекает протоколы UDP или TCP. OSPF имеет свой код (89) в протокольном поле IP–заголовка. Маршрутизация в этом протоколе определяется IP–адресом и типом сервиса. Так как протокол не требует инкапсуляции пакетов, сильно облегчается управление сетями с большим количеством мостов и сложной топологией (исключается циркуляция пакетов, сокращается транзитный трафик). Автономная система может быть поделена на отдельные области, каждая из которых становится объектом маршрутизации. Этот прием позволяет значительно сократить необходимый объем маршрутной базы данных. В OSPF используется термин опорной сети (backbone) для коммуникаций между выделенными областями. Протокол работает лишь в пределах автономной системы. В пределах выделенной области может работать свой протокол маршрутизации.

Для передачи статусной информации OSPF использует бродкастовые сообщения Hello. Для повышения безопасности предусмотрена авторизация процедур. OSPF–протокол требует резервирования двух мультикастинг–адресов:

224.0.0.5 (предназначен для обращения ко всем маршрутизаторам, поддерживающим этот протокол.)

224.0.0.6 (служит для обращения к специально выделенному маршрутизатору.)

Все OSPF – пакеты ассоциируются с той или иной областью. Большинство из них не преодолевает более одного шага. Пакеты, путешествующие по виртуальным каналам, помечаются идентификатором опорной области (backbone) 0.0.0.0.

Протокол OSPF использует сообщение типа Hello для обмена данными между соседними маршрутизаторами.

Маршрутизаторы обмениваются сообщениями из баз данных OSPF, чтобы инициализировать, а в дальнейшем актуализовать свои базы данных, характеризующие топологию сети. Обмен происходит в режиме клиент–сервер. Клиент подтверждает получение каждого сообщения.

Маршрутизатор, получивший OSPF–пакет, посылает подтверждение его приема. Получение нескольких объявлений о состоянии линий может быть подтверждено одним пакетом. Адресом места назначения этого пакета может быть индивидуальный маршрутизатор, группа маршрутизаторов или все маршрутизаторы автономной системы.

Маршрутная таблица OSPF содержит в себе:

* IP–адрес места назначения и маску;
* тип места назначения (сеть, граничный маршрутизатор и т.д.);
* тип функции (возможен набор маршрутизаторов для каждой из функций TOS);
* область (описывает область, связь с которой ведет к цели, возможно несколько записей данного типа, если области действия граничных маршрутизаторов перекрываются);
* тип пути (характеризует путь как внутренний, межобластной или внешний, ведущий к AS);
* цена маршрута до цели;
* очередной маршрутизатор, куда следует послать дейтограмму;
* объявляющий маршрутизатор (используется для межобластных обменов и для связей автономных систем друг с другом).

2.3. Преимущества и недостатки OSPF

Преимущества:

1. Для каждого адреса может быть несколько маршрутных таблиц, по одной на каждый вид IP–операции (TOS).

2. Каждому интерфейсу присваивается безразмерная цена, учитывающая пропускную способность, время транспортировки сообщения. Для каждой IP–операции может быть присвоена своя цена (коэффициент качества).

3. При существовании эквивалентных маршрутов OSFP распределяет поток равномерно по этим маршрутам.

4. Поддерживается адресация субсетей (разные маски для разных маршрутов).

5. При связи точка–точка не требуется IP–адрес для каждого из концов. (Экономия адресов)

6. Применение мультикастинга вместо широковещательных сообщений снижает загрузку не вовлеченных сегментов.

Недостатки:

1. Трудно получить информацию о предпочтительности каналов для узлов, поддерживающих другие протоколы, или со статической маршрутизацией.

2. OSPF является лишь внутренним протоколом.

2.4 Конфигурация OSPF

Router1#**configure terminal**

Enter configuration commands, one per line. End with CNTL/Z.

Router1(config)#**router ospf *55***

Router1(config-router)#**network *10.0.0.0 0.255.255.255* area *2***

Router1(config-router)#**network *172.20.0.0 0.0.255.255* area *100***

Router1(config-router)#**network *0.0.0.0 255.255.255.255* area *0***

Router1(config-router)#**exit**

Router1(config)#**end**

Router1#

3. Внешний протокол BGP

3.1. Историческая справка

Протокол BGP разработан компаниями IBM и CISCO. Главная цель BGP– сократить транзитный трафик. Местный трафик либо начинается, либо завершается в автономной системе (AS); в противном случае–это транзитный трафик. Системы без транзитного трафика не нуждаются в BGP. Но не всякая ЭВМ, использующая протокол BGP, является маршрутизатором, даже если она обменивается маршрутной информацией с пограничным маршрутизатором соседней автономной системы. AS передает информацию только о маршрутах, которыми она сама пользуется. BGP–маршрутизаторы обмениваются сообщениями об изменении маршрутов: UPDATE–сообщения (имеют поля: маркер, длина, тип…). Максимальная длина таких сообщений составляет 4096 октетов, а минимальная 19 октетов. Каждое сообщение имеет заголовок фиксированного размера. Объем информационных полей зависит от типа сообщения.

Поле маркер содержит 16 октетов и его содержимое может легко интерпретироваться получателем. Если тип сообщения «OPEN», или если код идентификации в сообщении open равен нулю, то поле маркер должно быть заполнено единицами. Маркер может использоваться для обнаружения потери синхронизации в работе BGP–партнеров. Поле длина имеет два октета и определяет общую длину сообщения в октетах, включая заголовок. Значение этого поля должно лежать в пределах 19–4096. Поле тип представляет собой код разновидности сообщения и может принимать следующие значения:

|  |  |
| --- | --- |
| 1 OPEN | (открыть) |
| 2 UPDATE | (изменить) |
| 3 NOTIFICATION | (внимание) |
| 4 KEEPALIVE | (еще жив) |

После того как связь на транспортном протокольном уровне установлена, первое сообщение, которое должно быть послано–это OPEN. При успешном прохождении этого сообщения партнер должен откликнуться сообщением KEEPALIVE («Еще жив»). После этого возможны любые сообщения.

Одно–октетный код идентификации позволяет организовать систему доступа, если он равен нулю, маркер всех сообщений заполняется единицами, а поле идентификационных данных должно иметь нулевую длину. При неравном нулю коде идентификации должна быть определена процедура доступа и алгоритм вычисления кодов поля маркера. Длина поля идентификационных данных определяется по формуле:

Длина сообщения = 29 + длина поля идентификационных данных.

Минимальная длина сообщения open составляет 29 октетов, включая заголовок.

Сообщения типа UPDATE (изменения) используются для передачи маршрутной информации между BGP–партнерами. Этот тип сообщения позволяет сообщить об одном новом маршруте или объявить о закрытии группы маршрутов, причем объявление об открытии нового и закрытии старых маршрутов возможно в пределах одного сообщения.

Атрибуты пути бывают «стандартные обязательные» (well–known mandatory), «стандартные на усмотрение оператора», «опционные переходные» и «опционные непереходные». Стандартные атрибуты должны распознаваться любыми BGP–приложениями. Опционные атрибуты могут не распознаваться некоторыми приложениями. Обработка нераспознанных атрибутов задается битом 1 поля флагов. Пути с нераспознанными переходными опционными атрибутами должны восприниматься, как рабочие. Один и тот же атрибут может появляться в списке атрибутов пути только один раз.

Предусмотрены следующие разновидности кодов типа атрибута:

ORIGIN (код типа = 1) – стандартный обязательный атрибут, который определяет происхождение путевой информации:

|  |  |
| --- | --- |
| Код атрибута | Описание |
| 0 | IGP – информация достижимости сетевого уровня является внутренней по отношению к исходной автономной системе; |
| 1 | EGP – информация достижимости сетевого уровня получена с помощью внешнего протокола маршрутизации; |
| 2 | Incomplete – информация достижимости сетевого уровня получена каким–то иным способом. |

3.3. Отличительные особенности BGP

BGP отличается от RIP и OSPF тем, что использует TCP в качестве транспортного протокола. Две системы, использующие BGP, связываются друг с другом и пересылают посредством TCP полные таблицы маршрутизации. В дальнейшем обмен идет только в случае каких–то изменений. ЭВМ, использующая BGP, не обязательно является маршрутизатором. Сообщения обрабатываются только после того, как они полностью получены.

BGP является протоколом, ориентирующимся на вектор расстояния. Вектор описывается списком AS по 16 бит на AS. BGP регулярно (каждые 30сек) посылает соседям TCP–сообщения, подтверждающие, что узел жив (это не тоже самое что «Keepalive» функция в TCP). Если два BGP–маршрутизатора попытаются установить связь друг с другом одновременно, такие две связи могут быть установлены. Такая ситуация называется столкновением, одна из связей должна быть ликвидирована. При установлении связи маршрутизаторов сначала делается попытка реализовать высший из протоколов (например, BGP–4), если один из них не поддерживает эту версию, номер версии понижается.

Разница между BGP и OSPF

Чтобы лучше понять разницу между BGP и OSPF, надо уяснить, что такое "автономная система".

OSPF работает только внутри автономной системы, BGP - только между автономными системами, и никак иначе.

фроутерами внутри автономной системы и передачей информации о маршрутах BGP в один из Interior Gateway Protocols (IGPs), которые работают внутри автономной системы (например, IGRP, IS-IS, RIP и OSPF).

BGP использует TCP в качестве транспортного протокола (port 179). Любые два роутера, между которыми открыто tcp соединение для обмена информацией о маршрутизации, называются "peers" или "neighbors". На приведенном выше рисунке роутеры A и B являются BGP парами, равно как и роутеры B и C, C и D. Таблица маршрутизации состоит из незакольцованной (loop-free) карты автономных систем. Заметим, что внутри AS BGP peers не являются подключенными непосредственно друг к другу (грубо говоря, не соединены между собой напрямую проводом).

BGP peers инициируют обмен полными таблицами BGP маршрутизации между собой. Позже они посылают лишь incremental updates. Кроме этого, BGP peers обмениваются keepalive messages (чтобы удостовериться, что связь между ними не потеряна), и notification messages (сообщениями об ошибках и другой служебной информацией).

Для простой конфигурации роутеров используются следующие команды:

***Router A***

***router bgp 100***

***neighbor 129.213.1.1 remote-as 200***

Как видно из приведенной выше конфигурации, router bgp 100 команды присваивают роутеру номер его AS и включают обмен BGP маршрутами между автономными системами.

Команда конфигурации **neighbor [ip address] remote-as [AS]** добавляет запись в таблицу маршрутизации BGP, и в этой записи говорится о том, что peer идентифицируется его IP адресом внутри его автономной системы. Для роутеров, в которых используется EBGP, neighbors обычно соединены напрямую, и IP адрес neighbor'а - это IP адрес сетевого интерфейса граничной с вашим роутером стороны.

Для роутеров, на которых используется IBGP, IP адресом может быть адрес любого из интерфейсов роутера в сети, которая входит в вашу AS.

**IP – адрессация в сетях**

**IP-адрес** — сетевой адрес узла в компьютерной сети, построенной по протоколу IP. При связи через сеть Интернет требуется глобальная уникальность адреса, в случае работы в локальной сети требуется уникальность адреса в пределах сети.

**IPv4** (англ. *Internet Protocol version 4*) — четвёртая версия IP протокола, первая широкоиспользуемая версия .

## Адресация

Каждомукомпьютерув Сети Интернет присвоен IP-адрес. ("IP" – Internet Protocol). Таким образом,IP-адрес – это адрес компьютера в Интернете в соответствии с протоколом IP.

IPv4 использует 32-битные (четырёхбайтные) адреса, ограничивающие адресное пространство 4 294 967 296 (232) возможными уникальными адресами. Удобной формой записи IP-адреса (**IPv4**) является запись в виде четырёх десятичных чисел (от 0 до 255), разделённых точками, например, *192.168.0.1*. (или *128.10.2.30* — традиционная десятичная форма представления адреса. Каждый компьютер, подключенный к Интернету, имеет свой уникальный IP-адрес. Так, к примеру, IP-адреса 192.28.543.315,  182.560.45.970,  192.28.543.316 принадлежат разным компьютерам.

Каждое устройство на сети должно быть уникально определено. На сетевом уровне пакеты должны быть идентифицированы с источником и адресами получателя двух систем конца. С IPv4 это означает, что у каждого пакета есть 32-битовый адрес отправителя и 32-битовый адрес получателя.

Двоичная запись

Двоичную запись адреса IPv4 можно представить в десятичном виде, отделяя каждый байт двоичной записи (называемую октетом) точкой. Это называют октетом, потому что каждое десятичное число представляет один байт или 8 битов.

Например, адрес:

10101100000100000000010000010100

выражен в десятичном виде как:

172.16.4.20

Устройства используют бинарную логику. Десятичный формат используется, чтобы облегчить для людей использование и запоминание адреса.

Адресные части сети и хостов

Для каждого адреса IPv4 некоторая часть старших битов представляет адрес сети. В третьем уровне мы определяем сеть как группу хостов, у которых есть идентичные битовые комбинации в части адреса сети их адресов.

Хотя все 32 бита определяют адрес хоста IPv4, у нас есть переменное число битов, которые называют частью хоста адреса. Число битов, используемых в этой части, определяет число хостов, которых мы можем иметь в пределах сети.

Например, если бы у нас должно быть по крайней мере 200 пользователей в сети, мы должны были бы использовать столько битов в части хоста, чтобы быть в состоянии представить по крайней мере 200 различных битовых комбинаций.

Чтобы назначить уникальное обращение к 200 пользователям, мы использовали бы весь последний октет. С 8 битами могут быть достигнуты в общей сложности 256 различных битовых комбинаций. Это означало бы, что биты для верхних трех октетов представят часть сети.

В пределах адресного пространства каждой сети IPv4 у нас есть три типа адресов:

Адрес сети - адрес, которым мы обращаемся к сети.

Бродкастовый адрес - специальный адрес, который рассылает данные всем пользователям в сети.

Адреса хоста - адреса оконечного устройства в сети.

**Адрес сети**

Адрес сети - стандартный способ обратиться к сети. Например, мы могли обратиться к сети как "10.0.0.0 сеть." Это - намного более удобный и описательный способ обратиться к сети чем использование такого термина как "первая сеть." У всех хозяев в 10.0.0.0 сетях будут те же самые биты сети.

В пределах адресного пространства IPv4 сети самый низкий адрес сохранен для адреса сети. У этого адреса есть 0 для каждого бита хоста в части адреса хоста.

**Бродкастовый адрес**

Бродкастовый адрес IPv4 - специальный адрес для каждой сети, который рассылает данные всем пользователям в сети. Чтобы послать данные всем пользователям сети, пользователь может послать единственный пакет, который адресован бродкастовому адресу сети.

Бродкастовый адрес использует самый высокий адрес в диапазоне сети. Это - адрес, в котором биты в части хоста - все 1s. Для сети 10.0.0.0 с 24 битами сети, бродкастовый адрес был бы 10.0.0.255.

**Адреса хоста**

Как описано ранее, каждое оконечное устройство требует, чтобы уникальный адрес соответствовал определенному пользователю.

Диапазон адресов IPv4 расположен в пределах от 0.0.0.0 до 255.255.255.255. Однако не все адреса используются в сети. Существуют так называемые экспериментальные адреса.

**Экспериментальные Адреса.**

Самый главный блок адресов, сохраненных для особых целей, является экспериментальным адресным пространством IPv4 240.0.0.0 … 255.255.255.254. В настоящее время эти адреса сохранены для будущего использования (RFC 3330). Это предполагает, что они могли быть преобразованы в адреса годные к использованию. В настоящее время они не могут использоваться в сетях IPv4. Однако, эти адреса могли использоваться для определенных исследований.

**Мультикастовые адреса**

Еще один блок адресов, сохраненных для особых целей, является адресным пространством передачи IPv4 224.0.0.0 … 239.255.255.255.

Мультикастовые адреса диапазона 224.0.0.0 … 224.0.0.255, так называемые местные адреса. Эти адреса должны использоваться для групп передачи на местной сети.

Адреса 224.0.1.0 … 238.255.255.255. Они могут использоваться для передачи данных через Интернет. Например, 224.0.1.1 был сохранен для протокола NTP, чтобы синхронизировать время и дату с устройствами сети.

# Бесклассовая адресация

**Бесклассовая адресация** — метод IP-адресации, позволяющий гибко управлять пространством IP-адресов, не используя жёсткие рамки классовой адресации. Использование этого метода позволяет экономно использовать ограниченный ресурс IP-адресов, поскольку возможно применение различных масок подсетей к различным подсетям.

## Диапазоны адресов

IP-адрес является массивом битов. Принцип IP-адресации — выделение множества (диапазона, блока, подсети) IP-адресов, в котором некоторые битовые разряды имеют фиксированные значения, а остальные разряды пробегают все возможные значения. Блок адресов задаётся указанием начального адреса и маски подсети. Бесклассовая адресация основывается на переменной длине маски подсети (англ. Variable Length Subnet Mask — VLSM), в то время, как в классовой (традиционной) адресации длина маски строго фиксирована 0, 1, 2 или 3 установленными октетами.

Вот пример записи IP-адреса с применением бесклассовой адресации: **192.0.2.32/27**.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**октеты**](http://ru.wikipedia.org/wiki/%D0%9E%D0%BA%D1%82%D0%B5%D1%82) **IP-адреса** | 192 | | | | | | | | 0 | | | | | | | | 2 | | | | | | | | 32 | | | | | | | |
| **биты IP-адреса** | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| **биты маски подсети** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| [**октеты**](http://ru.wikipedia.org/wiki/%D0%9E%D0%BA%D1%82%D0%B5%D1%82) **маски подсети** | 255 | | | | | | | | 255 | | | | | | | | 255 | | | | | | | | 224 | | | | | | | |

В данном примере видно, что в маске подсети 27 бит слева выставлены в единицу (так называемые *значащие биты*. В таком случае говорят о длине маски подсети в 27 бит (/27 — «**слэш двадцать семь**»).

Вот ещё один пример записи адреса в бесклассовой [нотации](http://ru.wikipedia.org/wiki/%D0%9D%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D1%8F): **172.16.0.1/12**.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **октеты IP-адреса** | 172 | | | | | | | | 16 | | | | | | | | 0 | | | | | | | | 1 | | | | | | | |
| **биты IP-адреса** | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **биты маски подсети** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **октеты маски подсети** | 255 | | | | | | | | 240 | | | | | | | | 0 | | | | | | | | 0 | | | | | | | |

Другие примеры: множество всех адресов обозначается как /0, а конкретный адрес IPv4 — как /32.

Для упрощения таблиц маршрутизации можно объединять блоки адресов, указывая один большой блок вместо ряда мелких. Например, 4 смежные сети класса C (4 × 255 адресов, маска 255.255.255.0 или /24) могут быть объединены, с точки зрения далёких от них маршрутизаторов, в одну сеть /22. И напротив, сети можно разбивать на более мелкие подсети, и так далее.

## Математическое обоснование

С точки зрения бесклассовой двоичной адресации пространство IP-адресов рассматривается как [ультраметрическое](http://ru.wikipedia.org/wiki/%D0%A3%D0%BB%D1%8C%D1%82%D1%80%D0%B0%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE). Разные блоки адресов являются в нём [шара́ми](http://ru.wikipedia.org/wiki/%D0%A8%D0%B0%D1%80_%28%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B3%D0%B5%D0%BE%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%8F%29), радиус которых убывает с увеличением n, и сами они формируют направленное двоичное [дерево](http://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%28%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2%29). То есть, от каждого блока (/n, 0\leqslant n < 32для IPv4) можно «перейти» на один из двух блоков меньшего размера (/*n*+1), из которых он состоит.

## Возможные маски

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **адресов** | **битов** | **префикс** | **класс** | **Маска** |
| 1 | 0 | /32 |  | 255.255.255.255 |
| 2 | 1 | /31 |  | 255.255.255.254 |
| 4 | 2 | /30 |  | 255.255.255.252 |
| 8 | 3 | /29 |  | 255.255.255.248 |
| 16 | 4 | /28 |  | 255.255.255.240 |
| 32 | 5 | /27 |  | 255.255.255.224 |
| 64 | 6 | /26 |  | 255.255.255.192 |
| 128 | 7 | /25 |  | 255.255.255.128 |
| 256 | 8 | /24 | 1C | 255.255.255.0 |
| 512 | 9 | /23 | 2C | 255.255.254.0 |
| 1Ki | 10 | /22 | 4C | 255.255.252.0 |
| 2Ki | 11 | /21 | 8C | 255.255.248.0 |
| 4Ki | 12 | /20 | 16C | 255.255.240.0 |
| 8Ki | 13 | /19 | 32C | 255.255.224.0 |
| 16Ki | 14 | /18 | 64C | 255.255.192.0 |
| 32Ki | 15 | /17 | 128C | 255.255.128.0 |
| 64Ki | 16 | /16 | 1B | 255.255.0.0 |
| 128Ki | 17 | /15 | 2B | 255.254.0.0 |
| 256Ki | 18 | /14 | 4B | 255.252.0.0 |
| 512Ki | 19 | /13 | 8B | 255.248.0.0 |
| 1Mi | 20 | /12 | 16B | 255.240.0.0 |
| 2Mi | 21 | /11 | 32B | 255.224.0.0 |
| 4Mi | 22 | /10 | 64B | 255.192.0.0 |
| 8Mi | 23 | /9 | 128B | 255.128.0.0 |
| 16Mi | 24 | /8 | 1A | 255.0.0.0 |
| 32Mi | 25 | /7 | 2A | 254.0.0.0 |
| 64Mi | 26 | /6 | 4A | 252.0.0.0 |
| 128Mi | 27 | /5 | 8A | 248.0.0.0 |
| 256Mi | 28 | /4 | 16A | 240.0.0.0 |
| 512Mi | 29 | /3 | 32A | 224.0.0.0 |
| 1024Mi | 30 | /2 | 64A | 192.0.0.0 |
| 2048Mi | 31 | /1 | 128A | 128.0.0.0 |
| 4096Mi | 32 | /0 | 256A | 0.0.0.0 |

**Ki** = 210 = 1024 ([«киби»](http://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D1%8B%D0%B5_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B8))

**Mi** = 220 = 1 048 576 ([«меби»](http://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D1%8B%D0%B5_%D0%BF%D1%80%D0%B8%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B8))

**Общественные и частные адреса**

Хотя большинство адресов хозяина IPv4 - общественные адреса, определяемые для использования в сетях, которые доступны в Интернете, есть блоки адресов, которые используются в сетях, которые требуют ограниченный или запрещенный доступ в Интернет. Эти адреса называются частными адресами.

Частные Адреса

Частные блоки адреса:

10.0.0.0 к 10.255.255.255 (10.0.0.0 / 8)

172.16.0.0 к 172.31.255.255 (172.16.0.0 / 12)

192.168.0.0 к 192.168.255.255 (192.168.0.0 / 16)

Блоки адреса личного пространства, как показано в числе, обойдены для использования в частных сетях. Использование этих адресов не должно быть уникальным среди внешних сетей. Пользователи, которые не требуют доступа к Интернету в целом, могут сделать неограниченное использование частных адресов. Однако, внутренние сети все еще должны проектировать схемы адреса сети гарантировать, что хозяева в частных сетях используют IP-адреса, которые уникальны в пределах их сетевой среды.

Общественные Адреса

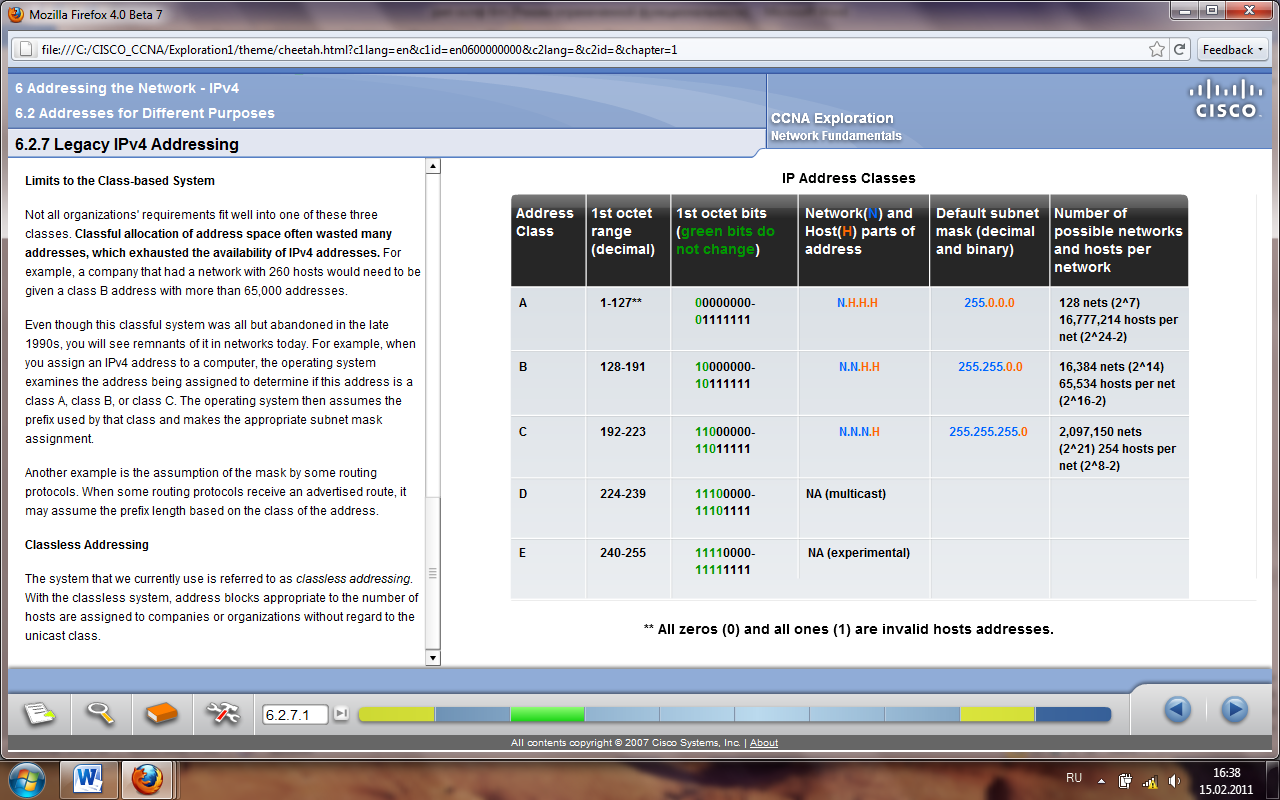
Огромное большинство адресов в IPv4 используют диапазон адресов, являющихся общественными адресами. Эти адреса разработаны, чтобы быть доступными из Интернета. Даже в пределах этих блоков адреса, есть много адресов, которые определяются для других особых целей.

## 

Исторические Классы Сети

Исторически, RFC1700 сгруппировал диапазоны адресов в определенные размеры, названные классом A, классом B, и адресами класса C. Это также определило класс D (мультикастовые) и класс E (экспериментальные) адреса, как ранее.

Классы A, B, и C определили размеры сети и блоки адресов для этих сетей. Компании или организации назначили весь класс A, класс B, или блок адреса класса C. Это использование адресного пространства упоминается как полноклассовая адресация.



Классификация блоков адресов

Сети класса А были разработаны, чтобы поддержать чрезвычайно большие сети больше чем с 16 миллионами адресов хозяина. IPv4, адреса использовали неподвижную/8 приставку с первым октетом, чтобы указать на адрес сети. Оставление тремя октетами использовалось для адресов хозяина.

Чтобы зарезервировать адресное пространство для остающихся классов адреса, весь класс, адреса потребовали, чтобы самая существенная часть старшего октета была нолем. Это означало, что было только 128 возможных классов сети, 0.0.0.0 / 8 к 127.0.0.0 / 8, перед выниманием сохраненных блоков адреса. Даже при том, что класс адреса зарезервировали половину адресного пространства из-за их предела 128 сетей, они могли только быть ассигнованы приблизительно 120 компаниям или организациям.

Сети класса B

Адресное пространство класса B было разработано, чтобы поддержать потребности умеренных к большого размера сетям больше чем с 65 000 хостов. IP-адрес класса B использовал два старших октета, чтобы указать на адрес сети. Другие два октета определили адреса пользователя. Как с классом A, должно было быть сохранено адресное пространство для остающихся классов адреса.

Для адресов класса B самые существенные два бита старшего октета были 10. Это ограничило блок адреса для класса B к 128.0.0.0 / 16 к 191.255.0.0 / 16. У класса B было немного более эффективное распределение адресов чем класс A, потому что это одинаково разделило 25 % полного адресного пространства IPv4 приблизительно среди 16 000 сетей.

Сети класса C

Адресное пространство класса C было обычно доступно из исторических классов адреса. Это адресное пространство было предназначено, чтобы обеспечить адреса для маленьких сетей с максимумом 254 хоста. Сети адреса класса C использовали/24 приставку. Это означало, что сеть класса C использовала только последний октет в качестве адресов хозяина с тремя старшими октетами, используемыми, чтобы указать на адрес сети. Адрес класса C блокирует отложенное адресное пространство для класса D (передача) и класс E (экспериментальный) при использовании постоянного значения 110 для трех самых существенных частей старшего октета. Это ограничило блок адреса для класса C к 192.0.0.0 / 16 к 223.255.255.0 / 16. Хотя это заняло только 12.5 % полного адресного пространства IPv4, это могло обеспечить обращения к 2 миллионам сетей.

Обратная маска сети

|  |
| --- |
| **Wildcard mask** – обратная маска. Чаще всего используется при создании списков доступа. Обратная маска предназначена для выполнения роли указателя, она указывает на IP адреса отдельно взятой подсети, которые необходимо обработать согласно правилу списка доступа. Самая распространенная шаблонная маска - это маска 0.0.0.255. Так, например, если у нас есть сеть 192.168.1.0 255.255.255.0 и нам необходимо создать список доступа, который будет запрещать весь входящий траффик из этой сети, шаблонная маска 0.0.0.255 подойдет как раз кстати. Но это всего лишь вершина айсберга. В сущности шаблонная маска представляет собой более сложный и очень мощный инструмент для выборки значений. Как применять шаблонные маски, мы разберемся далее.  Возьмем, к примеру, шаблонную маску 0.0.240.255 и сеть 172.16.10.0/24. Чтобы понять как она работает, необходимо представить шаблонную маску в двоичном виде, также как и перевести в двоичную систему саму подсеть.  Двоичный вид шаблонной маски **0.0.240.255** будет представлять из себя:  00000000.00000000.11110000.11111111  Двоичный вид сети **172.16.10.0** будет выглядеть так:  10101100.00010000.00001010.00000000  Сопоставим шаблонную маску и сеть:  00000000.00000000.11110000.11111111 - WM  10101100.00010000.00001010.00000000 - IP    Значения IP адреса, которые находятся под значением 0 шаблонной маски находятся вне ее диапазона, поэтому закрасим их серым цветом (шалбонной маской они учитываться не будут):  00000000.00000000.11110000.11111111 - WM  10101100.00010000.00001010.00000000 - IP  Все значения IP адреса, находящиеся под значением 1 шаблонной маски находятся в ее диапазоне и будут учитываться. Обозначим их зеленым цветом:  00000000.00000000.**1111**0000.**11111111** - WM  10101100.00010000.**0000**1010.**00000000** - IP  Мы видим, что все значения четвертого октета IP сети находятся в диапазоне, который охватывает шаблонная маска. Если быть более точным, то в позиции четвертого октета IP сети может стоять любое число от 0 до 255, а это, как вы понимаете, представляет собой количество различных IP адресов, которые могут быть выданы сетевым устройствам.  Выясним теперь, какие же значения третьего октета будут находиться в диапазоне нашей шаблонной маски. Для этого начнем изменять значения, стоящие под значением 1 шаблонной маски в третьем октете. Обозначим изменения красным цветом:  00000000.00000000.**1111**0000.**11111111** - WM  10101100.00010000.**0001**1010.**00000000** - IP  Переведем получившийся IP адрес из двоичной в десятичную систему и получим следующее значение: **172.16.26.0**. Заметим, что на месте четвертого октета в данный момент стоит 0, но как мы помним, в данной позиции может стоять любое число от 0 до 255. Отсюда мы можем сделать вывод, что у нас получилась подсеть 172.16.26.0 255.255.255.0, которую охватывает шаблонная маска 0.0.240.255.  Далее изменим еще одно значение:  00000000.00000000.**1111**0000.**11111111** - WM  10101100.00010000.**0010**1010.**00000000** - IP  Получим сеть **172.16.42.0/24**.  Если будем накручивать числа дальше, то получим следующий двоичный код:  00000000.00000000.**1111**0000.**11111111** - WM  10101100.00010000.**0011**1010.**00000000** - IP  Переведем обратно в десятичную систему и получим сеть **172.16.58.0/24**.  Таким образом можно продолжать пока не переберем все возможные сети с шаблонной маской 0.0.240.255.   Возможно у вас возник вопрос, для чего мы использовали обратную маску и что делать с теми подсетями, которые охватывает данная обратная маска.  Допустим у вас огромная компания, которая состоит из следующих подсетей:   * 172.16.1.0/24 * 172.16.2.0/24 * ... * 172.16.9.0/24 * 172.16.10.0/24 * 172.16.11.0/24 * ... * 172.16.25.0/24 * 172.16.26.0/24 * 172.16.27.0/24 * ... * 172.16.41.0/24 * 172.16.42.0/24 * 172.16.43.0 * ...   Предположим, босс дал задание, чтобы красным подсетям был закрыт доступ к сети Интернет. Вместо того, чтобы писать огромный список доступа, указывая каждую отдельную сеть, можно воспользоваться шаблонной маской 0.0.240.255 для сети 172.16.10.0, которая затронет исключительно красные сети. Все остальные сети будут вне ее диапазона и поэтому не будут подвергнуты правилу списка доступа. Если вы были внимательны, то могли заметить, что шаблонная маска 0.0.240.255 охватывает каждую 16-ю по счету сеть. |