ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Северо-Кавказский филиал ордена Трудового Красного Знамени
федерального государственного бюджетного образовательного учреждения
высшего образования
«Московский технический университет связи и информатики»

Манин А.А.

Сосновский И.А.

Системы коммутации. Принципы и технологии пакетной коммутации

Учебное пособие

Издание 2-е, переработанное и дополненное

Ростов-на-Дону

2017 г.

УДК 621.395
ББК 32.88
М23

Манин А.А., Сосновский И.А. Системы коммутации. Принципы и технологии пакетной коммутации: у*чеб. пособие.*- 2-е изд, перераб. и доп. – Ростов-на-Дону: СКФ МТУСИ, 2017. – 198 с.

В учебном пособии рассмотрены принципы и технологии пакетной коммутации в сетях связи, построенных на базе стека протоколов TCP/IP. Рассмотрены технологии канального (на примере семейства технологий Ethernet), сетевого и транспортного уровней модели ISO/OSI, принципы функционирования сетевых и оконечных устройств, особенности их конфигурирования на примере оборудования Cisco Systems.

Пособие предназначено для студентов, обучающихся по направлению 11.03.02 «Инфокоммуникационные технологии и системы связи» (бакалавриат) дневной и заочной форм обучения, профиль «Сети связи и системы коммутации».

Рецензенты:
Д.т.н., профессор Безуглов Д.А.
Доцент, к.т.н. Болдырихин Н.В.

И з д а т е л ь с т в о С К Ф М Т У С И

Сдано в набор 20.10.17. Изд. № 255. Подписано в печать 20.11.17. Зак. 269.
Печ. листов 12,1. Учетно-изд. л. 9,7. Печать оперативная. Тир. 100 экз.
Отпечатано в Полиграфическом центре СКФ МТУСИ, Серафимовича, 62.

СОДЕРЖАНИЕ

Введение

Как известно [1], основным направлением развития сетей связи является переход к сетям следующего поколения (NGN – Next Generation Network). Рассмотрению принципов NGN посвящены последующие дисциплины учебного плана. Одной из особенностей таких сетей является передача всех видов трафика на единой транспортной платформе, основанной на технологиях коммутации пакетов [1, 2]. Кроме того, коммутация пакетов уже достаточно давно используется в существующих сетях как фиксированной, так и подвижной связи (начиная с третьего поколения). Таким образом, для успешного освоения последующих дисциплин учебного плана необходимо детально изучить принципы пакетной коммутации.

В настоящее время существует достаточно большое количество технологий пакетной коммутации – X.25, Frame Relay, ATM, SMDS, IP [3]. При этом инфокоммуникационные сети в настоящее время предполагают передачу данных на основе технологии IP, поэтому в настоящем пособии основное внимание уделяется именно этой технологии. Как известно, IP-пакет передается в составе кадра (Frame) той канальной технологии, на базе которой построена сеть. В качестве технологий канального уровня будем рассматривать семейство технологий Ethernet как наиболее распространенное (но не единственное). В отличие от большинства литературных источников, посвященных Ethernet, в настоящем пособии рассматриваются практические примеры конфигурирования коммутаторов на примере оборудования Cisco Systems.

При рассмотрении протоколов сетевого уровня рассматривается интернет-протокол четвертой версии (IPv.4) как наиболее широко использующийся в настоящее время. Кроме того, рассматриваются

распространенные протоколы маршрутизации, а также примеры конфигурирования маршрутизаторов.

При рассмотрении технологий транспортного уровня основное внимание уделяется протоколу TCP, но рассматриваются и другие протоколы – UDP, RTP, RTCP.

Представленный в учебном пособии материал является основой, необходимой для изучения остальных дисциплин учебного плана направления подготовки 11.03.02 «Инфокоммуникационные технологии и системы связи», и позволит студенту составить целостное представление о принципах распределения информации в современных инфокоммуникационных сетях. Первое издание данного пособия [4] было существенно дополнено, в частности, в настоящем пособии рассмотрены протоколы динамической маршрутизации, даны основы функционирования протокола транспортного уровня TCP, показаны основные приемы конфигурирования NAT и ACL.

1 Основы сетевых технологий

1.1 Способы коммутации в сетях связи

Современные сети представляют собой комплекс аппаратных и программных средств, предназначенных для передачи информации с заданными параметрами качества. Это отражает и название таких сетей – инфокоммуникационные, то есть такие сети построены на базе как информационных, так и телекоммуникационных технологий [5].

Пользователь, получающий инфокоммуникационные услуги, использует оконечное оборудование с соответствующим программным обеспечением – компьютер с сетевой картой, IP-телефон, IP-видеокамеру, и т.д. Непосредственное соединение оконечного оборудования между собой в большой сети обычно невозможно, поэтому оконечное оборудование соединяется с коммутационным узлом (КУ), обеспечивающим коммутацию передаваемой информации.

Под коммутацией в общем случае понимается процесс распределения поступающих пользовательских данных в соответствии с адресной информацией. Наборы формализованных правил, описывающих формат представления передаваемых данных, адресной информации, способы ее обработки, методы кодирования и передачи, называются телекоммуникационными протоколами.

Рассмотрим основные способы коммутации, использующиеся в современных сетях связи.

Как указывалось выше, под коммутацией в общем случае понимается распределение поступающей информации. Различают три способа коммутации – каналов, сообщений и пакетов.

Под коммутацией каналов понимается образование сквозного тракта передачи информации через определенное количество коммутационных

узлов. Соответственно, фазе передачи информации предшествует фаза установления соединения.

С учетом этого при коммутации каналов всегда можно выделить три фазы:

- установление соединения;

- передача информации;

- разрушение соединения.

Для установления соединения на коммутационный узел должна быть передана адресная информация, причем эта информация должна быть принята узлом до начала процесса установления соединения. Например, в телефонной сети общего пользования (типичный пример сети с коммутацией каналов) на коммутационный узел должен быть передан номер вызываемого абонента.

Одним из основных признаков, отличающим коммутацию каналов от остальных способов коммутации, является то, что при физической невозможности установления соединения абонент получает отказ в обслуживании. Поэтому сеть с технологией коммутации каналов называют сетью с отказами.

Очевидно, что качество обслуживания в сети с отказами можно оценить вероятностью отказа при попытке установления соединения – чем эта вероятность ниже, тем качество обслуживания выше. При этом вероятность отказа на участке сети, состоящем из нескольких транзитных участков, всегда выше, чем вероятность отказа на каждом из участков.

Для большей наглядности рассмотрим участок сети, представленный на рисунке 1.1.

На рисунке символом О обозначено оконечное оборудование пользователя (в простейшем случае – телефон), а КУ i – i -й коммутационный узел, обеспечивающий соединение.

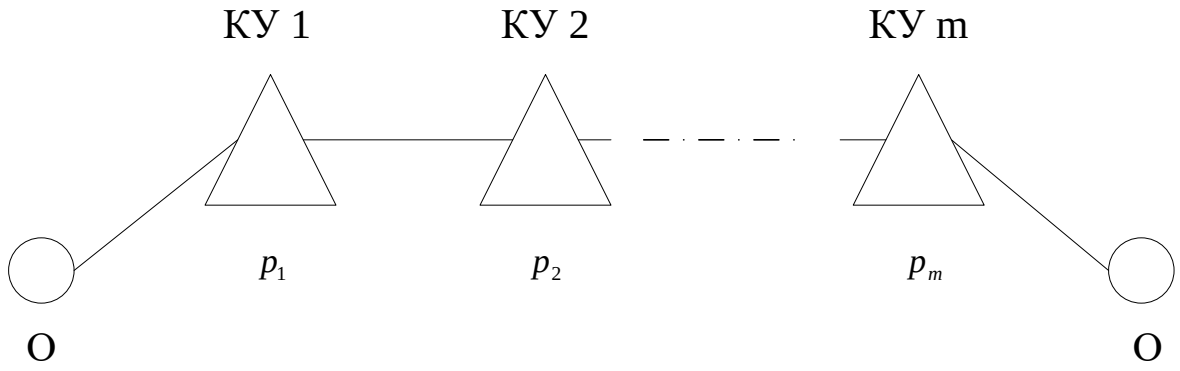

Рисунок 1.1 – Участок сети связи

Обозначим через p_1, p_2, …, p_m вероятности отказов на каждом из m транзитных участков. На каждом из участков отказ в обслуживании и обслуживание составляют полную группу несовместных событий, то есть сумма вероятностей этих событий равна единице. Соответственно, вероятность обслуживания на каждом i-ом участке составляет $1-p_i$. Для того, чтобы вызов был обслужен на всех m транзитных участках, необходимо, чтобы он был обслужен на каждом из них. Следовательно, вероятность обслуживания на всем участке сети составит (для простоты без строгости рассуждений полагаем, что события на разных КУ независимы)

$$\prod_{i=1}^{m}(1-p_i).$$

Так как отказ в обслуживании и обслуживание на участке сети составляют полную группу несовместных событий, суммарная вероятность отказа на участке сети составит

$$p_{отк}=1-\prod_{i=1}^{m}(1-p_i).$$

Из теории телетрафика известно [7], что для снижения вероятности отказа необходимо увеличивать объем оборудования. Так как вероятности отказов (или, что то же самое, нормы потерь) на всех участках сети связи нормированы [6], сеть с коммутацией каналов при прочих равных условиях требует большего объема оборудования, чем сеть с коммутацией сообщений

или пакетов. Отсюда следует неэффективность коммутации каналов – для обеспечения значения $p_{\text{отк}}$, например, не выше 0,01 необходимо снижать величину p_i до 0,001 или меньше.

Кроме того, каналы связи в сети с коммутацией каналов используются крайне неэффективно. Например, при передаче голосового трафика в телефонной сети до 40% времени по каналу ничего не передается, так как в разговоре абонентов имеются паузы.

Рассмотрим теперь коммутацию пакетов. При таком способе коммутации данные передаются фрагментами фиксированного или переменного размера, называемыми пакетами. Для пакета всегда задается его максимальный размер – MTU (Maximum Transmission Unit), определяемый конкретным телекоммуникационным протоколом. В состав каждого пакета входит адресный заголовок, на основе анализа которого коммутационный узел определяет путь дальнейшей передачи. Таким образом, соединение в этом случае заранее не устанавливается (за исключением установления так называемого виртуального канала, о чем речь пойдет ниже).

В случае физической невозможности немедленной передачи пакета он ставится в очередь ожидания. Таким образом, в идеальном случае пакет не теряется, то есть потерь в явном виде в такой сети нет. Однако при загрузке сети пакеты будут задерживаться, то есть потери носят условный характер, и выражаются в задержке передачи. Поэтому сеть с коммутацией пакетов называют сетью с ожиданием. Соответственно, качество обслуживания в такой сети уже нельзя оценивать с использованием вероятности отказа. В связи с этим для этой цели используются два параметра – вероятность ожидания и среднее время ожидания [7].

Заметим, что здесь рассматривается идеализированный случай, то есть предполагается, что очередь может иметь бесконечный размер, и пакеты могут храниться в очереди бесконечно долго. На практике могут иметь место

явные потери, вызванные переполнением буфера ожидания, а также хранением пакета в буфере дольше определенного интервала времени.

Задержки передачи данных в сети с коммутацией пакетов можно разделить на два вида:

1. Алгоритмические задержки;

2. Случайные задержки.

Алгоритмические задержки обусловлены самим алгоритмом коммутации пакетов – каждый коммутационный узел буферизирует пакет, после чего производится анализ адресного заголовка и последующая передача. Понятно, что на это уходит определенное время, что и приводит к задержке.

Случайные задержки зависят от загрузки сети – чем больше загрузка, тем больше пакету приходится ждать в очереди.

Алгоритмические задержки не приводят к существенному ухудшению качества обслуживания, так как все пакеты, относящиеся к одному сообщению, задерживаются на одно и то же время. Случайные задержки ухудшают качество обслуживания, так как различные пакеты задерживаются на различные интервалы времени.

Весь трафик, передаваемый в современных сетях, можно разделить на три вида – видео, аудио и данные. Первые два вида представляют собой так называемый потоковый трафик, характеризующийся примерно равномерной скоростью передачи, высокой чувствительностью к вариации задержки (джиттеру) и некритичностью к потере отдельных пакетов. Данные, в свою очередь, характеризуются переменностью скорости передачи, нечувствительностью к джиттеру, а также критичностью к потере отдельных пакетов.

С учетом этого технология коммутации каналов лучше подходит для передачи потокового трафика – если соединение установлено, данные передаются с минимальными задержками. Именно этим объясняется тот факт, что современные цифровые телефонные сети используют канальную

коммутацию. Для передачи трафика данных лучше подходит пакетная коммутация.

У коммутации пакетов существуют две модификации:

1. Коммутация пакетов с установлением виртуального канала (логического соединения);

2. Коммутация пакетов без установления виртуального канала (дейтаграммный способ).

Для повышения качества обслуживания была разработана модификация способа коммутации каналов с использованием логического соединения. В этом случае передающая и принимающая сторона «договариваются» о предстоящем сеансе связи, то есть устанавливают так называемое логическое соединение. После установления соединения цепочка пакетов, относящаяся к одному сообщению, передается по единому пути, называемому виртуальным каналом. К протоколам, использующим логическое соединение, относятся транспортный протокол TCP, канальные протоколы Frame Relay, ATM. К протоколам, не поддерживающим логическое соединение, относится Интернет-протокол сетевого уровня IP. Следует отметить, что протокол TCP, более подробно рассматриваемый в главе 7, устанавливает логическое соединение «поверх» сетевого уровня, а пакеты сетевого уровня передаются без установления виртуального канала.

Как указывалось выше, инфокоммуникационные сети предполагает передачу информации именно с использованием IP-протокола. Так как дейтаграммный способ не дает никаких гарантий доставки информации, необходимо использовать дополнительные средства, повышающие надежность передачи данных.

1.2 Модель взаимодействия открытых систем ISO/OSI

Как указывалось выше, под телекоммуникационным протоколом понимается набор правил, в соответствии с которыми осуществляется информационное взаимодействие между удаленными объектами.

Традиционно телекоммуникационные протоколы организуются в стековую структуру, основой которой служит модель взаимодействия открытых систем – OSI (Open System Interconnection). Под открытой понимается система, функционирующая в соответствии со стандартизированными протоколами, чем обеспечивается корректное взаимодействие аппаратного и программного обеспечения различных производителей.

Рассмотрим сначала общий принцип организации стековой структуры протоколов. Стековая структура подразумевает наличие нескольких уровней, на которых располагаются телекоммуникационные протоколы. Протокол вышележащего уровня обращается к протоколу нижележащего уровня с некоторым запросом. Протокол нижележащего уровня предоставляет запрошенный набор услуг, скрывая от вышележащего протокола детали предоставления этих услуг (рисунок 1.2).

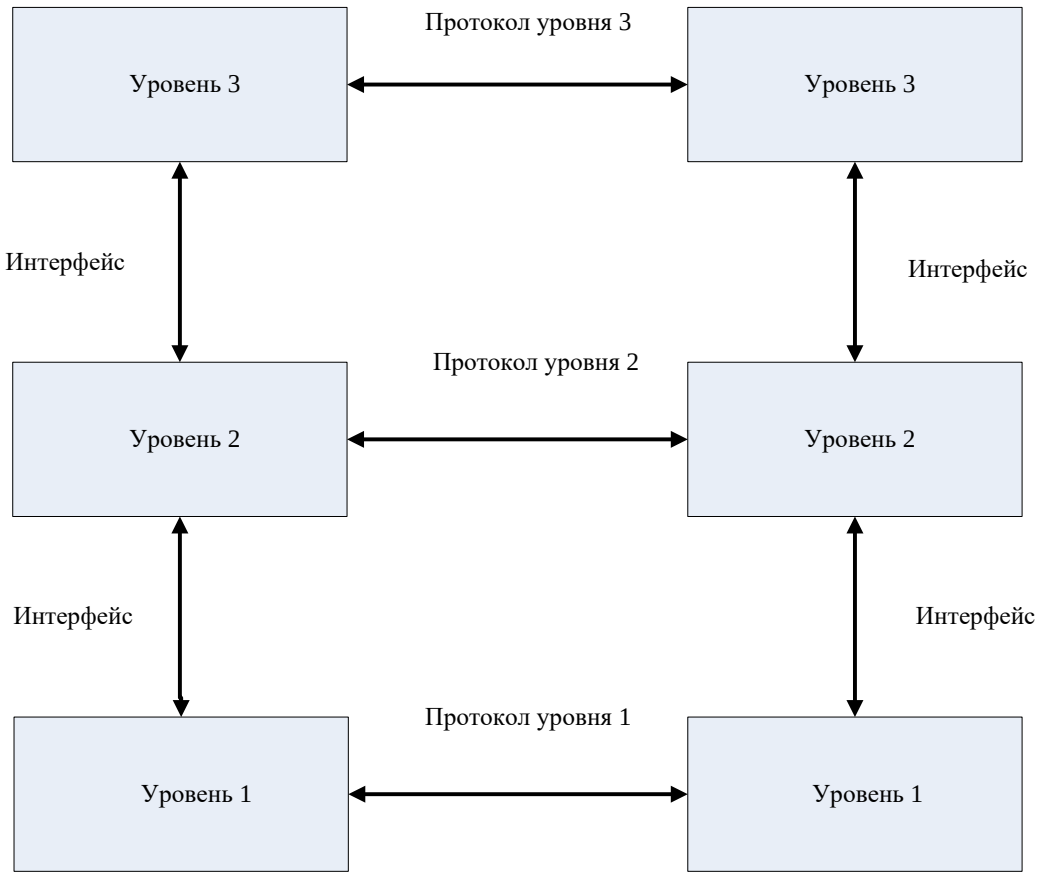

Рисунок 1.2 – Уровни протоколов в стеке

На рисунке 1.2 представлены для примера три уровня и две взаимодействующие открытые системы. Взаимодействие между одноименными уровнями происходит с использованием телекоммуникационных протоколов уровня (протокол уровня 1, протокол уровня 2 и т.д.). Между вышележащим и нижележащим уровнями расположен интерфейс, определяющий набор услуг, предоставляемых вышележащему уровню.

Услуга представляет собой набор примитивов, предоставляемых протоколу вышележащего уровня. С вышележащего уровня на нижележащий передается информационный блок, называемый элементом данных протокола – Protocol Data Unit (PDU). В PDU, помимо передаваемых данных, включаются служебные заголовки, необходимые для корректной обработки и передачи данных от одной открытой системы к другой.

В начале 80-х годов прошлого века была разработана эталонная модель взаимодействия открытых систем ISO/OSI, которая определяет семь уровней. Названия уровней и PDU, формируемых на каждом из уровней, представлены на рисунке 1.3.

Нумерация уровней традиционно ведется снизу вверх, то есть первым уровнем является физический, седьмым – прикладной.

На прикладном уровне располагаются протоколы, с которыми непосредственно взаимодействует пользователь или прикладное программное обеспечение. Так как прикладной уровень является самым верхним, протоколы прикладного уровня обеспечивают передачу информации от одной открытой системы к другой, используя для этого услуги нижележащих уровней. По интерфейсу между прикладным уровнем и уровнем представления передается PDU, называемый потоком. Различные прикладные протоколы в одной открытой системе могут формировать различные потоки.

Уровень представления отвечает за единый формат представления данных, передаваемых по сети.

Сеансовый уровень обеспечивает проведение сеанса между прикладными процессами двух или нескольких открытых систем.

Рассмотренные три верхних уровня являются сетенезависимыми, то есть они не зависят от тех сетевых технологий, на которых базируется инфокоммуникационная сеть. Четыре нижних уровня являются сетезависимыми, соответственно, будут рассмотрены более подробно.

Поток данных, сформированный на сетенезависимых уровнях, поступает на транспортный уровень. Основной функцией транспортного уровня является транспортировка сообщений и управление потоком информации от источника до получателя с обеспечением надежности доставки [5]. PDU, сформированный на транспортном уровне, получил название сегмента (Segment).

Протоколы транспортного уровня обеспечивают передачу сегмента с той степенью надежности, который требует прикладной протокол. Модель ISO/OSI определяет пять классов сервиса, предоставляемых транспортным уровнем. Выбор класса сервиса зависит от требований к надежности доставки сегментов, а также от надежности самой сети.

Сегмент, сформированный на транспортном уровне, поступает на сетевой уровень. PDU, формируемый на этом уровне, получил название пакета. Основной функцией сетевого уровня является обеспечение продвижения пакета через составную сеть, которая может состоять из множества сетей (часто называемых подсетями – Subnets), построенных на различных принципах (на различных технологиях канального уровня). Соответственно, сетевой уровень должен использовать адресацию, единую для всей составной сети. Соединение разнородных подсетей между собой обеспечивается маршрутизаторами – устройствами, обеспечивающими передачу пакетов из одной подсети в другую. Подробнее протоколы сетевого уровня будут рассмотрены в главе 4.

В свою очередь, подсети строятся на базе протоколов канального уровня. PDU, формируемый на канальном уровне, носит название кадра

(Frame). Основная задача канального уровня – обеспечение передачи кадров данных в пределах подсети.

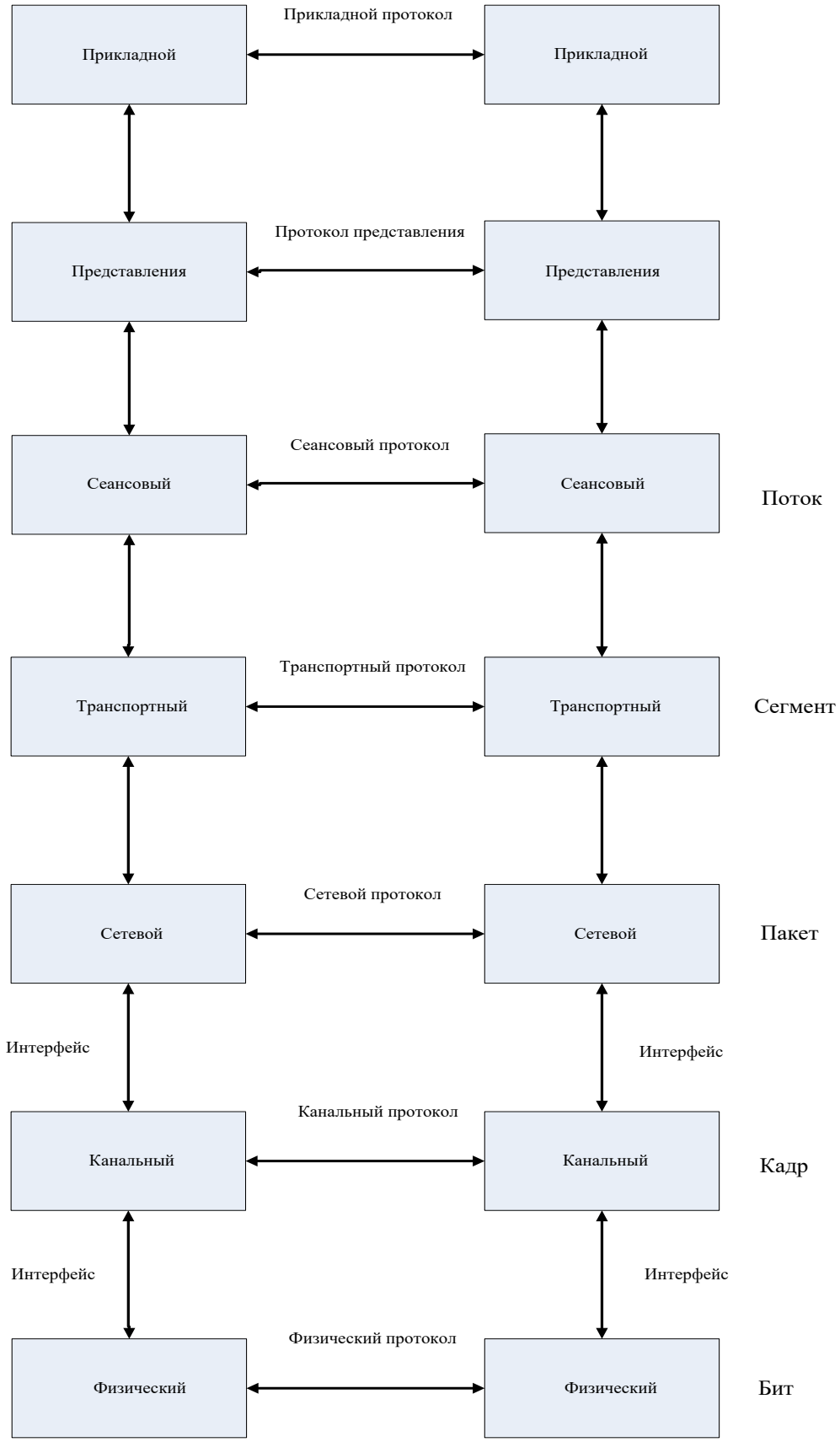

Рисунок 1.3 – Эталонная модель ISO/OSI

Наконец, на физическом уровне формируется поток битов, которые передаются с использованием той или иной среды. На физическом уровне определяются требования к физическому представлению передаваемых сигналов, физическим разъемам для соединения сетевых устройств между собой, и т.д.

Таким образом, при передаче данных между открытыми системами данные последовательно преобразуются в соответствии с протоколами различных уровней. Данные вышележащего уровня как бы «вкладываются» (инкапсулируются) в PDU нижележащего уровня [3 – 5]. При приеме данных ситуация обратная – PDU вышележащего уровня «извлекается» (декапсулируется) из PDU нижележащего уровня. Процесс инкапсуляции при передаче иллюстрируется рисунком 1.4.

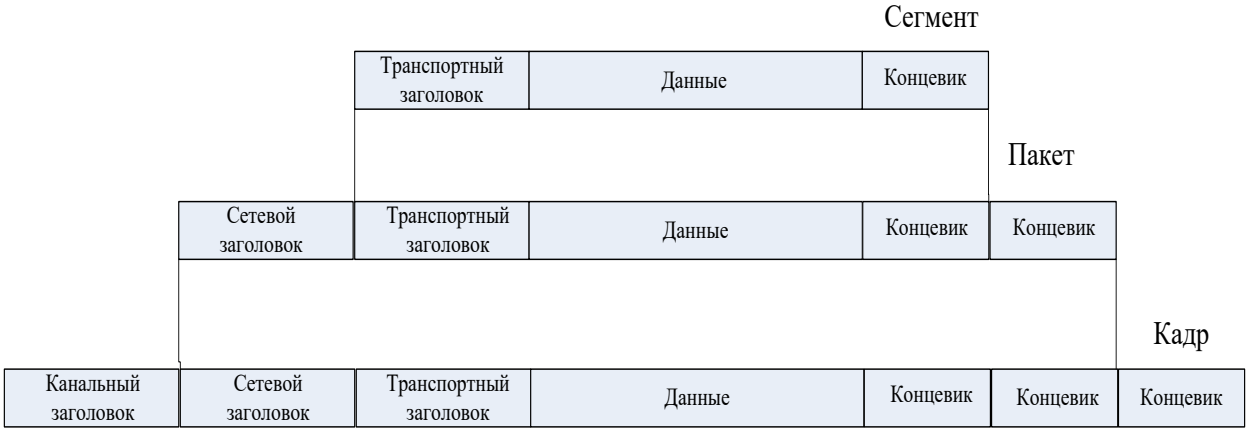

Рисунок 1.4 – Процесс инкапсуляции

Как видно из рисунка 1.4, служебные поля PDU на разных уровнях могут содержать не только заголовки, но и концевики, например, кадр Ethernet канального уровня имеет концевик в виде контрольной суммы. Протоколы каждого из уровней не воспринимают заголовков вышележащих уровней, считая их полем данных. Например, канальный протокол обеспечивает передачу кадров только на основе канального заголовка.

При приеме ситуация обратная – из принятого кадра декапсулируется пакет, из пакета – сегмент, данные сегмента передаются сетенезависимым уровням.

Промежуточные сетевые узлы декапсулируют принимаемую информацию только до «своего» уровня. Например, маршрутизатор, являясь устройством сетевого уровня, извлекает из принятого кадра пакет, определяет подсеть, через которую этот пакет нужно передать дальше, инкапсулирует его в кадр канальной технологии следующей подсети и передает. Подробнее функционирование маршрутизаторов будет рассмотрено в главе 4.

В заключение необходимо отметить, что рассмотренная семиуровневая структура является именно моделью, а не конкретным стеком телекоммуникационных протоколов. Наиболее распространенным стеком протоколов на сегодняшний день является стек TCP/IP, который рассмотрим более подробно в следующем параграфе.

1.3 Стек протоколов TCP/IP

Стек TCP/IP является далеко не единственным стеком телекоммуникационных протоколов, но остановимся мы более подробно именно на нем. Это вызвано, во-первых, тем, что де-факто в настоящее время на базе TCP/IP функционирует подавляющее большинство сетей, а во-вторых, только начинающие свое развитие сети NGN также используют этот стек.

Стек TCP/IP содержит не семь, а четыре уровня. Это объясняется тем, что некоторые уровни TCP/IP объединяют функции сразу нескольких уровней модели OSI. Например, прикладной уровень TCP/IP обеспечивает выполнение функций прикладного, представительского и сеансового уровней модели OSI. Кроме того, в TCP/IP нет некоторых уровней, присутствующих в модели OSI – канальный и физический уровни в нем не регламентируются. Это связано с необходимостью передачи IP-пакетов «поверх» любой канальной технологии и с использованием любой среды передачи.

Структура стека TCP/IP представлена на рисунке 1.5.

Рисунок 1.5 – Структура стека TCP/IP

На прикладном уровне располагаются программные средства двух видов – приложения (Applications) и службы (Services). Приложения организуют интерфейс между пользователем и сетью, службы готовят данные для дальнейшей передачи по сети.

Наиболее распространенными протоколами и службами прикладного уровня являются [8]:

- протоколы электронной почты (Simple Mail Transfer Protocol – SMTP, Post Office Protocol – POP, Internet Messaging Access Protocol – IMAP);

- протокол передачи гипертекстовой информации (Hypertext Transfer Protocol – HTTP);

- протокол передачи файлов (File Transfer Protocol – FTP) и простой протокол передачи файлов (Trivial FTP – TFTP);

- служба доменных имен (Domain Name System – DNS);

- протоколы удаленного доступа (Telnet, SSH), обеспечивающие виртуальное соединение с удаленными сетевыми устройствами;

- протокол динамического конфигурирования узлов (Dynamic Host Configuration Protocol – DHCP).

Различные протоколы прикладного уровня формируют поток данных, поступающий на основной (транспортный) уровень. Назначение этого уровня полностью соответствует назначению транспортного уровня модели OSI. На этом уровне функционирует два основных протокола – протокол управления передачей (Transfer Control Protocol – TCP) и протокол дейтаграмм пользователя (User Datagram Protocol – UDP). Основное различие между этими протоколами заключается в том, что TCP является протоколом, ориентированным на логическое соединение (connected-oriented), а UDP является протоколом без установления соединения (connectedless).

Оба протокола обеспечивают сегментацию данных при передаче и обратную процедуру при приеме. Кроме того, протоколы транспортного уровня обеспечивают мультиплексирование данных различных протоколов прикладного уровня. Это означает, что протокол TCP способен передавать данные различных приложений «поверх» одного логического соединения. Для идентификации протокола прикладного уровня используется условное 16-битовое число, называемое номером порта. Номер порта указывается в каждом передаваемом сегменте, в результате чего протокол транспортного уровня на принимающей стороне определяет приложение или службу, которой адресованы принятые данные.

Протокол TCP гарантирует доставку сегментов получателю путем реализации двух функций:

- подтверждение приема сегмента;
- контроль потока сегментов.

Для обеспечения надежности доставки сегментов протокол TCP на приемной стороне высылает подтверждение (ACK) о приеме сегмента. В случае отсутствия подтверждения сегмент отправляется повторно. Для контроля потока сегментов определяется так называемый размер окна (Window), который определяет количество байт, которое может принять получатель, не переполняя свои буферы.

Более детально механизмы обеспечения надежности доставки сегментов протоколом TCP будут рассмотрены в главе 7.

Сформированный на транспортном уровне сегмент поступает на уровень межсетевого взаимодействия, где инкапсулируется в пакет. Пакет содержит IP-адреса источника и получателя, а также некоторую другую управляющую информацию.

Важно, что протоколы транспортного уровня и протокол уровня межсетевого взаимодействия функционируют в тесной взаимосвязи, поэтому стек и получил название TCP/IP. Например, логическое соединение, устанавливаемое протоколом TCP, не может идентифицироваться только номером порта, так как в одной сети может быть несколько соединений с одинаковыми номерами портов. Поэтому соединение идентифицируется как номером порта, так и IP-адресом. Совокупность номера порта и IP-адреса получила название сокет (Socket).

Основная задача уровня межсетевого взаимодействия непосредственно следует из его названия – обеспечение продвижения пакетов по составной сети, состоящей из нескольких подсетей. На этом уровне располагаются два типа протоколов - маршрутизирующие протоколы и протоколы маршрутизации.

К маршрутизирующим относятся протоколы, обеспечивающие передачу IP-пакетов из одной подсети в другую на основе данных, содержащихся в таблицах маршрутизации. В настоящее время в качестве таких протоколов используются Интернет-протоколы четвертой и шестой версий (IPv4 и IPv6). К протоколам маршрутизации относятся протоколы, обеспечивающие наполнение таблиц маршрутизации путем обмена между маршрутизаторами служебной информацией. К таким протоколам относятся Routing Information Protocol (RIP), Open Shortest Path First (OSPF) и ряд других. Эти протоколы будут рассмотрены в главе 5.

Уровень сетевых интерфейсов, как было указано выше, в стеке TCP/IP не регламентируется. На этом уровне используются рекомендации,

описывающие правила инкапсуляции пакетов в кадры всех существующих на сегодняшний день технологий канального уровня. Задача в этом случае ставится так – IP-пакет должен иметь возможность продвижения через подсеть, построенную на базе любой существующей на сегодняшний день технологии канального уровня. Здесь будем рассматривать канальную технологию Ethernet как наиболее распространенную.

Соответственно, сначала необходимо рассмотреть принципы коммутации в технологии Ethernet. Этим принципам посвящена следующая глава.

2 Технологии канального уровня

2.1 Некоммутируемые сети Ethernet

Как известно, технология Ethernet изначально создавалась для построения локальных вычислительных сетей (ЛВС). Основным отличием ЛВС от современных сетей является разница в объемах передаваемой информации. В ЛВС оконечные устройства (компьютеры), как правило, отсылают более мощным компьютерам (серверам) запросы на выполнение некоторых действий вычислительного характера, а серверы, в свою очередь, передают по сети результаты вычислений. Очевидно, что объем передаваемых данных в таких сетях был невелик. Современные сети обеспечивают передачу так называемого мультимедийного трафика (совокупность аудио, видео и компьютерных данных), соответственно, повышаются требования к их производительности. Тем не менее, современные сети являются преемниками ЛВС, поэтому рассмотрение начнем именно с них.

Исторически первыми сетями Ethernet были некоммутируемые сети, использующие в качестве среды передачи коаксиальный кабель, неэкранированную «витую пару» или оптоволокно. Физические спецификации, использующие коаксиальный кабель (10 Base-5, 10 Base-2), были построены по топологии «общая шина», спецификации, использующие «витую пару» или оптоволокно (10 Base-T, 10 Base-F) – по звездообразной или иерархической топологии. Несмотря на это, с логической точки зрения, любая сеть «классического» Ethernet представляла собой «общую шину». Это объясняется тем, что в центре звездообразной топологии располагалось простейшее многопортовое сетевое устройство – концентратор (Hub). Концентратор, получив кадр данных на одном из своих портов, ретранслировал его на все остальные порты. Соответственно, кадр, адресуемый любому компьютеру, принимали все остальные компьютеры,

подключенные к данной сети, как это происходит в топологии «общая шина». Компьютеры с сетевыми картами, подключенные к сети, получили название рабочей станции (Workstation), этот термин и будем использовать далее при описании данной технологии.

Традиционно блок данных, передаваемых на канальном уровне, носит название «кадр» (Frame). Структура кадра зависит от конкретной реализации технологии Ethernet, однако для дальнейшего рассмотрения принципов коммутации достаточно указать поля, входящие во все типы кадров. Данные поля представлены на рисунке 2.1.

Преамбула	SFD	DA	SA	L	DSAP	SSAP	Control	Data	FCS
7 байт	10101011	6 байт	6 байт	2 байта	1 байт	1 байт	1 байт	46-1497 байт	4 байта

Рисунок 2.1 – Структура кадра

Преамбула кадра состоит из семи байт вида 10101010, необходимых для вхождения приемника в режим синхронизации. Начальный ограничитель кадра (Start of Frame Delimiter – SFD) – 10101011 вместе с преамбулой в итоге составляют 8 байт. Далее следуют физические адреса узла назначения (DA – Destination Address) и узла источника (SA – Source Address). В технологиях Ethernet физические адреса получили название MAC-адресов. Они содержат 48 двоичных разрядов (6 байт) и представляются в шестнадцатеричной системе. В локальных сетях адресация узлов производится на основе MAC-адресов, которые "прошиты" в ПЗУ сетевых карт.

Адрес, состоящий из всех единиц FFFFFFFFFFFF, является широковещательным адресом (broadcast), когда передаваемая в кадре информация предназначена всем узлам локальной сети.

Младшие 24 разряда MAC-адреса (6 шестнадцатеричных разрядов) задают уникальный номер оборудования, например, номер сетевой карты. Следующие 22 разряда задают идентификатор производителя оборудования. При передаче кадра старший бит, равный 0, указывает на то, что адрес является индивидуальным, а равный 1 – на то, что адрес является групповым. Второй старший бит, равный 0, указывает, что идентификатор задан централизованно комитетом IEEE. В стандартной аппаратуре Ethernet идентификатор всегда задан централизованно. Несмотря на то, что в MAC-адресе выделена старшая и младшая части, MAC-адрес считается плоским (flat).

Поле L определяет длину поля данных Data, которое может быть от 46 до 1497 байт (в информационных кадрах процедуры LLC2 – до 1496 байт, поскольку поле Control – 2 байта). Если поле данных меньше 46 байт, то оно дополняется до 46 байт.

Поле контрольной суммы (FCS – Frame Check Sequence) длиной в 4 байта позволяет определить наличие ошибок в полученном кадре за счет использования алгоритма проверки на основе циклического кода.

Так как кадр, передаваемый по некоммутируемой сети Ethernet, принимают все рабочие станции, каждая из них определяет, ей или не ей предназначен кадр, сравнивая собственный MAC-адрес с адресом, указанным в поле DA кадра. Если адреса совпадают, рабочая станция копирует кадр в свой буфер и обрабатывает, в противном случае кадр просто игнорируется.

При таком способе все станции используют общую разделяемую среду передаче данных. Поэтому при одновременной передаче двумя или более станциями своих кадров сигналы в общей разделяемой среде накладываются друг на друга и искажаются. Такое событие получило название коллизии. Очевидно, что возникновение коллизий приводит к ошибкам в передаваемых данных.

Для предотвращения ошибок, вызванных коллизиями, вводится понятие метода доступа.

Под методом доступа понимается набор правил, регламентирующих передачу данных рабочими станциями по общей разделяемой среде. При этом передача должна быть организована таким образом, чтобы кадры разных станций не «сталкивались» в общей среде (не возникали, или, по крайней мере, гарантированно обнаруживались коллизии).

Различают два вида методов доступа – детерминированные и вероятностные.

При детерминированном методе доступа каждой рабочей станции предоставляется для передачи свой временной интервал. Такой метод при всплесковом трафике обладает существенным недостатком – заранее неизвестно, каким рабочим станциям необходимо будет передавать данные, и каков объем этих данных. Поэтому в технологии Ethernet используется вероятностный метод, называемый методом коллективного доступа с контролем несущей и обнаружением коллизий (CSMA/CD – Carrier Sence Multiply Access with Collision Detection).

Суть метода заключается в том, что рабочая станция, которой необходимо передать данные, сначала «прослушивает» общую разделяемую среду. Если по общей среде в этот момент передаются данные какой-либо другой станцией, передача откладывается. В противном случае начинается передача с одновременным контролем общей разделяемой среды. Если в процессе передачи обнаруживается коллизия, в сеть выдается сигнал специального вида (в технологии Ethernet она называется jam-последовательностью), приняв который любая станция немедленно прекращает свою передачу. Выдержав случайный временной интервал, станция может вновь попытаться начать передачу.

Сеть, построенная на общей разделяемой среде, образует так называемый домен коллизий. Кратко рассмотрим работу простейшего домена коллизий, построенного на базе физической спецификации 10Base-T (рисунок 2.2).

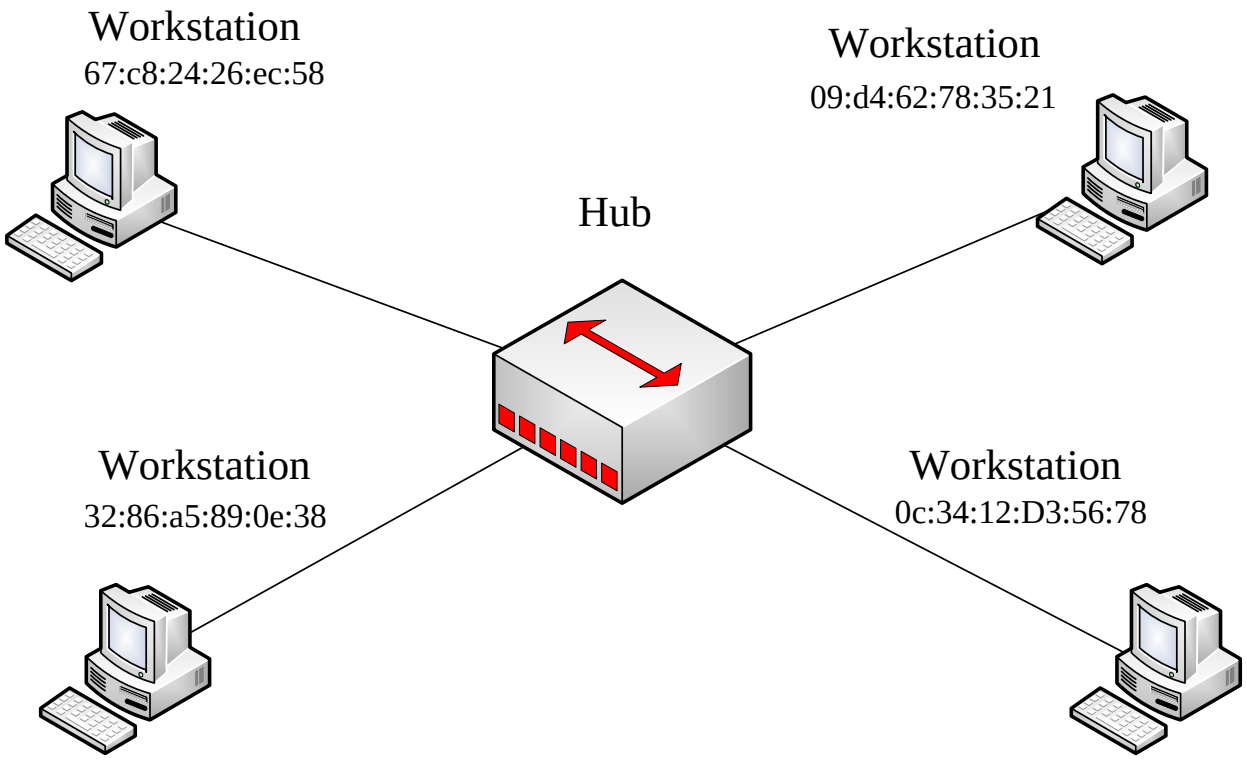

Рисунок 2.2 – Пример сети

В основе сети используется концентратор (Hub), основной функцией которого является ретрансляция кадра, пришедшего на один из портов, на все остальные порты. Таким образом, с логической точки зрения сеть на концентраторе представляет собой общую шину, несмотря на то, что с физической точки зрения используется звездообразная топология.

Кадр, переданный любой рабочей станцией, принимается всеми остальными станциями сети. Каждая из станций сравнивает свой MAC-адрес с MAC-адресом, указанным в поле DA принятого кадра. Очевидно, что при их совпадении кадр предназначен именно этой станции, поэтому она копирует его в свой буфер и обрабатывает. Остальные станции данный кадр игнорируют.

Для построения сети с большим числом узлов несколько концентраторов соединяют между собой, однако максимальное число концентраторов между двумя любыми компьютерами не должно быть

больше 4. Требования к сети определяются правилом 5-4-3, в котором 5 – максимальное число сегментов сети, 4 – максимальное число концентраторов между любыми хостами, 3 – хосты могут быть только в трех сегментах. При этом диаметр сети может существенно увеличиться. Структура сети должна быть древовидной, петлевые соединения запрещены [3].

Основное достоинство такой сети – простота, что, в свою очередь, приводит к невысокой стоимости сетевого оборудования (в частности, сетевых адаптеров и концентраторов). В то время, когда создавался «классический» Ethernet, считалось, что пропускной способности 10 Мбит/с с использованием общей разделяемой среды более чем достаточно. Со временем появились приложения с интенсивным информационным обменом между рабочими станциями и (или) серверами, что привело к созданию технологии Fast Ethernet. Однако проблема в корне не решилась, так как некоммутируемая сеть обладает существенным недостатком, суть которого поясняют рисунки 2.3 и 2.4.

Введем в рассмотрение понятие коэффициента загрузки сети как отношение скорости передачи информации в сети к максимальной пропускной способности протокола:

$$\rho = \frac{c}{c_{\max}}.$$

Например, если в сети 100Base-T (Fast Ethernet) данные передаются со скоростью 20 Мбит/с, коэффициент загрузки равен 0,2. Очевидно, что максимальное значение коэффициента – единица.

На рисунке 2.3 представлена зависимость скорости передачи информации в реальной и идеальной некоммутируемых сетях. В идеальной сети при увеличении коэффициента загрузки скорость линейно увеличивается до своего максимального значения. В реальной сети, начиная с некоторого порогового значения, скорость передачи резко снижается вплоть до нуля.

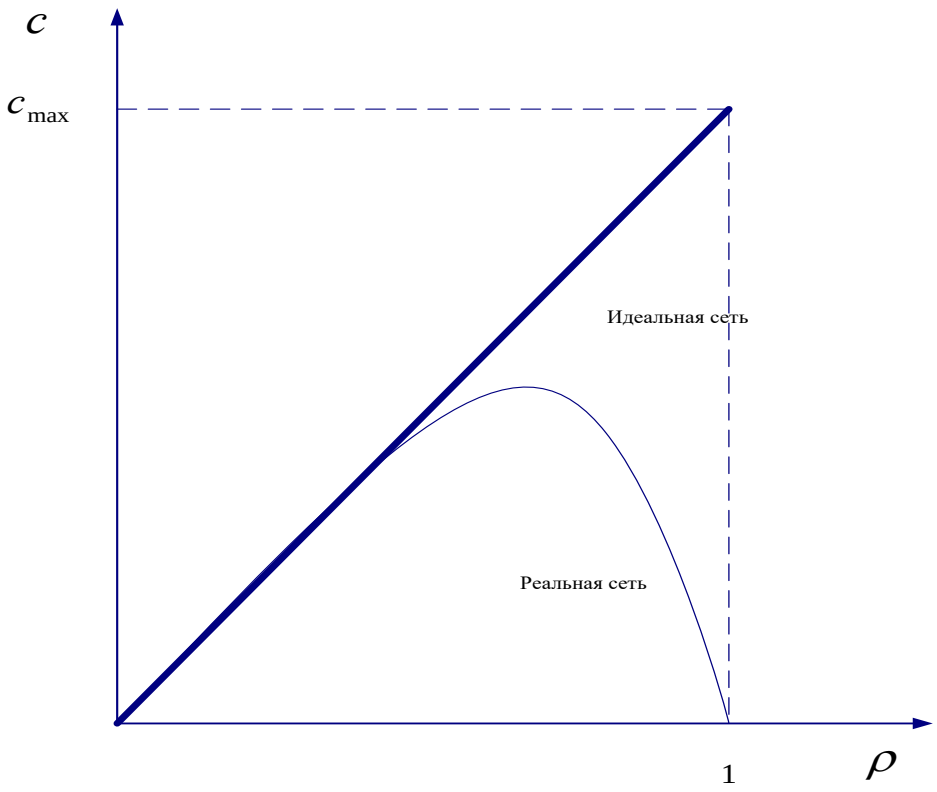

Рисунок 2.3 – Зависимость скорости передачи от коэффициента загрузки

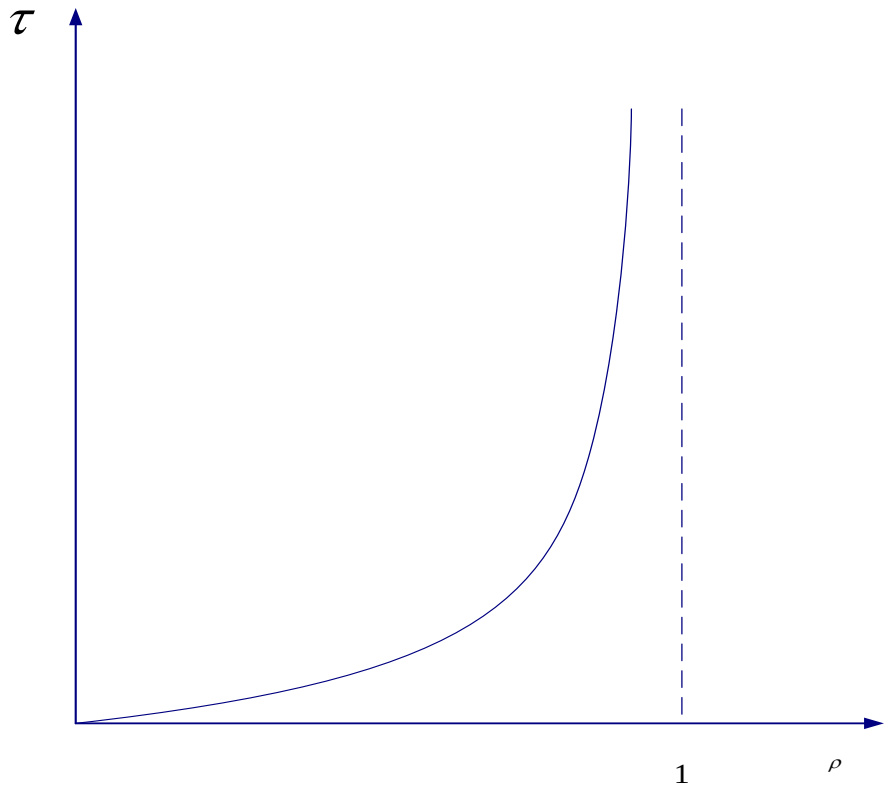

Рисунок 2.4 – Зависимость задержки передачи от коэффициента загрузки

Это происходит за счет того, что при существенной загрузке возникают множественные коллизии, сеть начинает их отрабатывать, рабочие станции прерывают передачу, опять пытаются получить доступ к общей разделяемой среде, что приводит к новым коллизиям, и т.д. Сеть фактически начинает работать сама на себя, не передавая пользовательские данные.

Соответственно, начинают увеличиваться и задержки передачи кадров. Зависимость этой задержки от коэффициента загрузки представлена на рисунке 2.4.

Из данных рисунков следует, что домен коллизий можно загружать только до определенного предела, для Ethernet этот предел составляет 0,4-0,5 [3]. При этом коэффициент загрузки в современных сетях, как правило, зависит не столько от количества рабочих станций, подключенных к общей разделяемой среде, сколько от параметров трафика. Если в сети функционируют приложения, требующие интенсивного информационного обмена, то и небольшое количество станций могут существенно загрузить сеть.

В результате были разработаны высокоскоростные Ethernet-технологии – Fast Ethernet, Gigabit Ethernet и т.д. Эти технологии во многом сохранили преемственность по отношению к «классическому» Ethernet с одновременным увеличением пропускной способности. Однако главное, что позволяет в современных сетях доступа избежать множественных коллизий, – это переход к коммутируемым сетям Ethernet.

Исторически первыми простейшими коммутирующими устройствами стали мосты – устройства, способные делить сеть на несколько логических сегментов. Рассмотрим принципы работы мостов и деление сети на сегменты (сегментацию) в следующем параграфе.

2.2 Сегментация сетей

Под сегментацией сети понимается ее деление на два или несколько логических сегментов, в каждом из которых локализуется внутрисегментный трафик.

Рассмотрим сеть, представленную на рисунке 2.5.

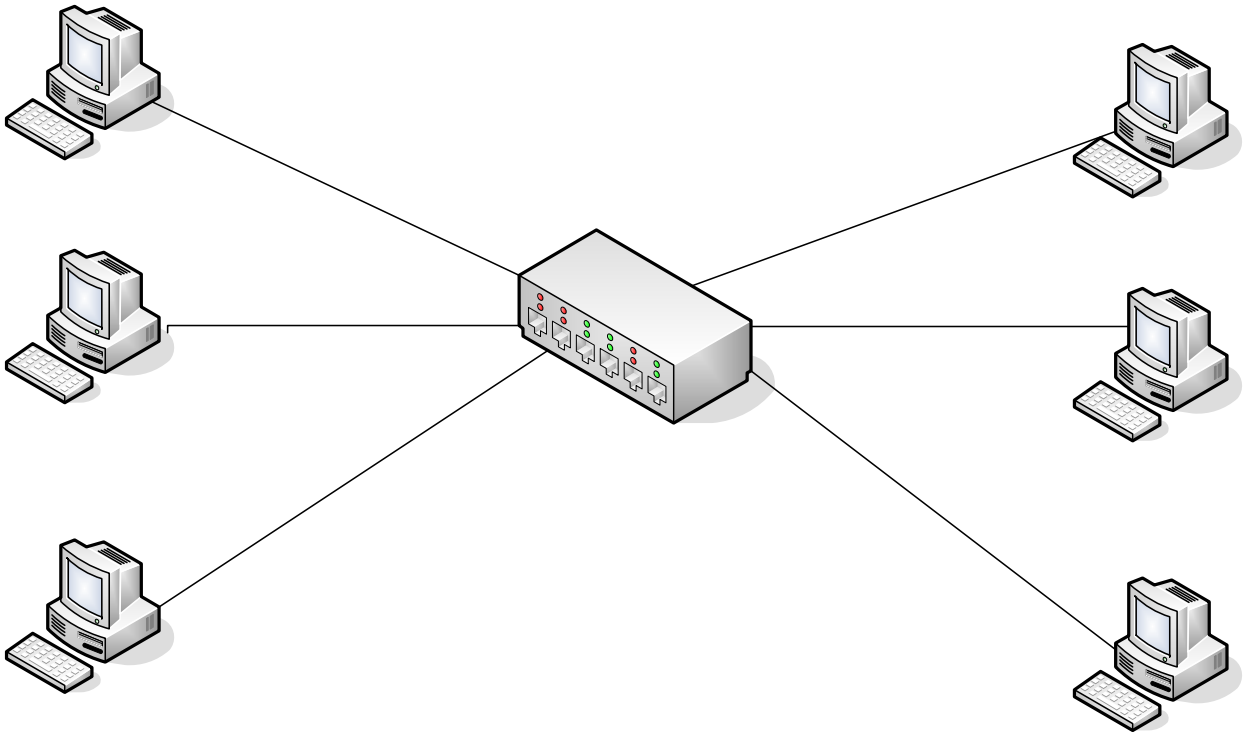

Рисунок 2.5 – Пример сети с концентратором

На рисунке представлена сеть на основе концентратора. Условно назовем рабочие станции, расположенные на рисунке слева от концентратора, логическим сегментом 1 (ЛС1), справа – логическим сегментом 2 (ЛС 2). Тогда суммарный трафик сети можно представить в виде

$$C = C_1 + C_{1-2} + C_2,$$

где C - суммарный трафик;

C_1 - трафик между рабочими станциями ЛС 1 (внутрисегментный трафик первого сегмента);

C_2 - трафик между рабочими станциями ЛС 2 (внутрисегментный трафик второго сегмента);

C_{1-2} - трафик между рабочими станциями разных сегментов (межсегментный трафик).

Очевидно, что при использовании концентратора (общей разделяемой среды передачи данных) все рабочие станции будут принимать все передаваемые кадры независимо от того, кому они предназначены. Если использовать вместо концентратора устройство, способное локализовать внутрисегментный трафик (то есть передавать кадры из одного сегмента в другой только в том случае, если получатель находится в другом сегменте), то трафик ЛС 1 составит

$$C = C_1 + C_{1-2},$$

а ЛС 2

$$C = C_2 + C_{1-2}.$$

Соответственно, загрузка каждого из сегментов уменьшится.

Устройство, способное производить сегментацию сети, называется мостом (Bridge). Принимая кадр данных, мост производит одну из следующих операций:

- продвижение – передача кадра из буфера входного порта в буфер выходного порта, после получения доступа к сегменту – передача кадра в сеть;

- фильтрация – стирание кадра из буфера входного порта.

Как следует из перечисленных операций, мост является простейшим коммутирующим устройством. Решение о том, какую из операций необходимо осуществить, мост принимает, обращаясь к находящейся в его памяти адресной таблице. Адресная таблица соотносит MAC-адреса рабочих станций с номерами сегментов, в которых они находятся.

Рассмотрим пример, представленный на рисунке 2.6.

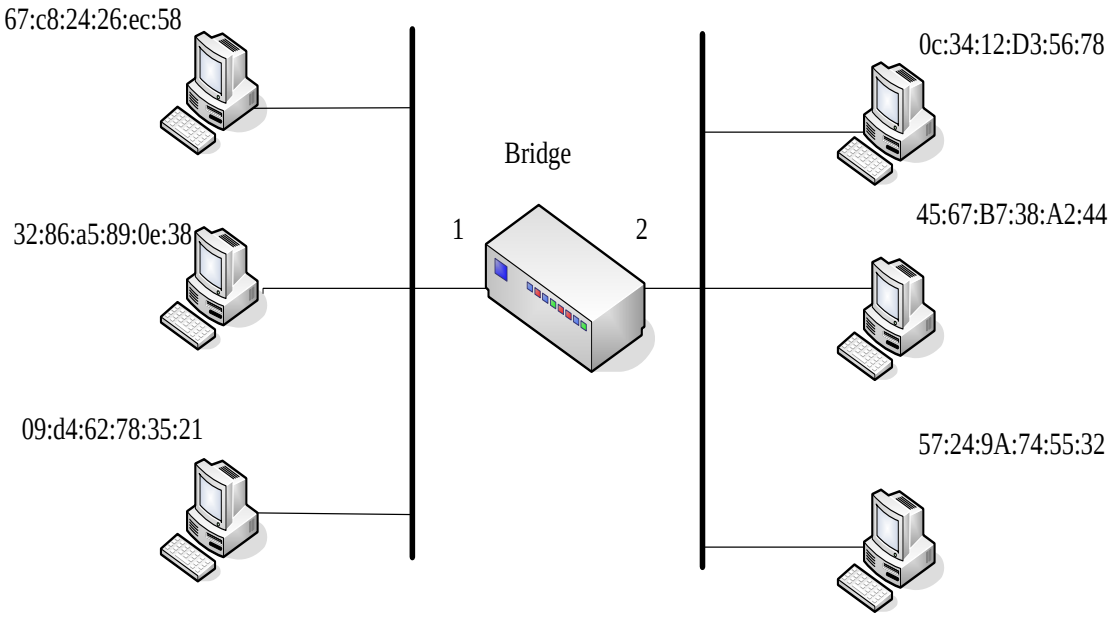

Рисунок 2.6 – Пример сети с мостом

На рисунке представлен двухпортовый мост, соединяющий между собой два сегмента сети, построенных, например, на концентраторах. Также на рисунке показаны MAC-адреса рабочих станций, выбранных в данном примере произвольно. Адресная таблица такого моста будет иметь вид, показанный в таблице 2.1.

Таблица 2.1 – Адресная таблица моста

Адрес	№ сегмента (порта)
67:c8:24:26:ec:58	1
32:86:a5:89:0e:38	1
09:d4:62:78:35:21	1
0c:34:12:D3:56:78	2
45:67:B7:38:A2:44	2
57:24:9A:74:55:32	2

В случае, если например мост принимает на первом порту (или, что тоже самое, из ЛС 1) кадр с адресом назначения DA=45:67:B7:38:A2:44, то

после буферизации этого кадра происходит обращение к адресной таблице и отыскивается строка с этим адресом. После нахождения адреса определяется, к какому сегменту (или, то же самое, порту) относится этот адрес. В данном примере адрес относится к порту 2, следовательно, кадр необходимо продвинуть. Соответственно, кадр передается в буфер порта 2, и после получения доступа к разделяемой среде второго сегмента в соответствии с методом CSMA/CD, передается в сеть.

Аналогично, если на первый порт приходит кадр с адресом DA=67:c8:24:26:ec:58, он просто стирается из буфера первого порта.

Из сказанного выше можно сделать вывод о том, что для своего успешного функционирования адресная таблица моста должна быть заполнена. Существуют два типа записей в адресной таблице [3]:

- статические записи, не имеющие срока жизни;

- динамические записи, имеющие срок жизни.

Статические записи вносятся в таблицу вручную администратором сети с использованием, например, консольного порта (об этом подробнее будет сказано ниже). То, что статическая запись не имеет срока жизни, означает, что если от источника с соответствующим MAC-адресом не будут поступать кадры, запись все равно остается актуальной.

Использование статических записей на практике имеет большую трудоемкость – при большом объеме адресной таблицы сложно уследить за актуальностью записей. Очевидно, что в этом случае при удалении рабочей станции из сети, добавления в сеть новой станции, перемещения станции из одного сегмента в другой необходима корректировка адресной таблицы. Поэтому статические записи используются ограниченно.

Динамическими являются записи, внесенные мостом в свою память самостоятельно, в процессе так называемого обучения. Процесс обучения заключается в исследовании принимаемых кадров на предмет MAC-адреса источника (SA), и соотнесения этого адреса с номером порта, на который кадр был принят.

Например (рисунок 2.6), адресная таблица моста пуста, и на первый порт принимается кадр с SA=32:86:a5:89:0e:38. Чаще всего на практике это оказывается широковещательный кадр DA=ff::ff, соответственно, он продвигается во второй сегмент. При этом мост заносит соответствующую запись в таблицу, которая указывает, что рабочая станция с адресом 32:86:a5:89:0e:38 относится к ЛС 1. Аналогично вносятся и остальные записи по мере приема кадров от других рабочих станций.

Мосты, работающие по данному алгоритму, называют «прозрачными» мостами, так как сами рабочие станции никакого участия в построении адресной таблицы не принимают – они просто отправляют и принимают кадры таким же образом, как и в сети с общей разделяемой средой, то есть работа моста для рабочих станций «прозрачна».

Процесс обучения моста никогда не заканчивается. Например, если перенести рабочую станцию с адресом 32:86:a5:89:0e:38 из ЛС 1 в ЛС 2, то после приема первого же кадра от этой станции соответствующая строка адресной таблицы изменится – будет указано, что станция относится ко второму порту. Кроме того, динамические записи имеют срок жизни. Это означает, что если какую-либо рабочую станцию удалить из сети, и от нее определенное время (обычно 300 с) не будут поступать кадры, соответствующая запись будет удалена из таблицы.

2.3 Коммутаторы Ethernet

Коммутатором (Switch) называется многопортовый мост, способный разделить сеть на несколько сегментов. Каждый порт коммутатора имеет собственный процессор и буферную память для хранения кадров. Работа по фильтрации и продвижению кадров коммутатором производится так же, как и мостами.

Каждый сегмент, образованный портом (интерфейсом) коммутатора с присоединенной к нему рабочей станцией (компьютером) с несколькими станциями, является доменом коллизий. При возникновении коллизии в сети,

реализованной на концентраторе, сигнал коллизии распространяется по всем портам концентратора. Однако на другие порты коммутатора сигнал коллизии не передается.

Существует два режима двусторонней связи: полудуплексный (half-duplex) и полнодуплексный (full-duplex). В полудуплексном режиме в любой момент времени одна станция может либо вести передачу, либо принимать данные. В полнодуплексном режиме станция может одновременно принимать и передавать информацию, т. е. обе станции в соединении "точка-точка" могут передавать данные в любое время, независимо от того, передает ли другая станция. Для разделяемой среды полудуплексный режим является обязательным. Ранее создававшиеся сети Ethernet на коаксиальном кабеле были только полудуплексными. Неэкранированная витая пара UTP и оптическое волокно могут использоваться в сетях, работающих в обоих режимах. Высокоскоростные сети (10-Gigabit Ethernet и выше) работают только в полнодуплексном режиме. Большинство коммутаторов могут использовать как полудуплексный, так и полнодуплексный режим.

В случае присоединения рабочих станций (хостов) индивидуальными линиями к портам коммутатора каждая станция вместе с портом образует микросегмент. Такое соединение рабочих станций с коммутатором получил название «микросегментация».

При микросегментации коллизия может возникнуть только в пределах микросегмента. А если учесть, что практически все современные коммутаторы поддерживают полнодуплексный режим работы (Full Duplex), при котором передача данных в различных направлениях производится независимо, коллизии исключаются.

В результате этого современные сети уже практически не используют концентраторов и мостов, а строятся на коммутаторах.

При всем разнообразии современных коммутаторов они строятся на базе трех типовых структур, соответственно, различают три типа коммутаторов:

- коммутаторы на основе коммутационной матрицы;

- коммутаторы на основе общей шины;

- коммутаторы на основе разделяемой памяти.

Коммутационная матрица обеспечивает основной и самый быстрый способ взаимодействия процессоров портов, именно он был реализован в первом промышленном коммутаторе локальных сетей. Однако реализация матрицы возможна только для определенного числа портов, причем сложность схемы возрастает пропорционально квадрату количества портов коммутатора. Структура такого коммутатора представлена на рисунке 2.7, заимствованном из [3].

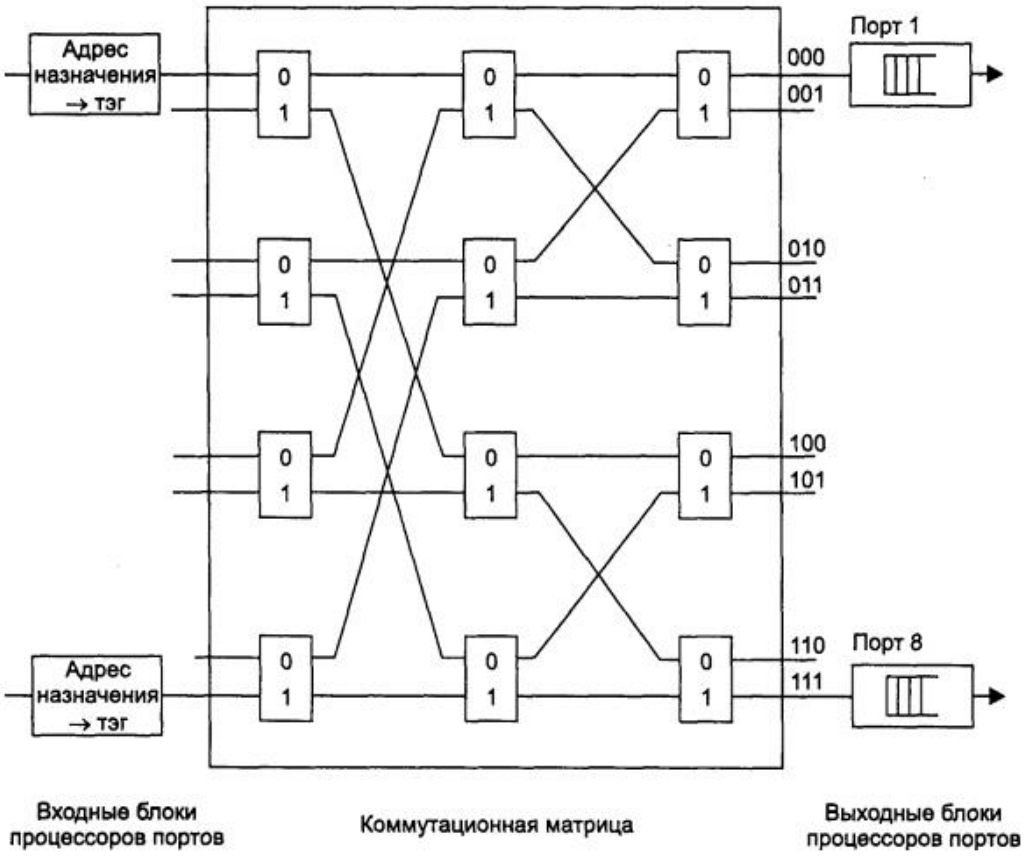

Рисунок 2.7 – Коммутатор на основе коммутационной матрицы

Матрица состоит из нескольких звеньев двоичных переключателей. После того, как процессор входного порта, обратившись к адресной таблице, определит номер выходного порта, к кадру добавляется так называемый тэг

(tag). Тэг представляет собой n-разрядное двоичное число, где n - количество звеньев. Тэг однозначно определяет номер выхода и одновременно соединительный путь через матрицу.

Например, если необходимо передать кадр с порта 0 на порт 8, тэг для представленной схемы будет иметь вид 111. После получения кадра на выходе матрицы и записи его в буфер выходного порта тэг удаляется, и кадр передается в сеть.

Необходимо отметить, что сама коммутационная матрица работает не в режиме коммутации пакетов, а в режиме коммутации каналов. Это означает, что если нужный путь через матрицу занят, наступает отказ в обслуживании. Однако кадр в этом случае будет храниться в буфере входного порта, и будет передан по мере освобождения матрицы, поэтому коммутатор в целом функционирует в режиме коммутации пакетов.

В коммутаторах с общей шиной процессоры портов связывают высокоскоростной шиной, используемой в режиме разделения времени. Структура такого коммутатора представлена на рисунке 2.8, взятом там же [3].

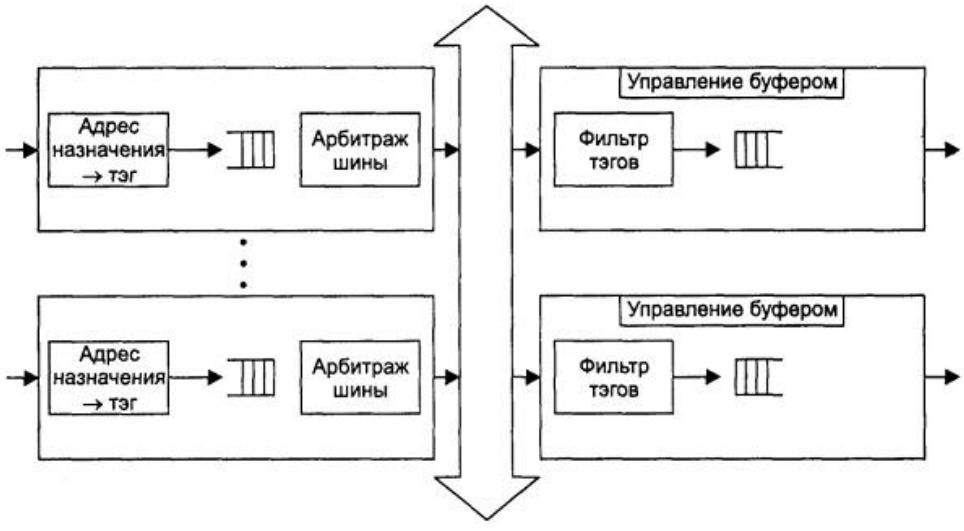

Рисунок 2.8 – Коммутатор на основе общей шины

В таком коммутаторе к каждому кадру также добавляется тэг, в качестве которого используется номер выходного порта. Так как все порты

подключены к общей шине, они принимают передаваемый по ней кадр, но буферизирует и затем передает его в сеть только тот порт, чей номер совпадает с тэгом.

У коммутатора на основе общей шины имеются две особенности. Во-первых, для исключения блокировок производительность общей шины должна быть, как минимум, равна сумме производительностей всех портов. Во-вторых, даже при выполнении этого условия при продвижении кадров могут возникать дополнительные задержки. Это связано с тем, что кадр должен храниться в буфере входного порта до тех пор, пока шина не освободится, то есть не передаст до конца предыдущий кадр. Для минимизации этой задержки шина работает в псевдопараллельном режиме, то есть кадры дополнительно фрагментируются, и шина поочередно передает фрагменты с разных портов.

Структура коммутатора с разделяемой памятью представлена на рисунке 2.9 [3].

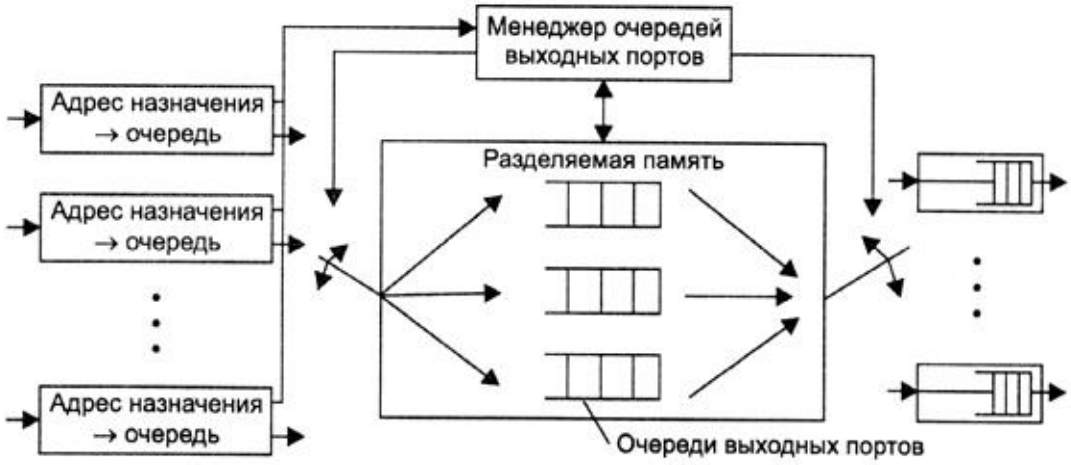

Рисунок 2.9 – Коммутатор с разделяемой памятью

Входные блоки процессоров портов соединяются с переключаемым входом разделяемой памяти, а выходные блоки этих же процессоров соединяются с переключаемым выходом этой памяти. Переключением входа и выхода разделяемой памяти управляет менеджер очередей выходных

портов. В разделяемой памяти менеджер организует несколько очередей данных, по одной для каждого выходного порта. Входные блоки процессоров передают менеджеру портов запросы на запись данных в очередь того порта, который соответствует адресу назначения кадра. Менеджер по очереди подключает вход памяти к одному из входных блоков процессоров и тот переписывает часть данных кадра в очередь определенного выходного порта. По мере заполнения очередей менеджер производит также поочередное подключение выхода разделяемой памяти к выходным блокам процессоров портов, и данные из очереди переписываются в выходной буфер процессора.

У каждой из описанных архитектур есть свои преимущества и недостатки, поэтому часто в сложных коммутаторах эти архитектуры применяются в комбинации друг с другом. Пример такого комбинирования приведен на рисунке 2.10 [3].

Рисунок 2.10 – Коммутатор с комбинированной архитектурой

Рассмотрим логику работы коммутатора Ethernet на примере простейшей сети, представленной на рисунке 2.11.

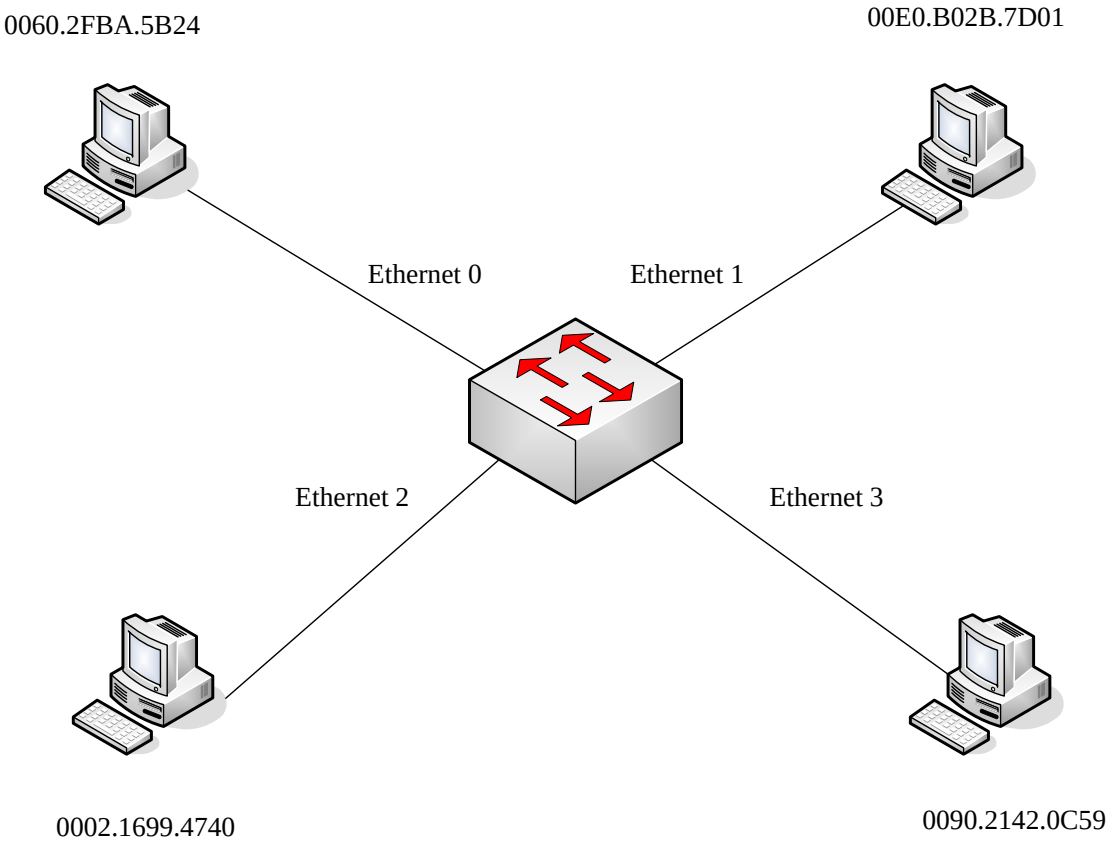

Рисунок 2.11 – Сеть с коммутатором

На рисунке представлен коммутатор, к портам которого подключены четыре рабочие станции. Для определенности у каждой станции указан ее MAC-адрес (адреса взяты произвольно и представлены в шестнадцатеричной форме). Порты коммутатора адресов не имеют, но имеют номера, которые, как правило, включают в себя название канальной технологии и порядковый номер (например, Ethernet 0, FastEthernet 0 и т.д.).

В отличие от концентратора, при поступлении на один из портов коммутатора кадра, он должен передать его не на все свои задействованные порты, а только на порт, к которому подключен получатель. Решение о том, на какой из портов необходимо передать кадр, коммутатор принимает аналогично мосту, анализируя адрес назначения (он указан в поле DA кадра, рисунок 2.1). Для этого кадр сначала записывается в буфер входного порта (буферизируется).

Для принятия решения коммутатор обращается к хранящейся в его памяти адресной таблице, которая связывает номера портов с адресами подключенных к ним рабочих станций (подобно адресной таблице моста). Упрощенная адресная таблица коммутатора для нашего примера имеет вид, представленный в таблице 2.2.

Таблица 2.2 – Упрощенный вид адресной таблицы

Порт	Адрес рабочей станции
Ethernet 0	0060.2FBA.5B24
Ethernet 1	00E0.B02B.7D01
Ethernet 2	0002.1699.4740
Ethernet 3	0090.2142.0C59

Реальные адресные таблицы содержат еще ряд записей, о которых будет сказано ниже.

2.4 Характеристики коммутаторов Ethernet

Быстродействие или производительность коммутатора определяются рядом параметров: скоростью фильтрации кадров, скоростью продвижения кадров, пропускной способностью, длительностью задержки передачи кадра [3].

Скорость фильтрации определяется временем приема кадра, запоминанием его в буфере, обращением к адресной таблице коммутации и удалением кадра из буферной памяти, если адресат и источник находятся в одном сегменте. Коммутатор обычно успевает фильтровать кадры в темпе их поступления в интерфейс, поэтому фильтрация не вносит дополнительной задержки.

Скорость продвижения кадров определяется временем приема кадра, запоминанием его в буфере, обращением к адресной таблице и передачей кадра с входного порта на выходной, который связан с устройством

назначения. Скорость фильтрации и скорость продвижения задаются в кадрах в секунду, причем для оценки этих параметров обычно берутся кадры минимальной длины 64 байта.

Пропускная способность коммутатора определяется количеством передаваемых данных, содержащихся в поле «Data» кадра в единицу времени и измеряется в битах в секунду. Пропускная способность достигает своего максимального значения при передаче кадров максимальной длины.

Задержка передачи кадров определяется временем от момента появления первого байта кадра на входном порте коммутатора до момента появления этого байта на выходном порте. В зависимости от режима коммутации время задержки составляет от единиц до сотен микросекунд.

Коммутаторы могут работать в нескольких режимах, при изменении которых меняются задержка и надежность. Для обеспечения максимального быстродействия коммутатор может начинать передачу кадра сразу, как только получит MAC-адрес узла назначения. Такой режим получил название сквозной коммутации или коммутации "на лету" (cut-through switching), он обеспечивает наименьшую задержку при прохождении кадров через коммутатор. Однако в этом режиме невозможен контроль ошибок, поскольку поле контрольной суммы находится в конце кадра. Следовательно, этот режим характеризуется низкой надежностью.

Во втором режиме коммутатор получает кадр целиком, помещает его в буфер, проверяет поле контрольной суммы (FCS) и затем пересылает адресату. Если получен кадр с ошибками, то он отбрасывается (discarded) коммутатором. Поскольку кадр перед отправкой адресату назначения запоминается в буферной памяти, такой режим коммутации получил название коммутации с промежуточным хранением или буферизацией (store-and-forward switching). Таким образом, в этом режиме обеспечивается высокая надежность, но низкая скорость коммутации.

Промежуточное положение между сквозной коммутацией "на лету" и буферизацией занимает режим коммутации свободного фрагмента (fragment-

free mode). В этом режиме читаются первые 64 байта, которые включают заголовок кадра и поле данных минимальной длины. После этого начинается передача кадра до того, как будет получен и прочитан весь кадр целиком. При этом производится верификация адресации и информации LLC-протокола, чтобы убедиться, что данные будут правильно обработаны и доставлены адресату.

Когда используется режим сквозной коммутации "на лету", порты устройств источника и назначения должны иметь одинаковую скорость передачи. Такой режим называется симметричной коммутацией. Если скорости не одинаковы, то кадр должен запоминаться (буферизироваться) перед тем, как будет передаваться с другой скоростью. Такой режим называется асимметричной коммутацией, при этом должен применяться режим с буферизацией.

Асимметричная коммутация обеспечивает связь между портами с разной полосой пропускания. Данный режим является характерным, например, для потока данных между многими клиентами и сервером, при котором многие клиенты могут одновременно соединяться с сервером. Поэтому на это соединение должна быть выделена более широкая полоса пропускания.

2.5 Дополнительные функции коммутаторов

Помимо собственно функции коммутации, то есть распределения поступающих кадров по портам, современные коммутаторы обладают рядом дополнительных функций. Часть этих функций не стандартизирована, то есть они являются фирменными разработками компаний, выпускающих сетевое оборудование, часть становится стандартом де-факто, когда фирменные разработки получают широкое признание и стандартизируются. Рассмотрим наиболее распространенные и используемые в современных сетях дополнительные функции коммутаторов, основываясь, в основном на [9].

Практически все современные управляемые коммутаторы поддерживают технологию виртуальных локальных сетей (VLAN). Данная технология позволяет разделить порты одного или нескольких коммутаторов по различным сетям, изолированным друг от друга.

Причин такого разделения единой сети на виртуальные сети несколько.

Во-первых, сети на коммутаторах слабо защищены от так называемого широковещательного шторма (Broadcast storm). Под широковещательным штормом понимается наводнение сети широковещательными кадрами, то есть кадрами, у которых в адресе назначения одни единицы (DA=F::F). Отказаться от широковещательных кадров нельзя, на их использовании основаны многие протоколы, которые будут рассмотрены ниже. Широковещательный шторм может быть спровоцирован некорректной работой сетевого оборудования, неправильной настройкой протокола покрывающего дерева STP (данный протокол будет рассмотрен ниже), а также действиями злоумышленников.

Во-вторых, при разделении сети легче локализовать возникающие проблемы, проще и быстрее можно устранить возникшие неисправности.

В-третьих, с использованием виртуальных сетей легче разграничить права доступа к ресурсам сети, хотя эту задачу решают не только VLAN, а еще ряд технологий и протоколов.

Технология VLAN описана в открытом стандарте IEEE 802.1Q [10], кроме того, существует фирменный стандарт Cisco Systems, называемый ISL.

Реализация технологии VLAN на одном коммутаторе предельно проста – при разделении портов между разными виртуальными сетями для каждой из них как бы создается своя адресная таблица (на практике адресная таблица остается единой, но в ней указывается номер VLAN).

Например, если в сети, представленной на рисунке 2.11, порты Ethernet 0 и Ethernet 1 принадлежат VLAN 2, а порты Ethernet 2 и Ethernet 3 – VLAN 3, адресная таблица примет вид, показанный в таблице 2.3.

Таблица 2.3 – Адресная таблица при использовании VLAN

VLAN	Порт	Адрес рабочей станции
2	Ethernet 0	0060.2FBA.5B24
2	Ethernet 1	00E0.B02B.7D01
3	Ethernet 2	0002.1699.4740
3	Ethernet 3	0090.2142.0C59

При поступлении кадра на один из портов коммутатор обращается к адресной таблице и отыскивает в ней запись с адресом рабочей станции, идентичным адресу DA в принятом кадре. Отыскав нужную запись, коммутатор сравнивает номер VLAN источника с номером VLAN получателя, и при их совпадении продвигает кадр. В противном случае кадр отбрасывается (фильтруется).

При получении широковещательного кадра, например, на порт Ethernet 0 он будет передан только на порт Ethernet 1, что исключает распространение широковещательного шторма во сей сети.

Следует отметить, что для рабочих станций организация виртуальных сетей (как и ведение адресных таблиц) на коммутаторе абсолютно прозрачна – станции отправляют и принимают кадры в обычном режиме, а разделение трафика разных VLAN осуществляет сам коммутатор.

Таким же образом можно организовать несколько виртуальных сетей на нескольких коммутаторах. Рассмотрим пример, представленный на рисунке 2.12.

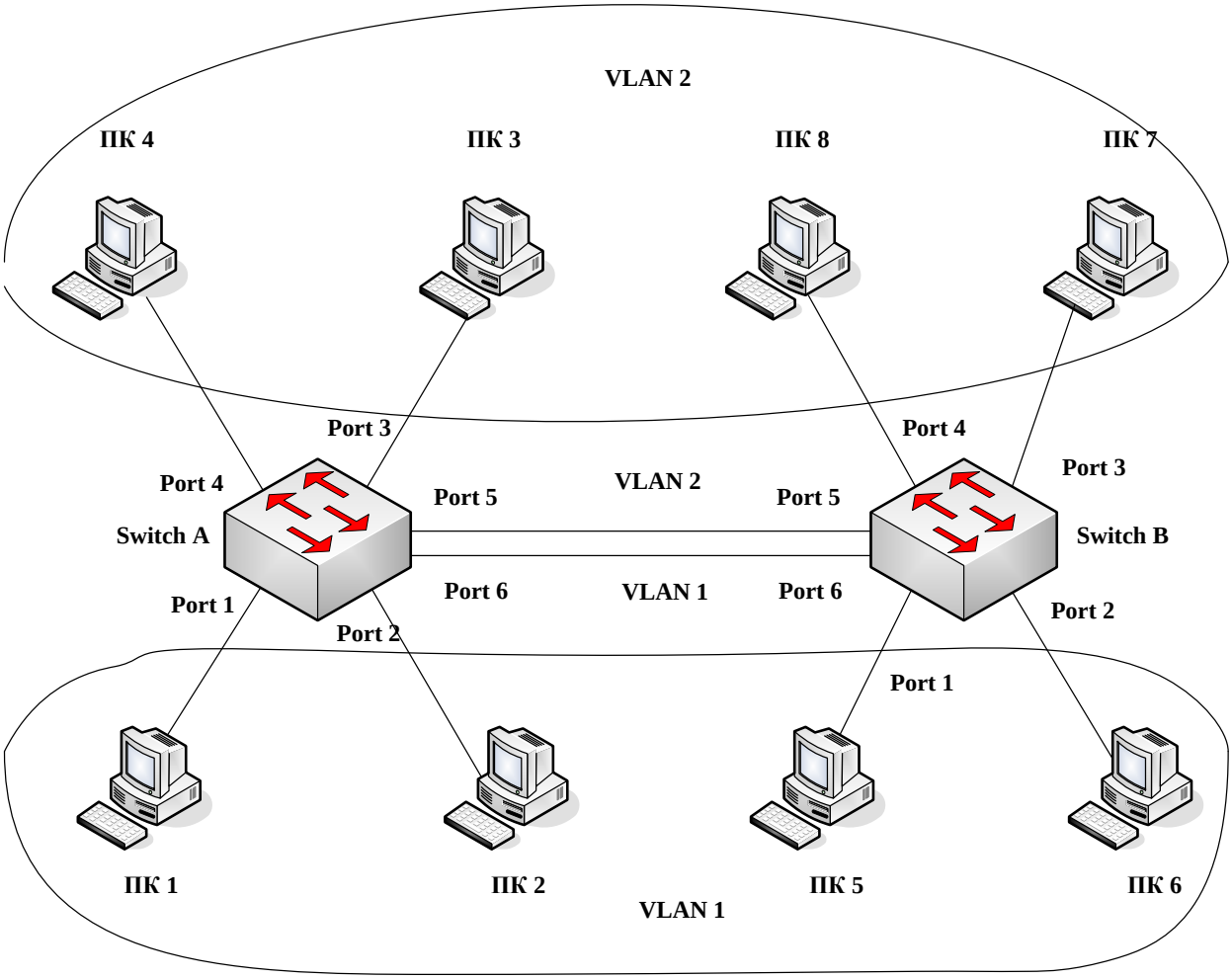

Рисунок 2.12 – Организация VLAN на двух коммутаторах

Для обеспечения корректной работы коммутаторов с виртуальными сетями достаточно в обоих коммутаторах порты 1,2 и 6 соотнести с VLAN 1, а порты 3,4 и 5 – с VLAN 2. Так же, как и в примере с одним коммутатором, никаких изменений в передаваемых кадрах не происходит, но виртуальные сети надежно изолированы друг от друга.

Такой способ организации виртуальных сетей на нескольких коммутаторах не совсем удобен – для каждой VLAN в соединении коммутатор-коммутатор необходимо выделить отдельный порт (в нашем примере это порт 5 для VLAN 2 и порт 6 для VLAN 1). Если виртуальных сетей несколько, это приведет к большому количеству избыточных соединений и расходованию портов. Гораздо экономичнее было бы

использовать только один порт (например, 5), и разрешить передавать через него трафик обеих виртуальных сетей.

В рассматриваемом примере порты коммутатора никаких изменений в передаваемые кадры не вносили, следовательно, если порт 5 использовать для передачи данных нескольких VLAN, коммутатор при приеме не сможет различить, к какой виртуальной сети относится источник, передавший принятый кадр. Если же внести в кадр определенные изменения («пометить» кадр), то проблема была бы решена.

Стандарт IEEE 802.1Q [10] предусматривает внесение для этой цели в кадр так называемого тэга (tag), который указывал бы принадлежность кадра к трафику той или иной виртуальной сети. Порт, способный добавлять к кадру Ethernet тэг, называется тэгированным (tagging).

Предположим, что в примере, показанном на рисунке 2.12, были отключены порты 6 обоих коммутаторов, и данные обеих виртуальных сетей необходимо передавать через тэгированный порт 5. При передаче, например, кадра от ПК4 к ПК8 коммутатор Switch A, приняв кадр в буфер порта 4, по адресной таблице определяет, что ПК8 доступен через порт 5. После приема кадра в буфер порта 5 коммутатора Switch A к нему добавляется тэг, соответствующий VLAN 2, и модифицированный таким образом кадр передается на порт 5 коммутатора Switch B. После буферизации кадра тэг считывается, определяется номер VLAN, после чего кадр с удаленным тэгом передается на порт 4.

Таким образом, тэгированные порты добавляют тэг при передаче кадра и отбрасывают его при приеме.

Остальные порты работают в прежнем режиме, никаких изменений в передаваемые кадры не вносят, поэтому называются нетэгированными (untagging).

Следует отметить, что различные производители по-разному называют рассмотренные выше порты. Например, в терминологии Cisco Systems тэгированный порт называется Trunk-порт, нетэгированный –

Access-порт. Все коммутаторы позволяют при конфигурировании назначать режим работы портов.

Как отмечалось выше, виртуальные сети надежно изолированы друг от друга. Однако на практике часто возникают задачи гибкого объединения нескольких VLAN между собой. Эту задачу решают маршрутизаторы и коммутаторы третьего уровня, которые будут рассмотрены ниже.

Следующей дополнительной функцией коммутаторов является блокировка резервных портов с целью исключения так называемых «петель».

Под «петлей» понимается такое соединение сетевых устройств между собой, когда между ними существует два или более путей передачи данных.

С одной стороны, «петель» можно избежать при грамотном проектировании сети. Однако при последующей модернизации и расширении сети «петли» могут возникнуть непроизвольно.

С другой стороны, наличие «петель» в сети желательно, так как при существовании нескольких путей передачи данных имеется возможность распараллелить передачу, а также повысить отказоустойчивость сети.

Рассмотрим сначала, к чему в сети на коммутаторах приводит наличие «петли». Рассмотрим для примера сеть, изображенную на рисунке 2.13.

Коммутаторы в нашем примере обозначим как Switch A, Switch B, Switch C, а MAC-адреса рабочих станций – A, B, C (однако следует помнить, что реальные MAC-адреса – это шестибайтовые числа). Интерфейсы коммутатора обозначены как Fa 0/1 – Fa 0/3 (Fa – сокращение от Fast Ethernet).

Считаем, что в адресные таблицы коммутаторов не внесены статические записи. Предположим, что в начальный момент времени станция A отправляет широковещательный кадр. Этот кадр поступает на интерфейс Fa 0/1 коммутатора Switch A. Коммутатор ретранслирует этот кадр на порты Fa 0/2 и Fa 0/3, соответственно, его принимают коммутаторы Switch B и Switch C. Одновременно с этим коммутатор Switch A формирует в своей

адресной таблице динамическую запись, представленную в таблице 2.4 (для упрощения столбец с указанием номера VLAN в таблице не показан).

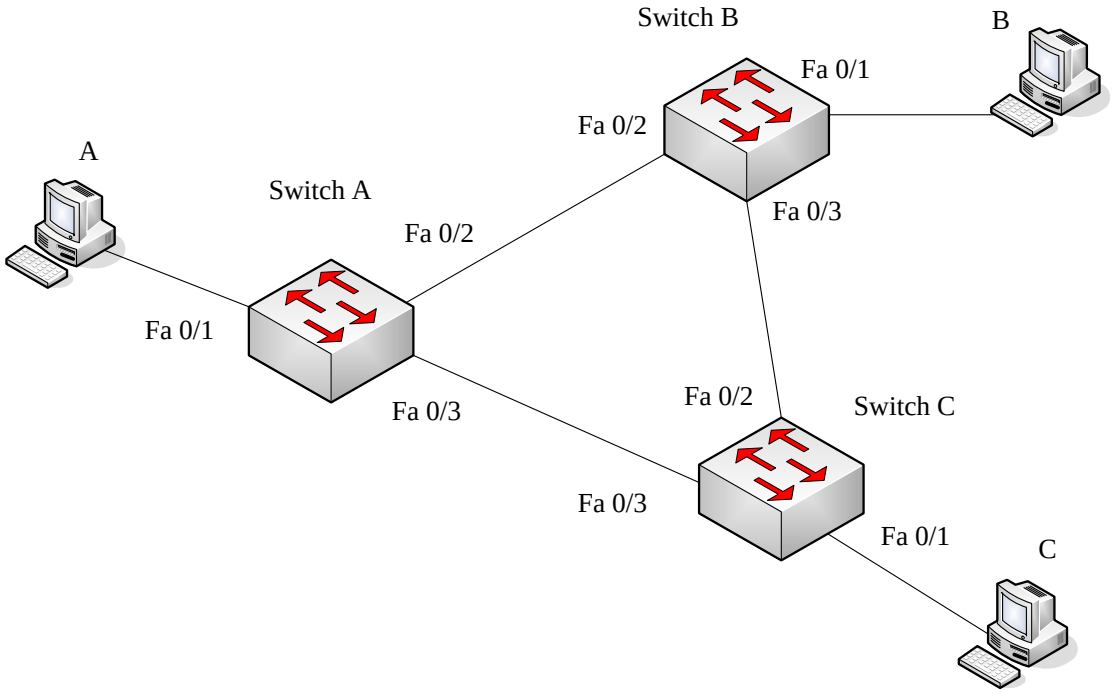

Рисунок 2.13 – Пример сети с «петлей»

Таблица 2.4 – Адресная таблица коммутатора Switch A

Порт	Адрес рабочей станции
Fa 0/1	A

Коммутатор Switch B, приняв кадр, формирует запись, представленную в таблице 2.5.

Таблица 2.5 – Адресная таблица коммутатора Switch B

Порт	Адрес рабочей станции
Fa 0/2	A

Коммутатор Switch C, приняв кадр, формирует запись, представленную в таблице 2.6.

Таблица 2.6 – Адресная таблица коммутатора Switch C

Порт	Адрес рабочей станции
Fa 0/3	A

Так как кадр широковещательный, коммутаторы Switch B и Switch C также распространяют его на все свои задействованные порты. Соответственно, коммутатор Switch B передает его на порт Fa 0/3, в результате чего кадр принимается на порту Fa 0/2 коммутатора Switch C. Коммутатор Switch C корректирует запись в адресной таблице (новая запись представлена в таблице 2.7) и отправляет его на все свои порты, в том числе и на порт Fa 0/3 по направлению к коммутатору Switch A.

Таблица 2.7 – Адресная таблица коммутатора Switch C после корректировки

Порт	Адрес рабочей станции
Fa 0/2	A

Коммутатор Switch A, приняв кадр, тоже корректирует таблицу, как это показано в таблице 2.8.

Таблица 2.8 – Адресная таблица коммутатора Switch A после корректировки

Порт	Адрес рабочей станции
Fa 0/3	A

Аналогичным образом копия того же кадра проходит через коммутатор Switch C в направлении коммутатора Switch B и т.д.

Так как в кадре Ethernet отсутствует поле «время жизни» (Time to Live – TTL), копии одного и того же кадра будут бесконечно циркулировать по «петле» в противоположных направлениях, постоянно корректируя содержимое адресных таблиц всех коммутаторов, через которые они проходят.

Очевидно, что для исключения такой ситуации между любой парой коммутаторов должен существовать единственный путь передачи данных. Даже если сеть небольшая, как в рассмотренном примере, наличие альтернативного пути передачи повысило бы отказоустойчивость сети – при отказе основного пути можно было бы переключиться на резервный.

Для решения этой задачи был разработан алгоритм покрывающего дерева (Spanning Tree Algorithm – STA) и реализующий его протокол покрывающего дерева (Spanning Tree Protocol – STP). Данный протокол описан в открытом стандарте IEEE 802.1D [10]. Реализация данного протокола в современных коммутаторах позволяет отдельные порты перевести в состоянии блокировки (blocking), в котором порты остаются задействованными, но они не передают пользовательский трафик. При отказе неблокированного порта автоматически подключаются заблокированные порты.

Кроме «классической» версии протокола STP на практике широко применяется «ускоренная» версия Rapid Spanning Tree Protocol (RSTP), а также Multiple Spanning Tree Protocol (MSTP), позволяющая работать с виртуальными сетями VLAN [3].

3 Конфигурирование коммутаторов

3.1 Основные сведения о программном продукте Cisco Packet Tracer

Современные коммутаторы делятся на два больших класса – неуправляемые и управляемые. Неуправляемые коммутаторы представляют собой простейшие устройства канального (второго в терминах ISO/OSI) уровня, выполняющие операции продвижения и фильтрации на основе динамических записей адресной таблицы.

Управляемые коммутаторы обладают гораздо большим функционалом. Например, в управляемый коммутатор можно вносить статические записи, производить фильтрацию передаваемого трафика, делить сеть на несколько виртуальных сетей (VLAN), и т.д. Соответственно, управляемый коммутатор нуждается в конфигурировании.

Конфигурирование коммутаторов будем рассматривать на примере продукции фирмы Cisco Systems главным образом потому, что эта фирма разработала для обучения программный продукт Cisco Packet Tracer. Конечно, это далеко не единственный программный продукт для моделирования сетей, однако к его несомненным достоинствам относится то, что конфигурирование сетевого оборудования в программе практически ничем не отличается от конфигурирования реального оборудования. Кроме того, продукт является бесплатным для участников Сетевой Академии Cisco, к которым относится и СКФ МТУСИ.

Реальное оборудование Cisco конфигурируется различными способами, Рассмотрим конфигурирование с использованием консольного порта с последовательным линейным интерфейсом (CLI – Common Line Interface). Для этого необходим консольный кабель, распайка и внешний вид которого показан на рисунках 3.1 и 3.2 соответственно.

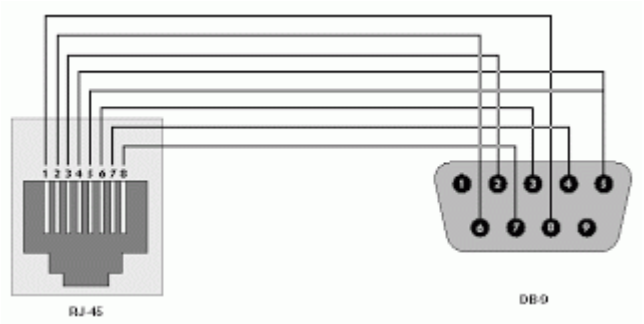

Рисунок 3.1 – Распайка консольного кабеля

Рисунок 3.2 – Внешний вид консольного кабеля

Cisco Packet Tracer является довольно легким и удобным в использовании симулятором устройств Cisco, идеально подходящим для обучения работе с реальным оборудованием. Конечно, он не позволяет воспроизвести абсолютно все функции реальных сетевых устройств, но способен дать более чем общее представление о принципах их работы, и способах их конфигурирования. Кроме того, конфигурирование устройств других производителей хоть и несколько отличается от конфигурирования устройств Cisco, но общие принципы остаются неизменными. Соответственно, получив навыки конфигурирования сетевого оборудования с использованием симулятора, несложно освоить и устройства других производителей.

Интерфейс Cisco Packet Tracer представлен на рисунке 3.3.

Большую часть данного окна занимает рабочая область, в которой можно размещать различные сетевые устройства, соединять их различными

способами и как следствие, получать различные сетевые топологии. Внешний вид интерфейса может несколько меняться от версии к версии, но основные функции, которые мы рассматриваем в учебном пособии, остаются неизменными.

Сверху, над рабочей областью, расположена главная панель программы и ее меню. Меню позволяет выполнять сохранение, загрузку сетевых топологий, настройку симуляции, а также много других интересных функций. Главная панель содержит на себе наиболее часто используемые функции меню (рисунок 3.4).

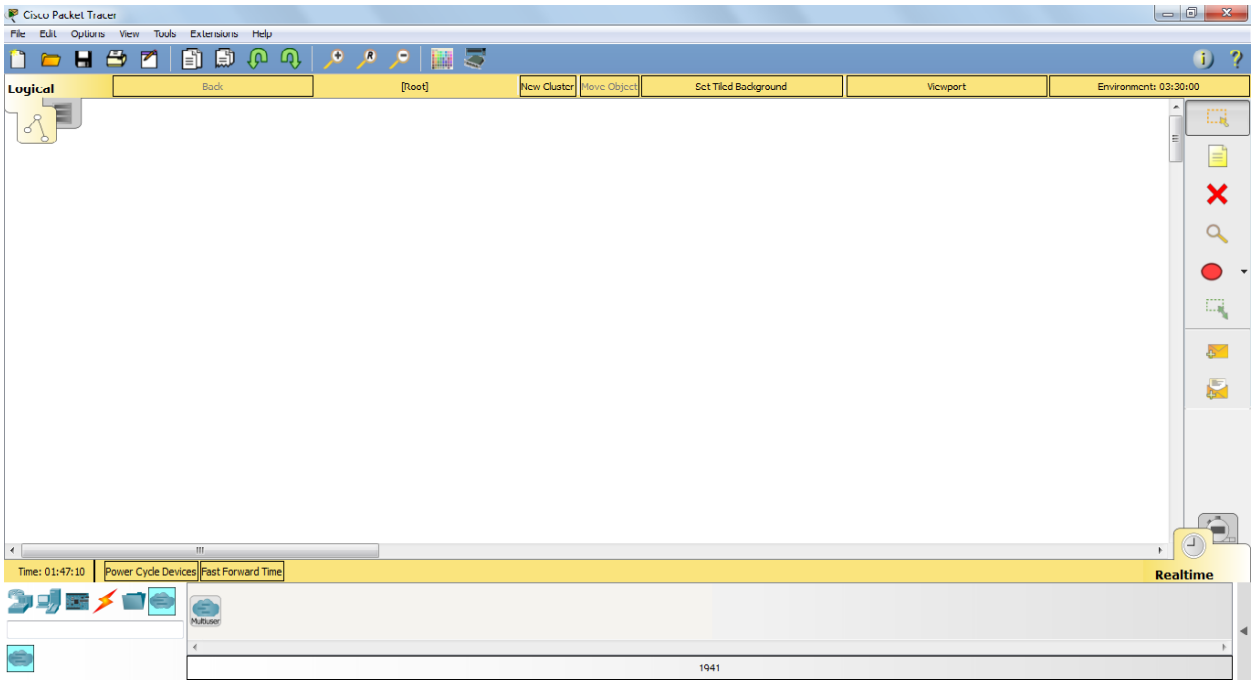

Рисунок 3.3 – Интерфейс Cisco Packet Tracer

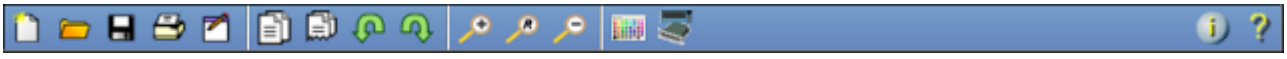

Рисунок 3.4 – Главное меню программы

Справа от рабочей области расположена боковая панель, содержащая ряд кнопок, отвечающих за перемещение полотна рабочей области, удаление

объектов и т.д. Снизу, под рабочей областью, расположена панель оборудования (рисунок 3.5).

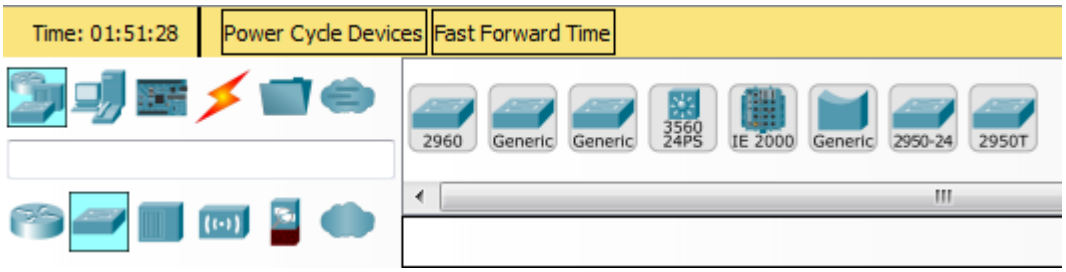

Рисунок 3.5 – Панель оборудования

Данная панель содержит в своей левой части типы доступных устройств, а в правой части доступные модели. При выполнении различных работ эту панель придется использовать намного чаще, чем все остальные. Поэтому рассмотрим ее более подробно.

При наведении на каждое из устройств, в прямоугольнике, находящемся в центре, между ними будет отображаться его тип. Типы устройств, наиболее часто используемые в Cisco Packet Tracer, представлены на рисунке 3.6.

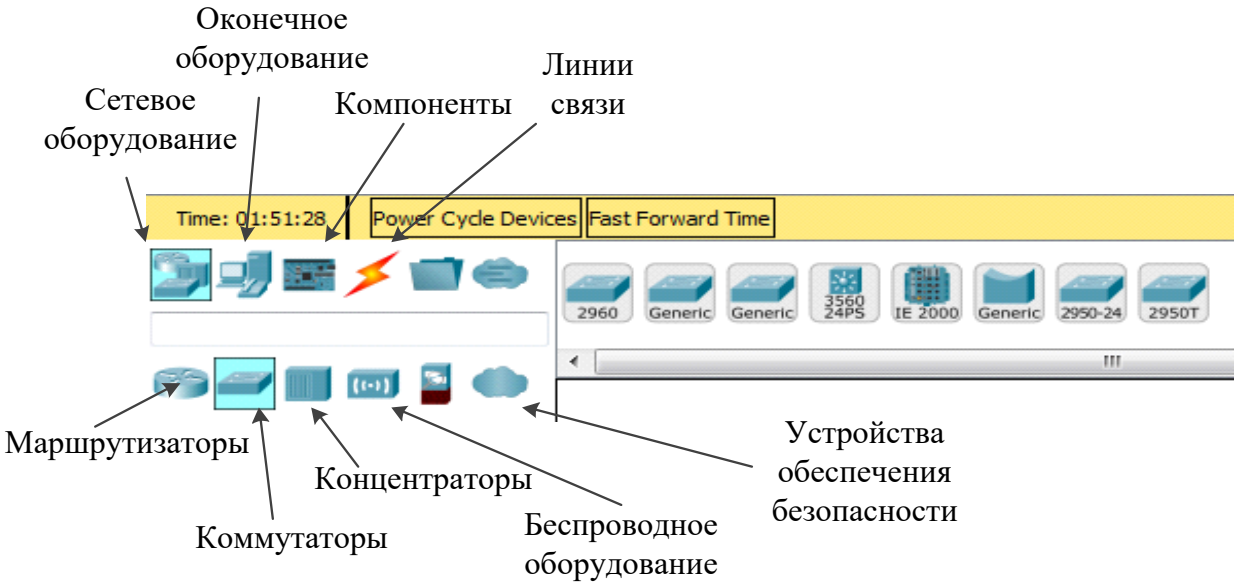

Рисунок 3.6 – Основные типы устройств

Отдельного рассмотрения заслуживают типы соединений. Перечислим наиболее часто используемые из них (рассмотрение типов подключений идет слева направо, в соответствии с рисунком 3.7).

Рисунок 3.7 – Типы соединений

— Автоматический тип – при данном типе соединения PacketTracer автоматически выбирает наиболее предпочтительные тип соединения для выбранных устройств;

— Консоль – консольные соединение;

— Медь Прямое – соединение медным кабелем типа витая пара, оба конца кабеля обжаты в одинаковой раскладке. Подойдет для следующих соединений: коммутатор – коммутатор, коммутатор – маршрутизатор, коммутатор – компьютер и др.;

— Медь кроссовер – соединение медным кабелем типа витая пара, концы кабеля обжаты как кроссовер. Подойдет для соединения двух компьютеров;

— Оптика – соединение при помощи оптического кабеля, необходимо для соединения устройств имеющих оптические интерфейсы;

— Телефонный кабель – обыкновенный телефонный кабель, может понадобиться для подключения телефонных аппаратов;

— Коаксиальный кабель – соединение устройств с помощью коаксиального кабеля.

Приведем пример построения простейшей сетевой топологии с использованием симулятора. Кликнув левой кнопкой мыши на значок коммутатора, необходимо выбрать его конкретный тип, например, коммутатор Cisco 2960, и «перетащить» его в рабочую область. Аналогично

разместим в рабочей области четыре компьютера, в результате чего рабочая область будет выглядеть так, как показано на рисунке 3.8.

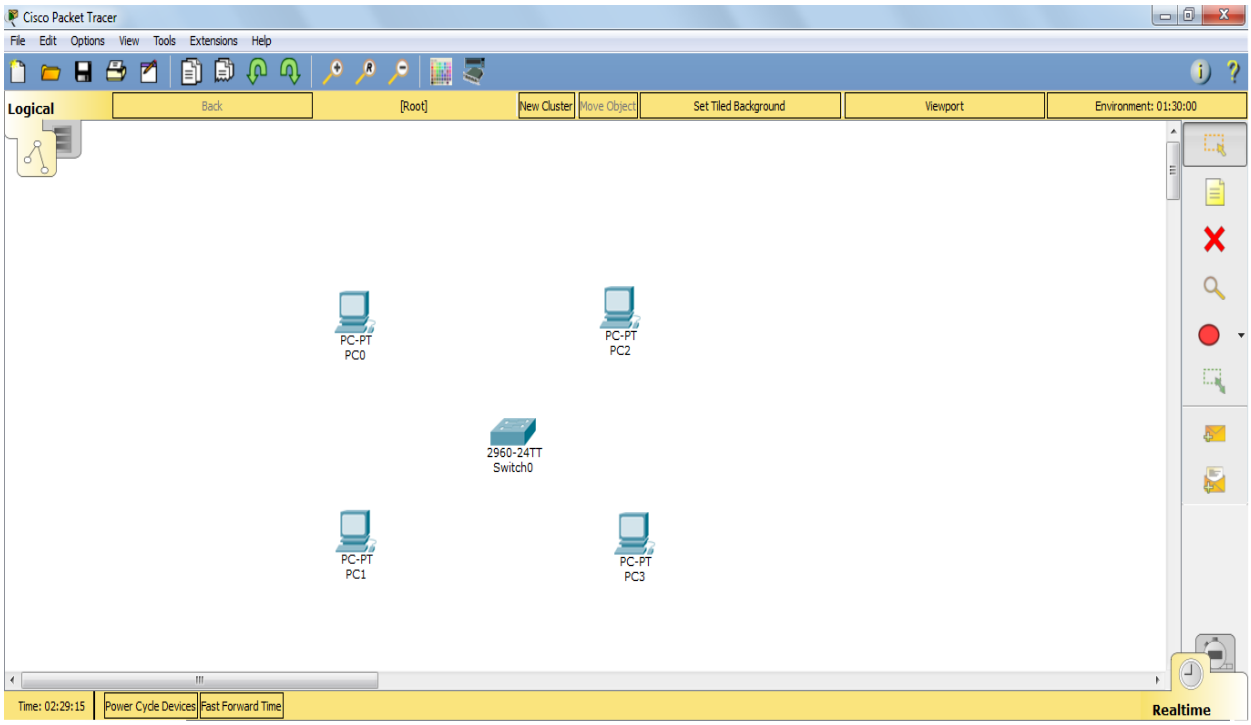

Рисунок 3.8 – Рабочая область с размещенным в ней оборудованием

Для соединения оборудования между собой необходимо щелкнуть левой кнопкой мыши на значок тип соединения в панели устройств, выбрать тип соединения – медь прямое – и кликнуть левой кнопкой мыши по коммутатору. Откроется окно выбора сетевого интерфейса (номера порта коммутатора) к которому необходимо подключить кабель. Для выбора одного из них необходимо кликнуть левой кнопкой мыши по его названию (рисунок 3.9).

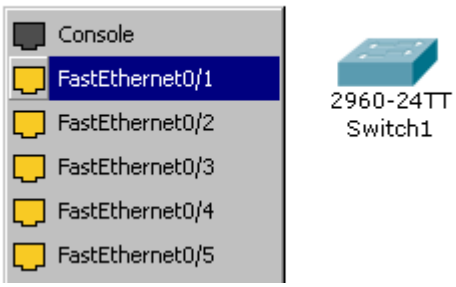

Рисунок 3.9 – Выбор интерфейса для соединения

В оборудовании Cisco интерфейсы нумеруются следующим образом: Название канальной технологии/номер слота/номер интерфейса. Например, запись

FastEthernet0/1

означает первый интерфейс Fast Ethernet, находящийся в нулевом слоте.

Аналогично подключаются порты рабочих станций. Структура построенной таким образом сети представлена на рисунке 3.10.

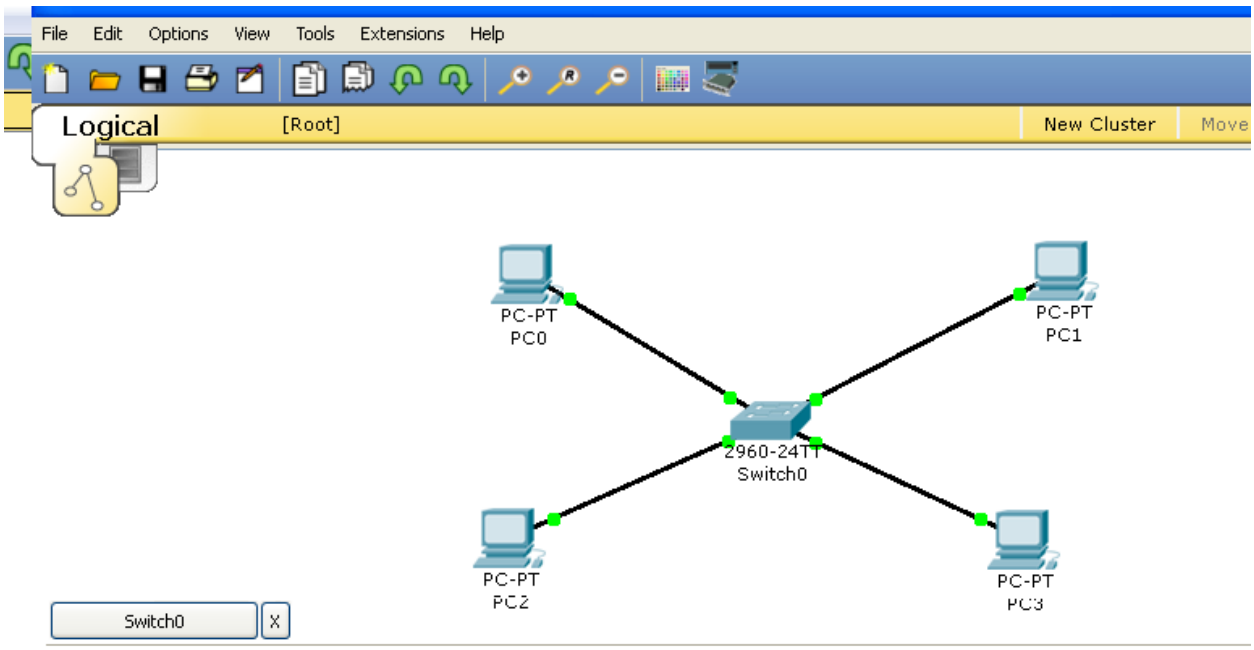

Рисунок 3.10 – Структура сети в рабочей области

Для конфигурирования коммутатора надо дважды щелкнуть на его значке, выбрать в открывшемся окне вкладку CLI, после чего фактически мы попадаем в консольный режим.

3.2 Режимы конфигурирования коммутаторов

Коммутаторы Cisco могут находиться в одном из режимов, представленных в таблице 3.1 (представлены оригинальные названия на английском языке).

Таблица 3.1 – Режимы конфигурирования

Название режима	Приглашение (Prompt)	Описание
User EXEC Mode	Switch>	Пользовательский режим
Privileged EXEC Mode	Switch #	Привилегированный режим
Global Configuration Mode	Switch (config)#	Режим глобального конфигурирования

В таблице 3.1 в первом столбце представлены названия режимов, во втором – приглашение, отображаемое в командной строке, в третьем – описание.

Пользовательский режим (user mode) используется для просмотра состояния устройства, а также для перехода в привилегированный режим (privileged mode). Никакие изменения в конфигурационном файле, в том числе удаления и сохранения текущей конфигурации, в пользовательском режиме производиться не могут. В этом режиме доступны только некоторые команды **show**, т. е. команды просмотра состояния устройства.

Для перехода в привилегированный режим необходимо выполнить команду

enable

Следует отметить, что оборудование Cisco допускает сокращенный ввод команд, если невозможно ее двоякое толкование. Например, вместо enable можно набрать en, и эта команда будет понятна коммутатору, так как никакая другая команда не начинается с сочетания символов en.

Выполнение каждой команды начинается после нажатия клавиши Enter. Для повтора ранее введенных команд можно использовать клавишу ↑.

В привилегированном режиме доступны все команды **show**, возможно удаление конфигурации и сохранение конфигурационного файла в памяти

NVRAM. Возврат в пользовательский режим производится командой **disable** или **exit**:

Switch#exit

Команда exit позволяет вернуться на один уровень вверх. Например, после выполнения команды в режиме глобального конфигурирования коммутатор переходит в привилегированный режим. Если необходимо из любого состояния устройства выйти сразу в пользовательский режим, используется комбинация CTRL-Z.

В глобальном режиме производятся изменения, которые затрагивают коммутатор в целом, поэтому этот режим и называется global configuration mode (режим глобального конфигурирования). Например, в нем можно устанавливать имя коммутатора командой hostname. Имя коммутатора не имеет значения в сети, оно удобно при конфигурировании. Пример:

Switch(config)#hostname Subnet_A

Subnet_A(config)#

В режиме глобального конфигурирования на коммутатор можно устанавливать пароли. Существует несколько видов паролей для обеспечения защиты устройств Cisco. Первые два пароля, enable secret и enable password, используются для обеспечения авторизованного входа в привилегированный режим. На коммутаторе устанавливается один (или оба) из этих паролей. После установки пароля система запрашивает его у пользователя, когда вводится команда enable. Формат команд установки паролей на коммутатор и менем Cisco_A cisco и cisco1 для входа в привилегированный режим приведен ниже:

Switch_A(config)#enable secret cisco

Switch_A(config)#enable password cisco1

Пароль enable secret криптографируется по умолчанию, поэтому является более строгим. Если установлены оба пароля – enable secret и enable password, – то в приведенном примере система будет реагировать на пароль cisco. Пароль enable password по умолчанию не криптографируется, поэтому

его можно посмотреть, например, по команде show running-configuration (сокращенно sh run), которая выполняется из привилегированного режима.

Изменение и создание конфигурации коммутатора Cisco возможно в режиме глобального конфигурирования, вход в который реализуется из привилегированного по команде configure terminal (сокращенно – conf), которая вводит устройство в глобальный режим и позволяет изменять текущую конфигурацию (running-config). При этом приглашение изменяет вид на Switch(config)#:

Switch >ena

Switch #conf t

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

Switch(config)#

Для просмотра адресной таблицы используется команда

show mac address-table

3.3 Утилиты проверки связности сети

Задача конфигурирования сети является достаточно сложной и трудоемкой, и сложность эта возрастает при увеличении масштабов сети и количества используемых приложений. Поэтому необходимы средства мониторинга и диагностики сети. Эти же средства необходимы нам для моделирования построенной и сконфигурированной сети с использованием Cisco Packet Tracer.

Одним из вспомогательных протоколов – протоколом межсетевых управляющих сообщений (Internet Control Message Protocol – ICMP) – как раз и предоставляются такие средства. Основная идея функционирования протокола ICMP заключается в том, что при возникающих проблемах в продвижении пакетов в сети источнику отсылается сообщение, оповещающее его об этой проблеме. Например, если пакет был отброшен маршрутизатором с использованием сетевого фильтра (маршрутизация и сетевые фильтры будут рассмотрены ниже), сообщение протокола ICMP

оповестит источник об этом событии. То же относится и к другим проблемам, возникающим при продвижении пакета – истечение пакетом времени жизни (TTL), отсутствием нужной маршрутной информации в таблице маршрутизации и т.д. Исключение составляют сообщения о проблемах передачи самих ICMP-сообщений, а также некоторых видов пакетов (широковещательных, например), чтобы не наводнять сеть избыточным трафиком.

Так как проблемы с продвижением пакетов могут иметь различную природу, число различных типов ICMP-сообщений также достаточно велико, но все они имеют одинаковую структуру. Несмотря на значительное число типов ICMP-сообщений, они все могут быть разделены на два основных вида [3]:

- сообщения об ошибках;
- сообщения вида «запрос-ответ».

Сообщения об ошибках, как следует из их названия, оповещают узел-источник об ошибках, произошедших в процессе передачи, и каждая такая ошибка характеризуется своим типом сообщения. Сами ошибки могут быть вызваны различными причинами, поэтому в теле такого сообщения содержится дополнительный код, характеризующий причину ошибки.

Сообщения «запрос-ответ» связаны в пары, то есть после передачи сообщения-запроса ожидается получение сообщения-ответа.

В рамках данного пособия нет смысла описывать все типы сообщений протокола ICMP, с ними можно ознакомиться в многочисленных справочниках [11 – 13]. Остановимся здесь более подробно на утилитах, позволяющих с использованием ICMP-сообщений проверять связность сети.

Утилита **ping** широко используется для создания ICMP-сообщений вида «запрос-ответ», с помощью которых принудительно вызывается ответ конкретного узла в сети. Это низкоуровневая утилита, поэтому она не требует наличия каких-либо запущенных процессов на удаленных узлах. Соответственно, успешное получение ответа от удаленного узла далеко не

всегда означает правильность выполнения на нем прикладного процесса, а говорит только о том, что маршрут к этому узлу находится в работоспособном состоянии, запрашиваемый узел находится в сети и включен.

Удаленный узел, получив ICMP-сообщение, сгенерированное утилитой ping (эхо-запрос), формирует и передает ответное сообщение (эхо-ответ). Запросы обычно посылаются несколько раз (количество запросов по умолчанию различно у разных ОС, кроме того, его можно задавать соответствующим ключом), после чего утилита ping выводит статистические данные.

Если не использовать дополнительные опции, утилита вызывается с использованием командной строки командой ping <доменное имя> или ping <IP-адрес>.

Пример использования утилиты ping в ОС Windows приведен на рисунке 3.11.

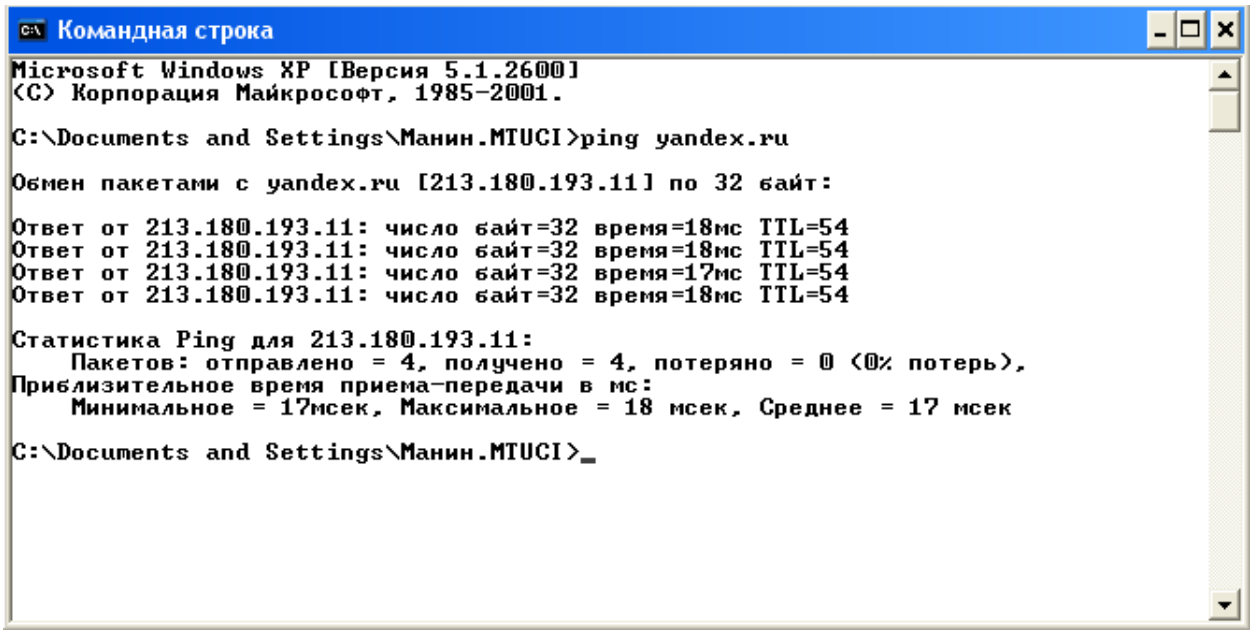

Рисунок 3.11 – Использование утилиты ping

Как следует из рисунка, было отправлено четыре запроса на сервер yandex.ru, получено четыре ответа объемом по 32 байта каждый. Параметр

«время» указывает на величину интервала между отправкой эхо-запроса и получением на него эхо-ответа. Параметр TTL позволяет определить количество маршрутизаторов, через которые проходили пакеты, пока не добрались до узла назначения, так как каждый из них уменьшает TTL на единицу.

При конфигурировании сети утилита ping находит широкое применение для проверки связности сети, правил сетевых фильтров, протоколов маршрутизации и т.д., соответственно, и мы будем ее широко использовать в дальнейшем.

Утилита **traceroute** (в реализациях Windows используется написание tracert) позволяет выявлять последовательность маршрутизаторов, через которые проходит пакет на пути к пункту своего назначения. У этой команды есть много опций, большинство из которых применяются редко. Традиционно используется формат traceroute <доменное имя>, которое может быть задано в символической или числовой форме. Выходная информация представляет собой список узлов, начиная с первого маршрутизатора и кончая пунктом назначения.

Принцип работы traceroute основан на установке поля времени жизни (TTL) исходящего пакета таким образом, чтобы это время истекало до достижения пакетом пункта назначения. При получении пакета с обнуленным полем TTL текущий маршрутизатор отправит сообщение об ошибке на узел-источник. Каждое приращение поля времени жизни позволяет пакету пройти на один шаг дальше.

Утилита traceroute посылает для каждого значения поля TTL три пакета. Если промежуточный шлюз распределяет трафик по нескольким маршрутам, то эти пакеты могут возвращаться разными промежуточными узлами (маршрутизаторами). Некоторые маршрутизаторы не посылают уведомлений о пакетах, время жизни которых истекло, а некоторые посылают уведомления, которые поступают обратно с задержкой, превышающей время ожидания на узле-источнике. Эти маршрутизаторы

обозначаются рядом звездочек. Если конкретный маршрутизатор определить нельзя, все равно с помощью утилиты traceroute можно увидеть следующие за ним узлы маршрута. Заметим, что в связи с использованием на сетях динамической маршрутизации, в разные моменты времени можно получить различные маршруты прохождения пакетов.

Пример использования утилиты **tracert** в ОС Windows представлен на рисунке 3.12.

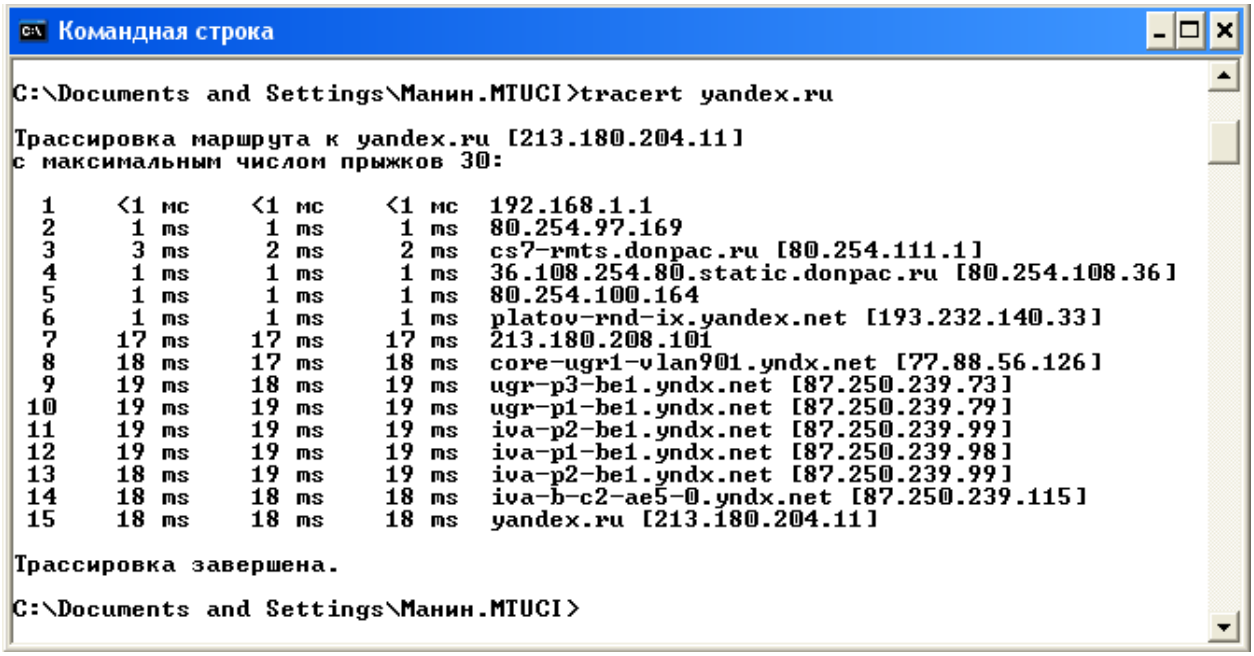

Рисунок 3.12 – Использование утилиты tracert

Как видно из рисунка, утилита позволяет отобразить не только IP-адреса промежуточных узлов, но и их доменные имена, если они существуют.

Утилиты **ping** и **traceroute** сочетают в себе хорошие возможности, предназначенные для тестирования сетевых подключений, но существует команда, которая совмещает в себе все эти возможности, а также содержит некоторые дополнительные особенности. Утилита **PathPing** отправляет многочисленные сообщения с эхо-запросами каждому маршрутизатору, который находится между исходным узлом и узлом назначения, после чего,

на основании пакетов, полученных от каждого из них, вычисляет процентное соотношение пакетов, возвращаемых в каждом шаге. Так как утилита **PathPing** показывает степень потери пакетов на каждом маршрутизаторе или узле, с ее помощью можно точно определить маршрутизаторы и узлы, на которых возникают сетевые проблемы. Эквивалентно утилите traceroute, утилита **PathPing** идентифицирует маршрутизаторы, которые расположены на пути к узлу назначения, после чего она периодически в течение заданного времени обменивается пакетами со всеми маршрутизаторами и на основании числа пакетов, полученных от каждого из них, обрабатывает статистику.

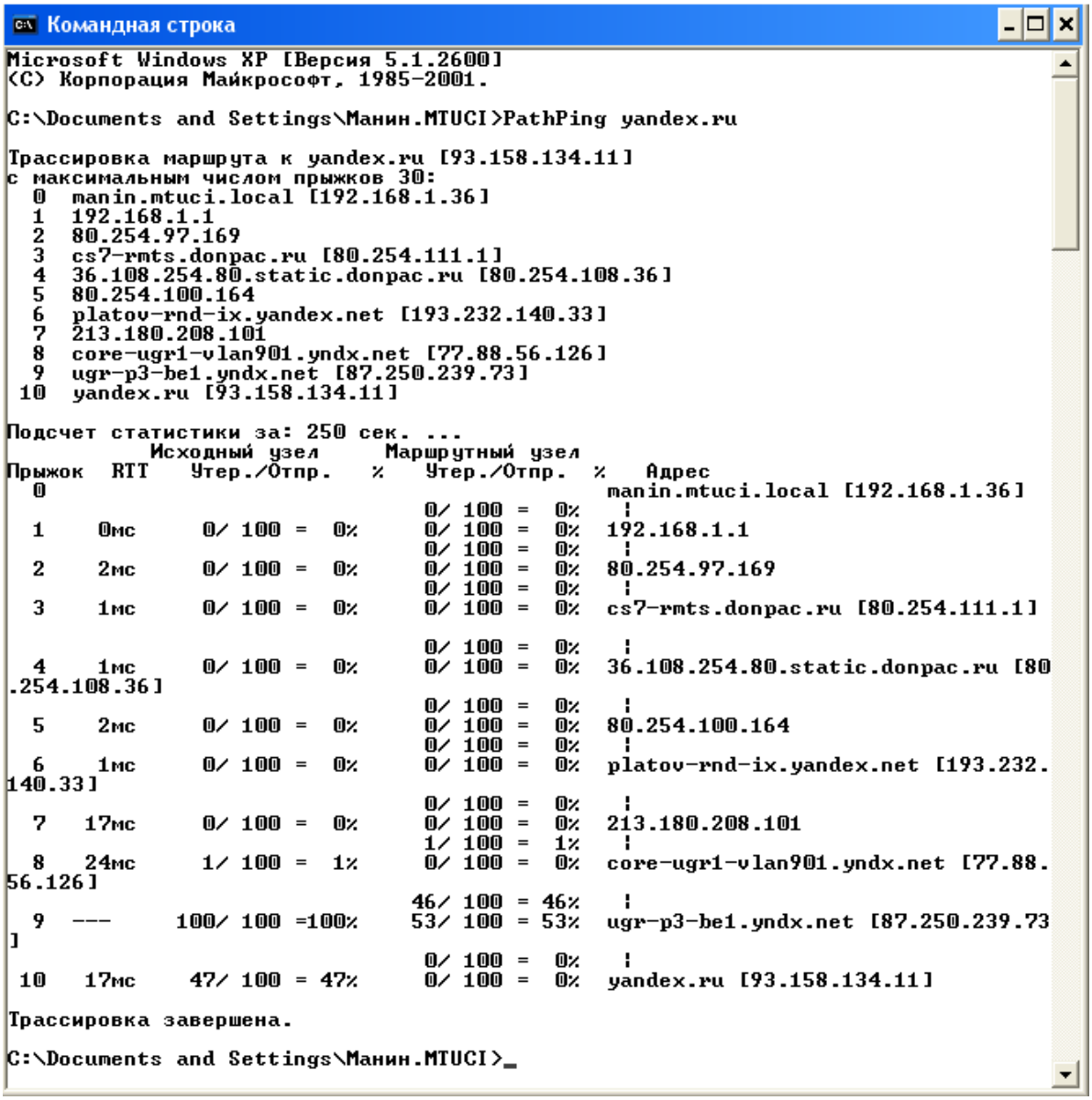

Рисунок 3.13 – Использование утилиты PathPing

В отличие от предыдущих утилит, во избежание перегрузки сети пакеты должны передаваться через довольно большие интервалы времени. Подобно утилите **traceroute** утилита **PathPing** сначала выводит путь, после чего в течение некоторого времени выдает сообщение о том, что она занята. Именно в этот период происходит сбор сведений со всех маршрутизаторов, перечисленных в статистике, и со всех соединений между ними.

В сети, в которой присутствуют только коммутаторы, имеет смысл использовать только утилиту **ping.**

Вернемся к сети, представленной на рисунке 3.10. Так как все рабочие станции соединены с коммутатором, они должны быть доступны друг другу. Для проверки этого необходимо ввести в рабочие станции минимальную сетевую конфигурацию, смысл которой будет рассмотрен ниже (в противном случае просто не будет возможности задать адрес узла в утилите **ping).**

В минимальную конфигурацию рабочей станции входят:

- IP-адрес;

- маска подсети;

Для ввода этой информации в Cisco Packet Tracer необходимо щелкнуть левой кнопкой мыши на рабочей станции, перейти на вкладку Desktop (рабочий стол), и нажать на ярлык IP Configuration (рисунок 3.14).

Интерфейс рабочей станции позволяет сконфигурировать ее как в соответствии с протоколом IPv.4 (IP Configuration), так и в соответствии с протоколом IPv.6 (IP6 Configuration). В данном случае будем использовать поля IP Configuration.

Как указывалось выше, в нашем примере достаточно задать только два параметра – IP-адрес и маску подсети. Зададим станциям следующие адреса:

192.168.1.1;

192.168.1.2;

192.168.1.3;

192.168.1.4.

Маска подсети (Subnet Mask) для всех четырех станций принимает значение 255.255.255.0.

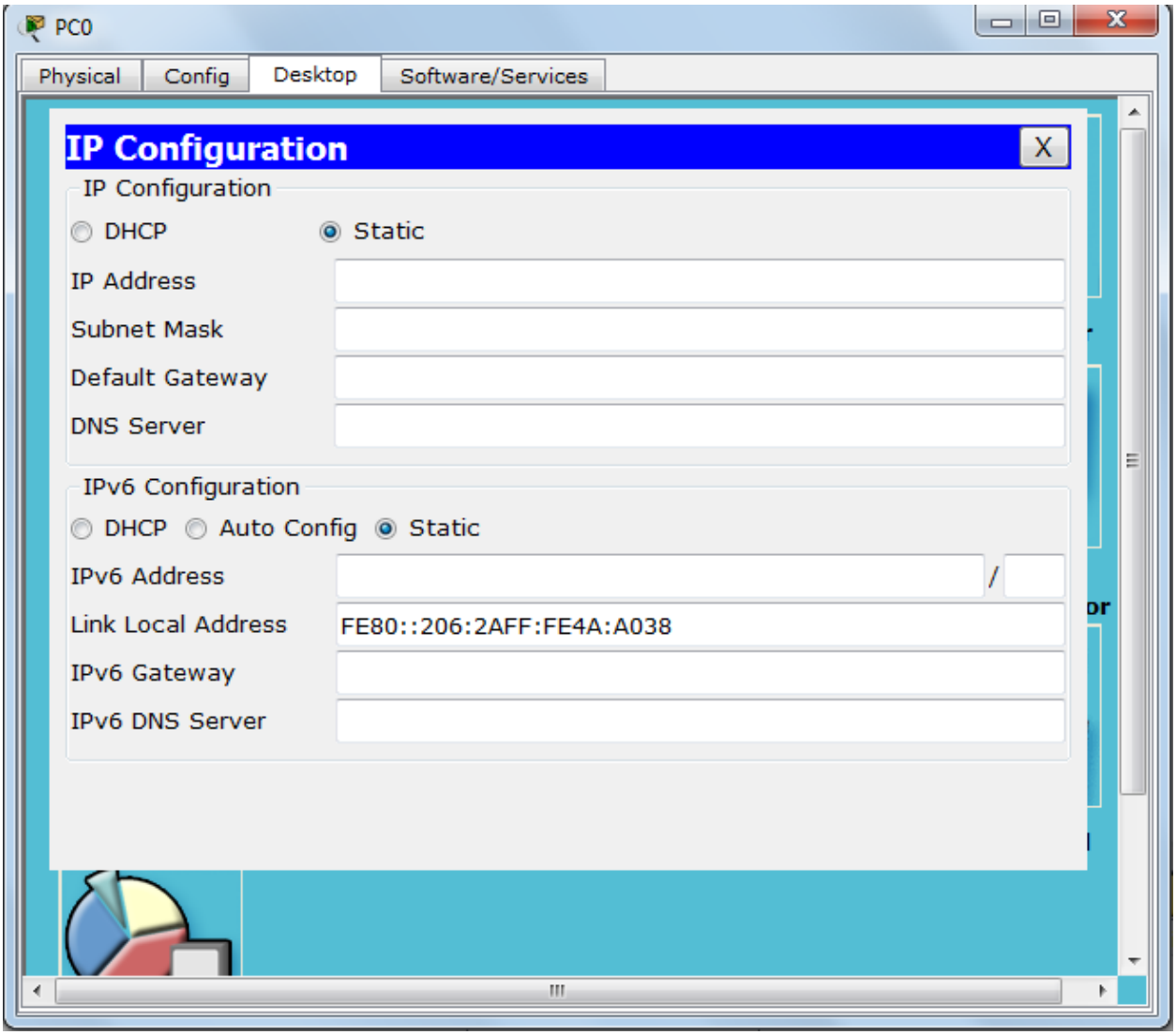

Рисунок 3.14 – Конфигурирование рабочей станции

Более подробно о данных адресах и о маске подсети речь пойдет в главе 4.

После этого можно использовать утилиту **ping** для проверки связности сети. Для этого на рабочем столе PC0 выберем значок Command Prompt (командная строка), и наберем команду

ping 192.168.1.2

Результат выполнения этой команды показан на рисунке 3.15.

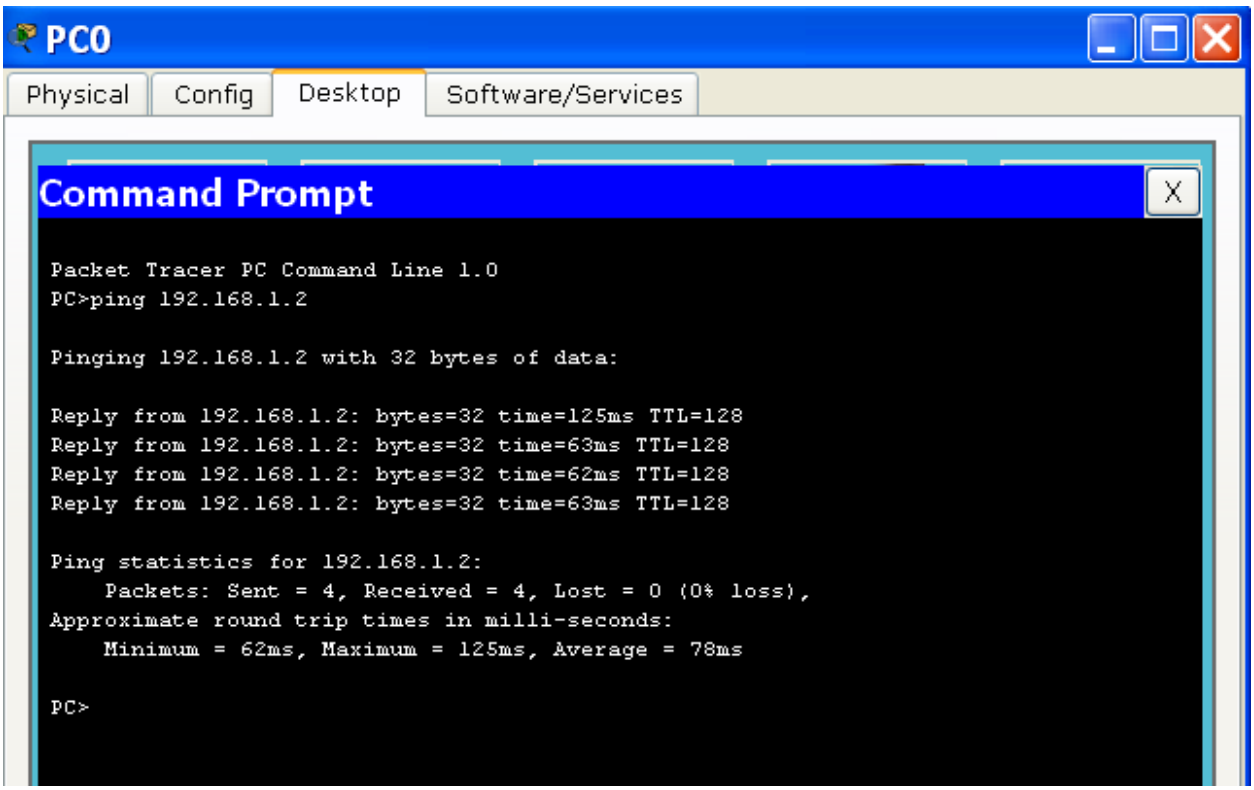

Рисунок 3.15 – Результат проверки связности PC0 и PC1

Как видно из рисунка, утилита **ping** четыре раза отсылала эхо-запросы по 32 байта каждый, параметр time (время) указывает на интервал между отправлением эхо-запроса и получением на него эхо-ответа. Параметр TTL (дословно расшифровывается как Time to live) помогает узнать через сколько маршрутизаторов прошли пакеты, пока добирались до пункта назначения. Операционная система устанавливает некоторое значение TTL по умолчанию (например, Windows устанавливает TTL равным 128), а при прохождении через каждый маршрутизатор данное значение уменьшается на 1. В нашем примере маршрутизаторов нет, поэтому TTL=128.

Внизу приведена статистика, показывающая, что отправлено было 4 пакета (Sent=4), принято 4 (Received=4), потеряно 0 (Lost=0), а также статистику интервала между передачей запроса и получением ответа – минимальное 62 мс (Minimum=62 ms), максимальное 125 мс (Maximum=125 ms), среднее 78 мс (Average=78 ms).

Таким образом, мы убедились, что рабочая станция PC0 и рабочая станция PC1 доступны друг другу. Чтобы увидеть результат выполнения команды в случае недоступности, отправим запрос по несуществующему в данной сети адресу, например, 192.168.1.5. Результат приведен на рисунке 3.16.

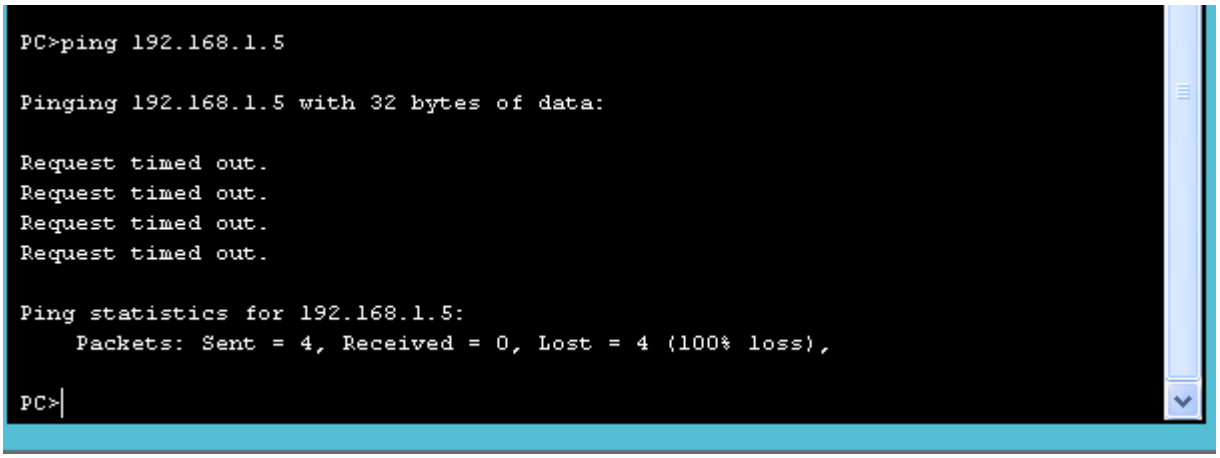

Рисунок 3.16 – Результат выполнения команды

Из рисунка 3.16 следует, что при ожидании ответа на каждый из четырех переданных пакетов был превышен интервал времени ожидания. Статистика сообщает, что было передано 4 запроса, получено 0 ответов, 4 пакета потеряно (100% потерь).

Конфигурирование коммутатора в этом примере не выполнялось, однако можно просмотреть его адресную таблицу. Для этого зайдем в привилегированный режим и выполним команду **show mac address-table** (рисунок 3.17).

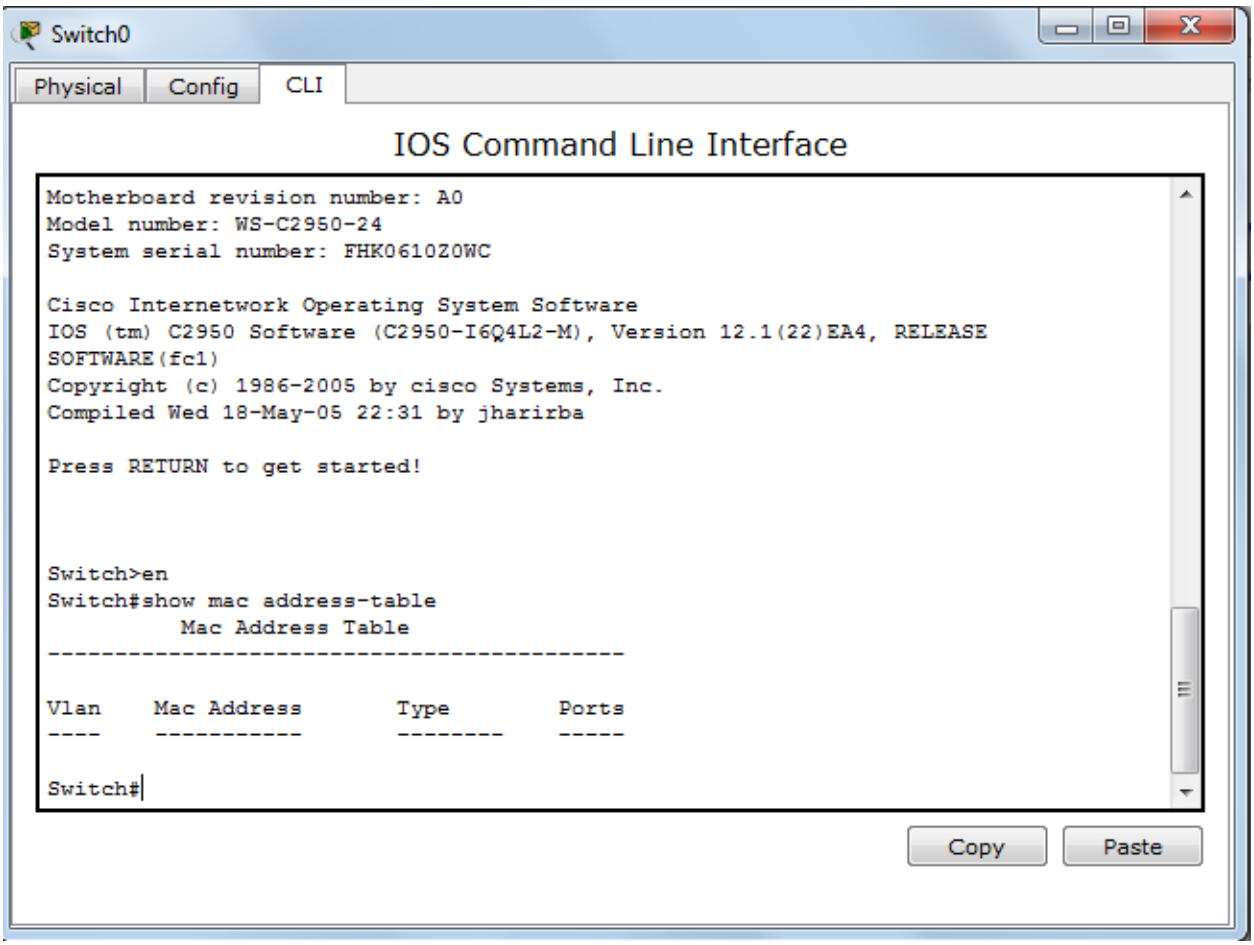

Рисунок 3.17 – Результат выполнения команды show mac address-table

Как видно из рисунка 3.17, адресная таблица пуста, в ней нет ни статических записей (мы их не вносили), ни динамических (коммутатор не прошел процедуру обучения). Для прохождения процедуры обучения достаточно передать от каждого из PC любые кадры, например, ICMP-запросы с использованием утилиты **ping**. Если после этого опять просмотреть адресную таблицу, то она примет вид, показанный на рисунке 3.18.

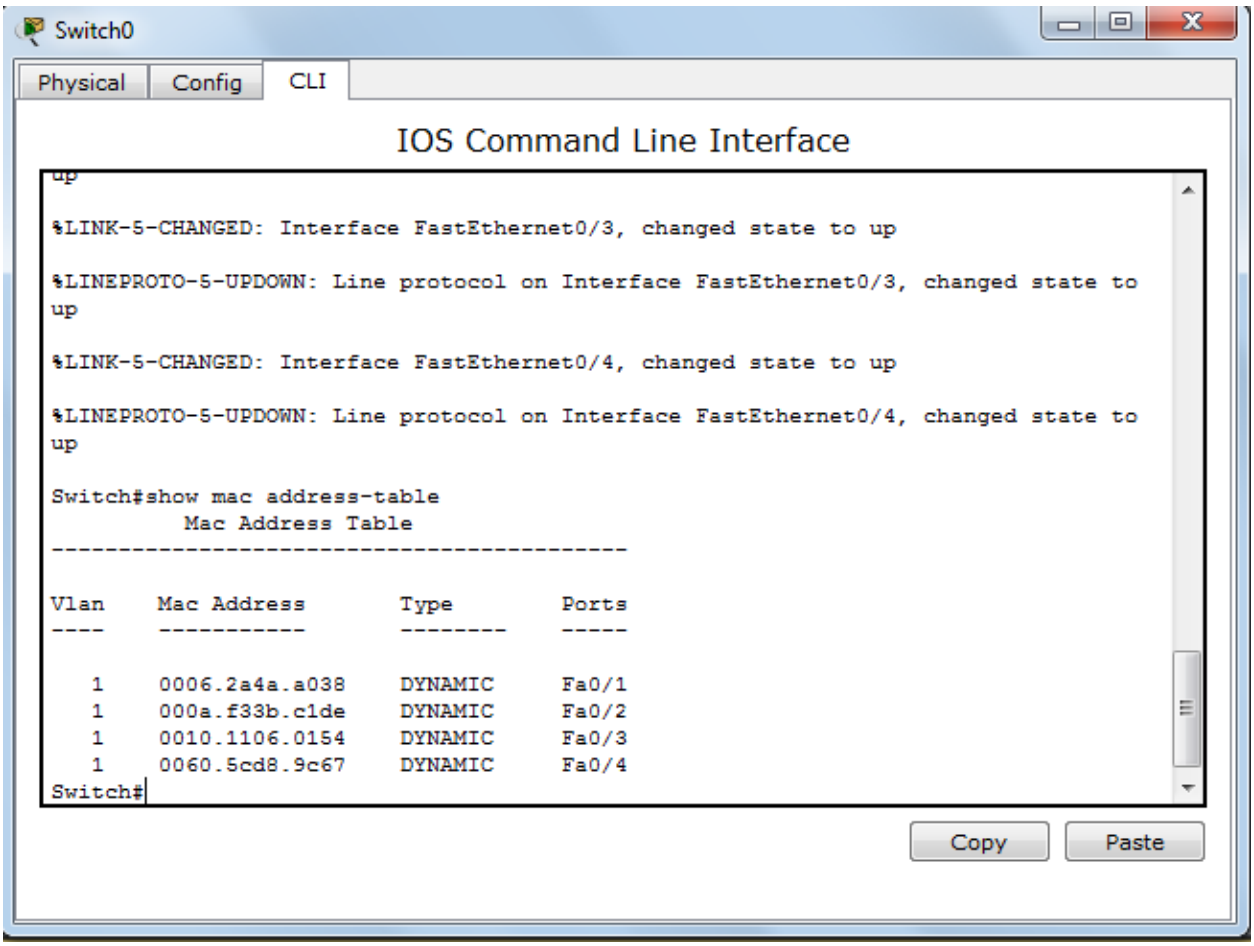

Рисунок 3.18 – Результат выполнения команды show mac address-table после обмена кадрами

Как видно, таблица изменилась. В первом ее столбце указан номер VLAN, MAC-адрес рабочей станции, тип записи (динамическая – Dynamic), и номер интерфейса коммутатора.

3.4 Конфигурирование виртуальных сетей

Технология VLAN была рассмотрена в параграфе 2.5, поэтому в этом параграфе остановимся только на особенностях конфигурирования.

Рассмотрим сначала создание двух VLAN на одном коммутаторе. Для этого по-прежнему будем использовать сеть, представленную на рисунке 3.10. Перейдем в привилегированный режим, выполнив команду enable, и просмотрим информацию о существующих на коммутаторе VLAN (рисунок 3.19), используя для этого команду **show vlan brief**.

В результате выполнения команды на экране появятся: номера VLAN – первый столбец, название VLAN - второй столбец, состояние VLAN (активна она в данный момент или нет) – третий столбец, порты принадлежащие к данной VLAN – четвертый столбец. Как видно из таблицы, по умолчанию на коммутаторе существует пять VLAN . Все порты коммутатора по умолчанию принадлежат VLAN 1. Остальные четыре VLAN не относятся к Ethernet, поэтому рассматривать их здесь не будем.

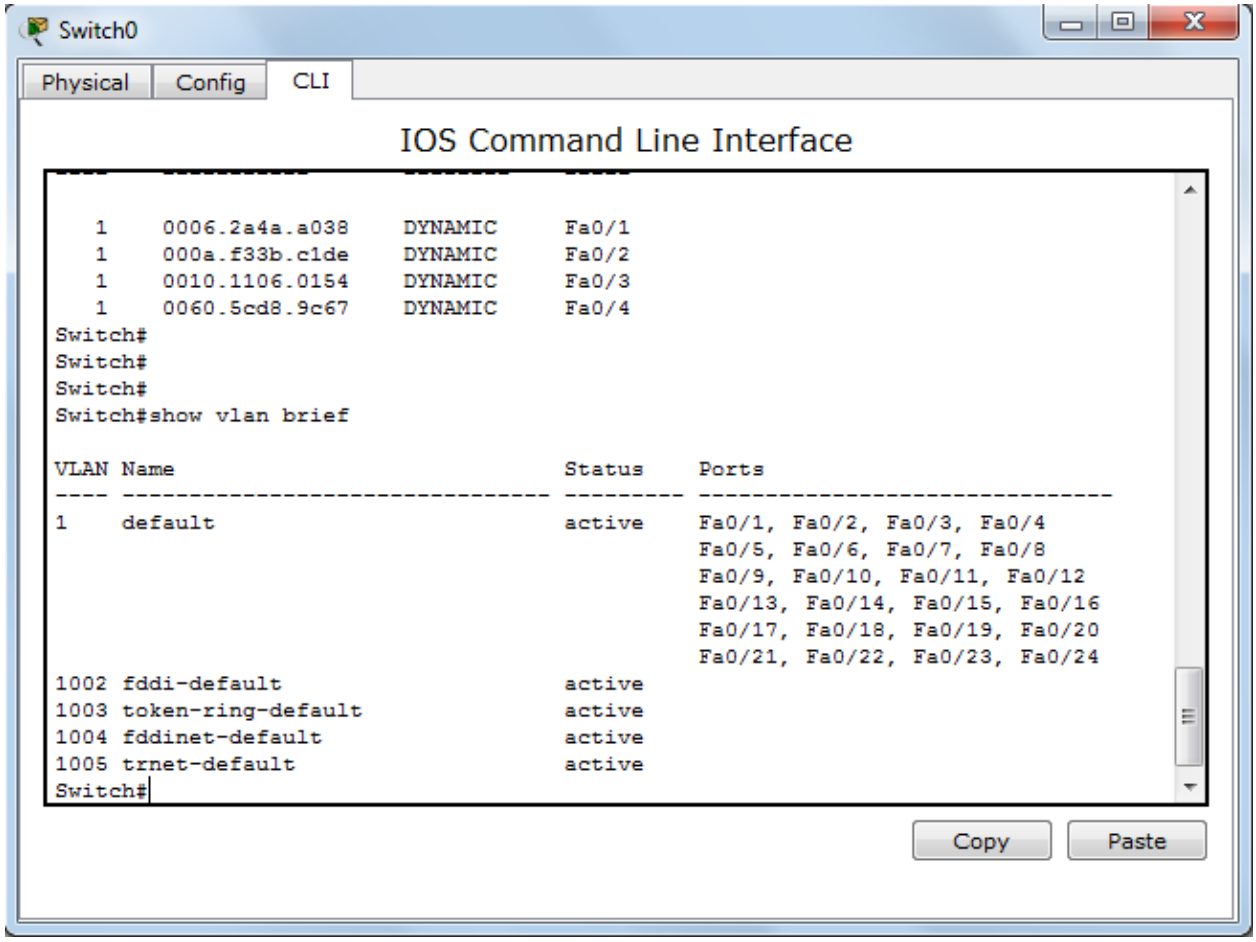

Рисунок 3.19 – Просмотр конфигурации VLAN

Для реализации сети, которую мы запланировали создать, создадим на коммутаторе еще две VLAN. Для этого в привилегированном режиме необходимо выполнить команду **conf t** для перехода в режим глобального конфигурирования. Вводим команду **vlan 2**. Данной командой создается на коммутаторе VLAN с номером 2. Указатель ввода Switch(config)# изменится

на Switch(config-vlan)#, это свидетельствует о том, что конфигурируется уже не весь коммутатор в целом, а только отдельная VLAN, в данном случае номер 2. Если использовать команду **vlan x**, где x номер VLAN, когда VLAN x еще не создана на коммутаторе, то она будет автоматически создана и будет осуществлен переход к ее конфигурированию.

Для решения поставленной задачи коммутатор необходимо сконфигурировать следующим образом.

Switch(config)#vlan 2

Switch(config-vlan)#name subnet_192

Switch(config)#interface range fastEthernet 0/1-2

Switch(config-if-range)#switchport mode access

Switch(config-if-range)#switchport access vlan 2

Разберем приведенные команды. Как уже говорилось ранее, командой **vlan 2** мы создаем на коммутаторе новую VLAN с номером 2. **Команда name subnet_192** присваивает имя subnet_192 виртуальной сети номер 2. Выполняя команду **interface range fastEthernet 0/1-2**, мы переходим к конфигурированию интерфейсов fastEthernet 0/1 и fastEthernet 0/2 коммутатора. Ключевое слово **range** в данной команде указывает на то, что мы будем конфигурировать не один единственный порт, а целый диапазон портов, в принципе ее можно не использовать, но тогда последние три строки придется заменить на следующие:

Switch(config)#interface fastEthernet 0/1

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 2

Switch(config)#interface fastEthernet 0/2

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 2

Команда **switchport mode access** конфигурирует выбранный порт коммутатора как порт доступа (в принципе, эту команду можно не использовать, так как по умолчанию порты коммутатора находятся в режиме

доступа). Команда **switchport access vlan 2** привязывает данный порт к VLAN номер 2.

Как и в предыдущих примерах, команды можно набирать сокращенно, кроме того, вместо **fastEthernet** можно использовать обозначение **fa**.

Просмотрим результат конфигурирования, выполнив команду **show vlan br** после выполнения приведенных выше команд, рисунок 3.20.

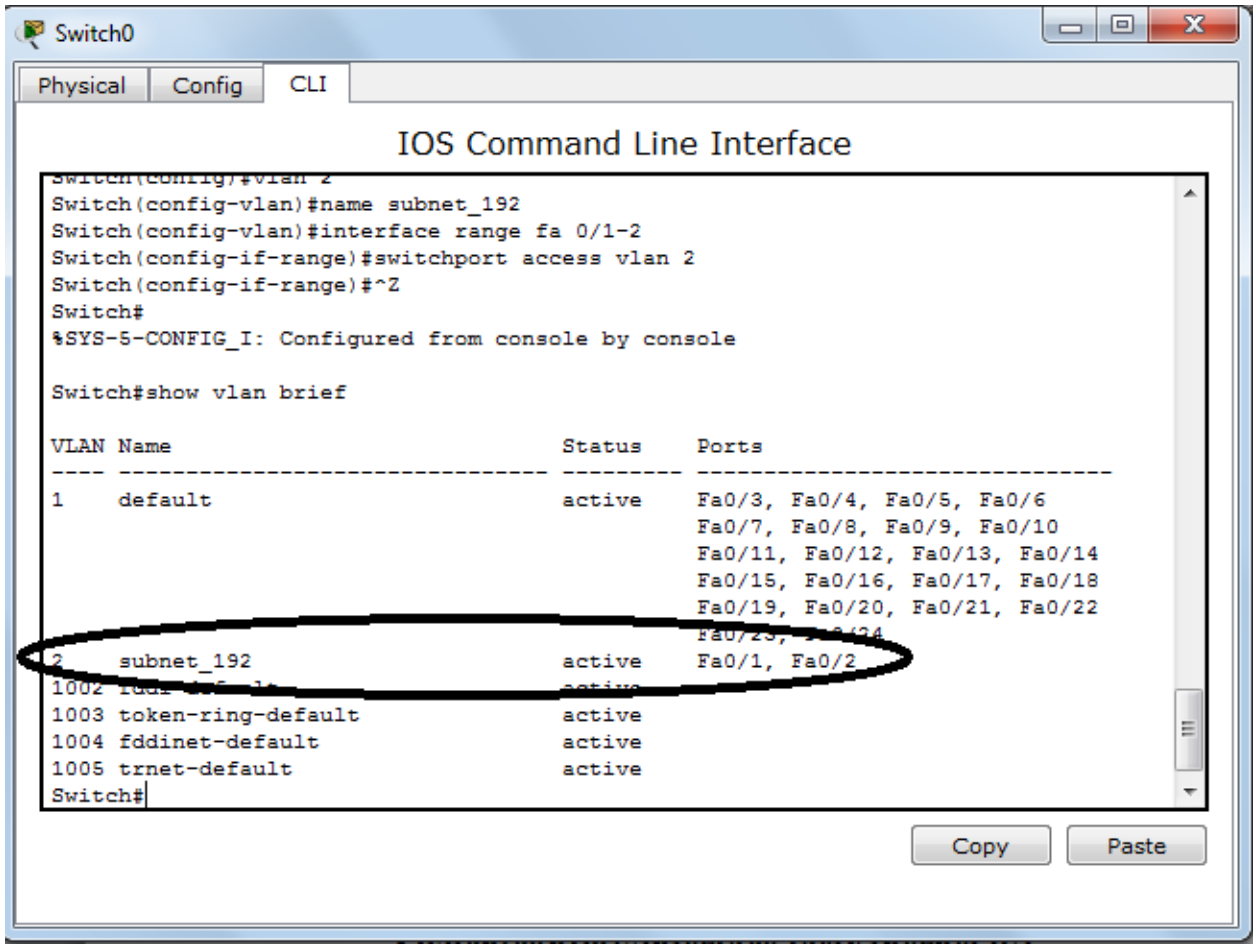

Рисунок 3.20 – Просмотр созданной VLAN 2 и привязанных к ней портов

Из рисунка видно, что в коммутаторе появилась вторая VLAN с именем subnet_192, к которой относятся порты fa 0/1 и 0/2 (выделены на рисунке).

Далее аналогичным образом создадим vlan 3 с именем subnet_172, и привяжем к ней интерфейсы fastEthernet 0/3 и fastEthernet 0/4.

Соответственно, компьютеры, находящиеся в разных виртуальных сетях, будут недоступны друг другу, что легко проверить с использованием утилиты **ping**.

Наибольшую практическую ценность, как было показано в параграфе 2.5, представляет конфигурирование VLAN на нескольких коммутаторах с использованием тэгированных портов. Рассмотрим конфигурирование коммутаторов в этом случае.

Рассмотрим сеть, показанную на рисунке 3.21.

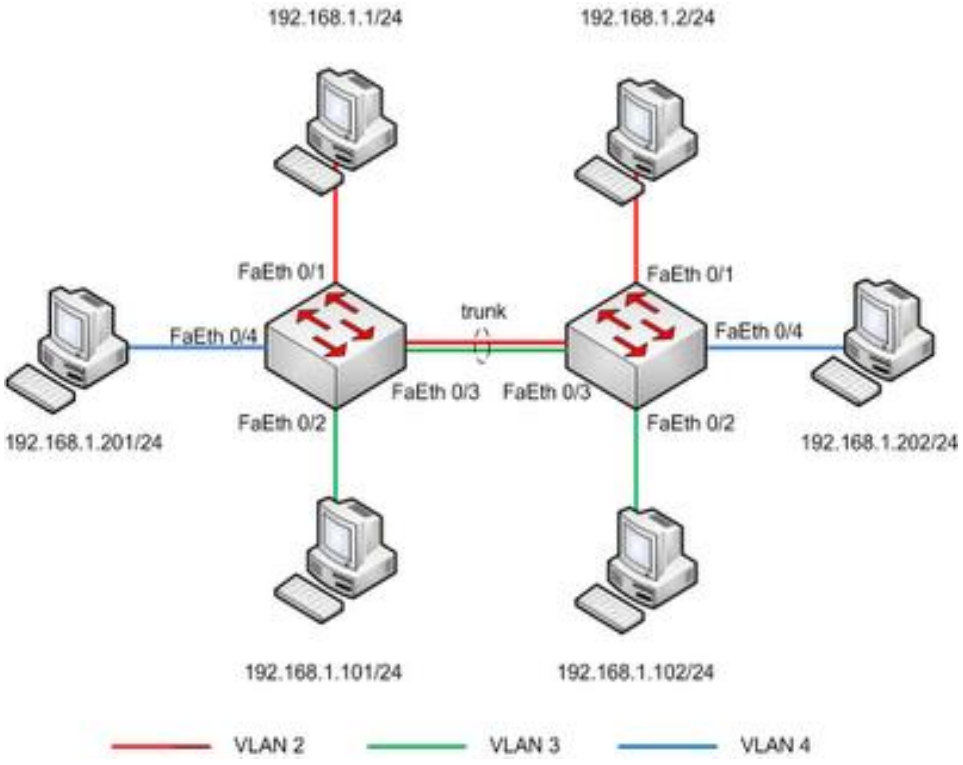

Рисунок 3.21 – Пример сети с двумя коммутаторами

По аналогии с предыдущим примером произведем конфигурирование обоих коммутаторов:

Switch(config)#vlan 2

Switch(config-vlan)#name subnet_2

Switch(config-vlan)#int fa 0/1

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 2

Switch(config-if)#exit

Switch(config)#vlan 3

Switch(config-vlan)#name subnet_3

Switch(config-vlan)#int fa 0/2

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 3

Switch(config-if)#exit

Switch(config)#vlan 4

Switch(config-vlan)#name subnet_4

Switch(config-vlan)#int fa 0/4

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 4

Switch(config-if)#exit

Как следует из перечня команд, на обоих коммутаторах созданы три VLAN. Теперь необходимо перевести третьи порты каждого из коммутаторов в тэгированный режим:

Switch(config)#int fa 0/3

Switch(config-if)# switchport mode trunk

В результате трафик всех трех созданных ранее VLAN будет проходить через порт 3.

Используя на интерфейсе команду switchport mode trunk, мы перевели его в тэгированный режим, в котором интерфейс пропускает через себя трафик всех существующх на коммутаторе VLAN, но иногда необходимо передавать через данный интерфейс трафик не всех VLAN, а лишь некоторых. Для этого на обоих коммутаторов выполним команды:

Switch(config)#interface fastEthernet 0/3

Switch(config-if)#switchport trunk allowed vlan 2-3

Команда **switchport trunk allowed vlan 2-3** указывает тэгированному порту коммутатора, трафик каких VLAN ему пропускать через себя (в нашем случае второго и третьего). После того, как будет выполнена эта команда,

компьютер PC4 должен перестать видеть компьютер PC5. Команда **switchport trunk allowed vlan** при своем использовании каждый раз задает разрешенные порты заново, то есть если выполнить команду **switchport trunk allowed vlan 5**, а потом выполнить команду **switchport trunk allowed vlan 6**, то разрешенным окажется только трафик VLAN номер 6. Для удаления VLAN из списка разрешенных используется команда **switchport trunk allowed vlan remove x**, где x номер удаляемой VLAN. Для просмотра информации о настроенных на коммутаторе магистральных портах служит команда **show int trunk**.

Таким образом, с использованием коммутаторов второго уровня единую сеть можно разделить на виртуальные сети с изолированным друг от друга трафиком. Однако на практике часто возникают задачи гибкого объединения нескольких VLAN между собой. Эту задачу решают маршрутизаторы и коммутаторы третьего уровня, которые будут рассмотрены ниже.

4 Технологии сетевого уровня

4.1 Ограничения сетей на коммутаторах второго уровня

С использованием рассмотренных выше коммутаторов второго уровня можно строить достаточно крупные и разветвленные сети. Однако такие сети имеют ряд ограничений.

Во-первых, в топологии сети на коммутаторах должны отсутствовать так называемые «петли», о чем уже речь шла выше.

Для борьбы с петлевидными соединениями используется алгоритм покрывающего дерева (STA – Spanning Tree Algorithm) и реализующий его протокол покрывающего дерева (STP – Spanning Tree Protocol). В результате действий, определяемых этим протоколом, часть портов коммутатора переводится в заблокированное состояние, что исключает петли. При этом пользовательский трафик через данные порты не передается. С учетом этого распараллелить передачу данных не представляется возможным.

Во-вторых, в сетях на коммутаторах неудобная система адресации – MAC-адреса являются плоскими, и задаются не администратором сети, а производителем оборудования. Соответственно, в проектируемой сети их можно использовать, но нельзя назначать.

В-третьих, логические сегменты сети слабо изолированы друг от друга – широковещательные кадры передаются на все порты коммутатора, что может привести к широковещательному шторму. Отказаться от широковещательных кадров нельзя – на них основаны многие протоколы. Виртуальные сети полностью изолируют трафик друг от друга, что не всегда удобно.

В-четвертых, возможностью трансляции протоколов канального уровня обладают далеко не все типы мостов и коммутаторов, к тому же эти возможности ограничены. В частности, в объединяемых сетях должны совпадать максимально допустимые размеры полей данных в кадрах, так как

мостами и коммутаторами не поддерживается функция фрагментации кадров.

Наличие серьезных ограничений у протоколов канального уровня показывает, что построение на основе средств этого уровня больших неоднородных сетей является весьма проблематичным. Естественное решение в этих случаях – это привлечение средств более высокого, сетевого уровня.

4.2 Маршрутизация пакетов в составной сети

Сначала необходимо отметить, что понятия маршрутизация и коммутация являются, по сути, синонимами. И в том, и в другом случаях под этими терминами понимаются процессы распределения поступающей информации. Однако термин коммутация применяется в случае, когда речь идет о распределении информации на канальном уровне, то есть коммутатор распределяет поступающие кадры, например, кадры Ethernet. Термин же маршрутизация применяется, когда речь идет о сетевом уровне, то есть маршрутизатор распределяет поступающие пакеты, например, IP-пакеты. Соответственно, маршрутизатор должен «уметь» упаковывать (инкапсулировать) пакеты в кадры различных технологий канального уровня (Token Ring, FDDI, Frame Relay, HDLC, и т.д.).

В соответствии с этим, с точки зрения сетевого уровня единая составная сеть представляет собой совокупность подсетей (которые могут строиться на различных технологиях канального уровня), и соединяющих их устройств – маршрутизаторов. При рассмотрении этой сети с «высоты» сетевого уровня структура каждой из подсетей уже не имеет значения, внутри подсети работают протоколы канального уровня, например, Ethernet, рассмотренный выше. Структура такой составной сети представлена на рисунке 4.1.

На рисунке 4.1 подсети обозначены как S1 – S5, маршрутизаторы – как R1 – R5. Маршрутизатор имеет несколько портов (как минимум, два),

каждый из которых входит в свою подсеть. В соответствии с этим каждый порт должен иметь собственный адрес сетевого уровня, а также адрес канального уровня той подсети, в которую он включен. Сетевой адрес первого порта маршрутизатора R1 условно обозначен как R1-1, второго – R1-2, и т.д.

В отличие от адресов канального уровня, сетевая адресация должна быть единой в рамках всей составной сети. Анализируя адреса сетевого уровня в принимаемых пакетах, маршрутизаторы определяют ту подсеть, в которую этот пакет должен быть перенаправлен.

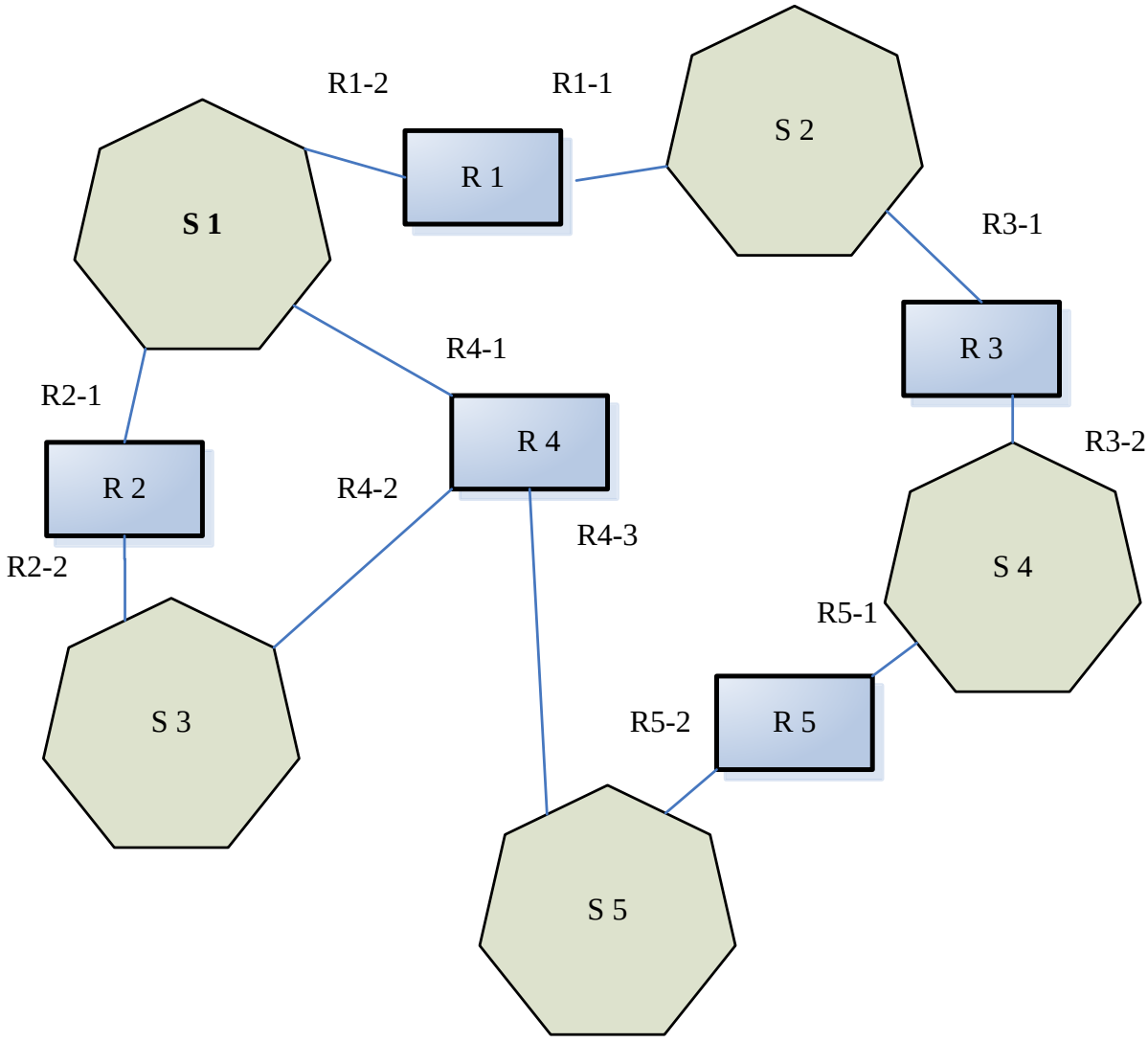

Рисунок 4.1 – Структура составной сети

При этом в пределах подсети пакет транспортируется в составе кадра канального уровня той технологии, на основе которой построена подсеть.

Для определенности будем считать, что все представленные на рисунке подсети построены на базе технологии Ethernet, то есть передача кадров в пределах подсети осуществляется с использованием рассмотренных выше коммутаторов на базе MAC-адресов. Считаем также, что узел-источник (назовем его узлом A) находится в подсети S2, узел-получатель (узел B) – в подсети S3.

Узел A формирует пакет, в который помещает заголовок сетевого уровня, в который, помимо всего прочего, помещается сетевой адрес узла B. Для перемещения пакета в пределах подсети S2 он помещается (инкапсулируется) в кадр Ethernet. В качестве MAC-адреса получателя кадра (DA) указывается адрес порта маршрутизатора, входящего в подсеть S2, например, порта 1 маршрутизатора R1 (R1-1).

Маршрутизатор R1 извлекает (декапсулирует) пакет из принятого кадра, анализирует сетевой адрес узла B и принимает решение, на какой из своих портов этот пакет продвинуть. Если, например, пакет должен быть передан через подсеть S1, маршрутизатор R1 передает пакет на свой порт R1-2, где он опять инкапсулируется в кадр и передается по подсети S1 к маршрутизатору R2. Аналогично поступает маршрутизатор R2, и пакет оказывается в подсети S3 в составе кадра Ethernet. Кадр достигает узла B, находящегося в подсети S3, и узел извлекает пакет из принятого кадра. Таким образом, по мере продвижения пакета по составной сети каждый маршрутизатор декапсулирует его из кадра, определяет, куда его передать дальше, заново инкапсулирует в кадр и передает. Соответственно, адреса канального уровня (в нашем случае MAC-адреса) меняются в каждой подсети, через которую проходит пакет, а адреса сетевого уровня остаются неизменными.

Следует отметить, что описанная здесь маршрутизация является одношаговой. Это означает, что каждый из маршрутизаторов, предав пакет,

«забывает» о нем. Кроме того, в составной сети, как правило, существует несколько маршрутов, ведущих от источника к приемнику. Решение о том, куда именно передать пакет дальше, маршрутизатор принимает исходя из значения так называемой метрики, которая будет рассмотрена ниже. Если какой-либо маршрут отказывает, пакет передается по другому маршруту, причем изменение маршрута должно быть произведено автоматически, без участия человека.

Остается выяснить вопрос, каким образом маршрутизатор определяет направление дальнейшей передачи пакета. Этот вопрос рассмотрим в следующем параграфе.

4.3 Принципы маршрутизации

Очевидно, что маршрутизатор для определения пути дальнейшей передачи принятого пакета должен обладать некоторой дополнительной информацией, примерно так же, как и коммутатор определял нужный выходной порт в соответствии со своей адресной таблицей. Аналогичная таблица имеется и у маршрутизатора, и называется она таблицей маршрутизации. Однако в отличие от адресной таблицы она содержит большее количество полей. Кроме того, построение таблиц маршрутизации в корне отличается от построения адресных таблиц. Наконец, в отличие от коммутации, в маршрутизации принимают участие не только маршрутизаторы, но и конечные узлы.

Во второй главе было показано, что динамические записи в адресную таблицу коммутатор вносит, наблюдая за проходящим через него трафиком. Причем наблюдение это пассивное, коммутатор просто отслеживает значение поля SA в принятых кадрах и при необходимости корректирует записи в таблице. Маршрутизаторы же обмениваются служебной информацией (пакетами), и каждый из них, приняв данные от «соседа», формирует свою таблицу маршрутизации. Содержимое этих служебных сообщений зависит от протокола маршрутизации, который реализуется на

том или ином маршрутизаторе. Современные маршрутизаторы способны одновременно поддерживать несколько протоколов маршрутизации.

Что касается статических записей, то они могут быть внесены вручную аналогично статическим записям в адресных таблицах коммутаторов.

Поля таблицы тоже могут быть различны у маршрутизаторов разных производителей. В самом общем случае таблица маршрутизации, например, маршрутизатора R2 (рисунок 4.1) имеет вид, представленный в таблице 4.1.

Таблица 4.1 – Таблица маршрутизации маршрутизатора R2

Адрес сети	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
S1	-	R2-1	0
S2	R1-2	R2-1	1
S3	-	R2-2	0
S4	R1-2	R2-1	2
S5	R4-1	R2-1	1
S5	R4-2	R2-2	1

В первом столбце таблицы содержится адрес сети назначения. Из этого непосредственно следует, что сетевой адрес, в отличие от канального, не должен быть плоским, а должен делиться, как минимум, на две части – адрес подсети и адрес узла в этой подсети. Как это реализуется на практике, будет рассмотрено ниже.

Второй столбец содержит сетевые адреса портов следующих маршрутизаторов, которым должен быть передан пакет. Если пакет адресуется узлу, находящемуся в подсети, к которой маршрутизатор подключен непосредственно, на следующий маршрутизатор его передавать не нужно. Это относится к сетям S1 и S3, соответственно напротив адресов этих сетей в данном столбце поставлен прочерк.

Третий столбец указывает на собственный порт маршрутизатора, с которого должен быть передан пакет.

В четвертом столбце расположена метрика – величина, определяющая оптимальность того или иного маршрута. В данном примере в качестве метрики используется количество транзитных маршрутизаторов, через которые должен быть передан пакет, чтобы достигнуть подсети назначения. Соответственно, если маршрутизатор непосредственно подключен к подсети назначения, метрика равна нулю (подсети S1 и S3).

Данная метрика – наиболее простой и не самый лучший случай. Например, ее использует наиболее «старый» протокол маршрутизации – Routing Information Protocol (RIP). В качестве метрики более современные протоколы могут использовать пропускную способность маршрута, задержку передачи, и т.д. Общее правило остается одно – чем меньше значение метрики, тем лучше маршрут.

В нашем примере маршруты к сети S5 имеют одинаковые метрики, то есть, например, с точки зрения протокола RIP они являются равнозначными. Однако с точки зрения других протоколов эти маршруты могут иметь разные метрики, соответственно, для передачи будет использован оптимальный маршрут, то есть маршрут с наименьшей метрикой.

Очевидно, что если сеть небольшая, как в нашем примере, то в таблице можно прописать маршруты ко всем имеющимся подсетям. Однако в крупных сетях это может привести к чрезмерно большим размерностям таблицы, вследствие чего используется запись, называемая «маршрутом по умолчанию».

Маршрут по умолчанию – это маршрут, по которому передаются пакеты, адресуемые в подсети, о которых маршрутизатору ничего не известно. Если маршрута по умолчанию в таблице нет, маршрутизатор пакеты с неизвестным адресом назначения просто отбрасывает.

Например, если предположить, что к подсети S3 подключен еще один маршрутизатор – R6 – а к нему, в свою очередь, подключены подсети, не

показанные на рисунке 4.1, то очевидно, что все пакеты, адресуемые в эти подсети, необходимо продвигать через маршрутизатор R6. Тогда в таблице 4.1 появляется еще одна строка:

Адрес сети	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
Default	R6-1	R2-2	-

Как указывалось выше, в маршрутизации пакетов участвуют не только маршрутизаторы, но и конечные узлы. Соответственно, конечные узлы также имеют свою таблицу маршрутизации. В отличие от маршрутизаторов, эти таблицы, как правило, имеют существенно меньший объем и гораздо чаще используют маршруты по умолчанию. Это связано с тем, что конечные узлы, как правило, находятся на периферии сети.

Например, таблица маршрутизации узла А, находящегося в подсети S2, могла бы выглядеть так, как это показано в таблице 4.2.

Таблица 4.2 – Таблица маршрутизации узла А

Адрес сети	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
S4	R3-1	A-1	1
Default	R1-1	A-1	-

Через А-1 здесь обозначен порт узла А, пакеты, предназначенные для сети S4, будут перенаправляться через маршрутизатор R3, все остальные пакеты – через маршрутизатор R1.

Реальные таблицы маршрутизации имеют несколько другой вид, определяемый, главным образом, операционными системами узлов и маршрутизаторов, а также реализуемыми ими протоколами. Таблицы маршрутизации устройств Cisco и ОС Windows будут рассмотрены ниже.

4.4 Классы IP-адресов

В предыдущем параграфе были рассмотрены основные принципы продвижения пакетов через составную сеть независимо от конкретных протоколов, на которых она построена. В первой главе отмечалось, что основным стеком протоколов, на котором базируются современные сети, является стек TCP/IP. Поэтому необходимо конкретизировать положения предыдущего параграфа с учетом того, что сеть функционирует в соответствии с названным стеком протоколов.

В стеке TCP/IP имеются три типа адресов – локальные, IP-адреса и символьные доменные имена [3].

В терминологии TCP/IP под локальным адресом понимается такой тип адреса, который используется средствами канальной технологии для доставки данных в пределах подсети, являющейся элементом составной сети. В разных подсетях допустимы разные технологии, разные стеки протоколов, поэтому при создании стека TCP/IP предполагалось наличие разных типов локальных адресов. Если подсетью интерсети является локальная сеть Ethernet, то локальный адрес - это MAC-адрес. MAC-адрес назначается сетевым адаптерам и сетевым интерфейсам маршрутизаторов производителями оборудования и является уникальным, так как управляется централизованно.

IP-адреса представляют собой основной тип адресов, на основании которых сетевой уровень передает пакеты между сетями. Эти адреса состоят из четырех байт (в четвертой версии протокола – IPv4), которые для удобства использования человеком записываются в десятичном формате и разделяются точками, например 109.26.17.100. Сетевое оборудование оперирует IP-адресом, как правило, двоичного формата. IP-адрес назначается администратором во время конфигурирования компьютеров и маршрутизаторов, или назначается динамически с использованием специально созданного для этого протокола – DHCP, который будет

рассмотрен ниже. IP-адрес состоит из двух частей: номера сети и номера узла.

Символьные имена в IP-сетях называются доменными и строятся по иерархическому принципу. Составляющие полного символьного имени в IP-сетях разделяются точкой и перечисляются в следующем порядке: сначала простое имя конечного узла, затем имя группы узлов (например, имя организации), затем имя более крупной группы (поддомена) и так до имени домена самого высокого уровня (например, домена объединяющего организации по географическому принципу: RU - Россия, UK - Великобритания, SU - США). Примером доменного имени может служить имя skf-mtusi.ru. Между доменным именем и IP-адресом узла нет никакого алгоритмического соответствия, поэтому необходимо использовать какие-то дополнительные таблицы или службы, чтобы узел сети однозначно определялся как по доменному имени, так и по IP-адресу. В сетях TCP/IP используется специальная распределенная служба Domain Name System (DNS), которая устанавливает это соответствие на основании создаваемых администраторами сети таблиц соответствия. Поэтому доменные имена называют также DNS-именами.

Вернемся к рассмотрению IP-адресов. Выше было показано, что в отличие от плоского MAC-адреса сетевой адрес должен быть иерархическим, то есть, как минимум, состоять из двух частей – адреса сети (подсети) и адреса узла. Так как IP-адрес относится к сетевому уровню, необходимо определиться, какая его часть будет относиться к сети, а какая – к узлу.

IP-адреса четвертой версии (IPv4) состоят из четырех байт. Можно, напрмер, раз и навсегда определить, что первый байт – это адрес сети, а остальные – адрес узла. Тогда во всем мире не может насчитываться более, чем $2^8 = 256$ сетей (так как в байте 8 бит). Зато возможное число узлов в каждой из сетей окажется значительным – 2^{24}.

Очевидно, что такая ситуация является неприемлемой – число уже созданных и функционирующих сетей гораздо больше 256, кроме того, большинству сетей не нужно такое большое число конечных узлов.

Если же отвести по два байта на адресацию сети и узла, то число сетей и узлов в каждой из них составит $2^{16} = 65536$. Понятно, что сетей во всем мире все равно больше, соответственно, адресов на всех не хватит. Кроме того, далеко не все сети включают в себя 65536 узлов, в большинстве сетей их гораздо меньше. Кроме того, существуют крупные сети, с числом узлов большим, чем 65536, и в этом случае такая адресация им не подойдет.

Из всего вышесказанного можно сделать вывод о том, что «граница» между адресом сети и адресом узла должна быть подвижной. Для крупных сетей, которых в мире немного, можно отвести на адресацию сети один байт. Тогда три оставшихся байта обеспечат в такой сети большое количество узлов. Средним сетям, которых гораздо больше, подходит схема, в которой первые два байта – это адрес сети, последние два байта остаются для адресации узлов. И, наконец, для небольших сетей, число которых очень велико, было бы разумно выделить три байта под адрес сети, и один – под адресацию узлов.

Таким образом, IP-адреса являются неравнозначными и делятся на так называемые классы. Структура IP-адреса класса A представлена на рисунке 4.2, заимствованном из [8].

<table>
<tr><td>0</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>2-й байт</td><td>3-й байт</td><td>4-й байт</td></tr>
<tr><td colspan="8">№ сети – 1 байт</td><td colspan="3">№ узла – 3 байта</td></tr>
</table>

Рисунок 4.2 – IP-адрес класса A

IP-адрес класса B представлен на рисунке 4.3.

<table>
<tr><td>1</td><td>0</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>2-й байт</td><td>3-й байт</td><td>4-й байт</td></tr>
<tr><td colspan="8">№ сети – 2 байта</td><td colspan="3">№ узла – 2 байта</td></tr>
</table>

Рисунок 4.3 – IP-адрес класса B

IP-адрес класса C представлен на рисунке 4.4.

1	1	0	x	x	x	x	x	2-й байт	3-й байт	4-й байт
№ сети – 3 байта										**№ узла – 1 байт**

Рисунок 4.4 – IP-адрес класса C

Из рисунков 4.2 – 4.4 видно, что в этих адресах первые разряды первого байта имеют определенные значения – 0 для класса A, 10 для класса B и 110 для класса C. Это сделано для того, чтобы маршрутизатор мог гарантированно определить к какому классу относится адрес, и ограничивает диапазон адресов сетей в пределах одного класса. Например, первый байт адреса класса B начинается с 10, следовательно, самое маленькое значение этого байта будет 10000000 или 128 в десятичной системе счисления. Самое большое значение этого байта составит 10111111 или 191 в десятичном формате. Это же касается и остальных классов. Таким образом, каждый из классов занимает определенный диапазон адресов, что иллюстрируется таблицей 4.3.

Таблица 4.3 – Характеристика классов IP-адресов

Класс	Первый байт	Наименьший адрес сети	Наибольший адрес сети	Число узлов
A	0xxxxxxx	1.0.0.0	126.0.0.0	2^{24}
B	10xxxxxx	128.0.0.0	191.255.0.0	2^{16}
C	110xxxxx	192.0.0.0	223.255.255.0	2^{8}

Существуют еще классы D и E, однако непосредственного отношения к коммутации они не имеют, поэтому рассматривать их здесь не будем.

Следует отметить, что на практике число узлов в каждом из классов уменьшается на 2, так как этот адрес не может состоять из одних нулей и одних единиц.

Маршрутизатор, работающий на основе классов, в своей таблице содержит адреса не узлов, а сетей (так как его основная задача – передать

пакет в нужную сеть). Соответственно, при прибытии очередного пакета он по первым битам адреса определяет класс адреса, отделяет адрес сети от адреса узла, и отыскивает адрес сети в таблице маршрутизации.

Таким образом, различным по размеру сетям присваиваются адреса различных классов, что в принципе решает проблему, о которой говорилось в начале этого параграфа. Однако такое жесткое деление на классы имеет недостатки, которые рассмотрим в следующем параграфе.

4.5 Использование масок в IP-адресации

Рассмотрим сначала недостатки, присущие делению адресов на классы.

Во-первых, все возможные сети могут быть классифицированы только на три вида – крупные (с адресами класса A), средние (с адресами класса B) и малые (с адресами класса C), никаких промежуточных видов сетей не предусматривается. Это приводит к тому, что крупные сети уже «заняли» все адреса класса A, а в малых сетях адресное пространство узлов зачастую недоиспользуется. Кроме того, на практике может возникнуть случай, когда предприятию, проектирующему для себя IP-сеть, может немного не хватить адресного пространства узлов класса C, а если выделить ему адрес класса B, то у него в запасе останется большое количество адресов узлов, которые использоваться не будут.

Во-вторых, уже сейчас наблюдается дефицит IP-адресов классов A и B. Кардинальным решением такой проблемы может стать IP-адресация шестой версии (IPv6), но пока еще повсеместно используется четвертая версия.

В-третьих, если какому-либо предприятию или организации провайдер выделил адрес класса C, то вся сеть будет единой, администратор не сможет дополнительно поделить ее на внутренние подсети. Между тем такое деление позволяет упростить администрирование сети, организовать различные права доступа к каким-либо ресурсам, и т.д.

Таким образом, в определенный момент времени возникла задача более гибкой IP-адресации. Эта задача была решена путем разработки технологии маски переменной длины (VLSM – Variable-Length Subnet Mask).

Маской называется число, которое используется в паре с IP-адресом; двоичная запись маски содержит единицы в тех разрядах, которые должны в IP-адресе интерпретироваться как номер сети [14]. Поскольку номер сети является цельной частью адреса, единицы в маске также должны представлять непрерывную последовательность (не должны чередоваться с нулями). Соответственно, с помощью маски в адресе можно выделить произвольное количество разрядов для номера сети, что частично устраняет перечисленные выше недостатки.

Общее правило использования масок в IP-адресации можно сформулировать следующим образом [15].

Для вычисления по адресу и маске номера сети необходимо применить побитовую операцию «И» к адресу и маске.

Для вычисления по адресу и маске номера узла необходимо применить побитовую операцию «И» к адресу и инвертированной маске.

Под инвертированной маской понимается результат применения операции «НЕ» ко всем разрядам маски. Проще говоря, инвертированная маска получается путем замены всех разрядов в маске на противоположные (1 на 0 и 0 на 1).

Приведем примеры.

Предположим, что имеется адрес

192.168.100.5

и маска

255.255.255.0

Представим адрес и маску в двоичном виде:

11000000.10101000.01100100.00000101 - адрес

11111111.11111111.11111111.00000000 - маска.

Для получения адреса сети применим побитовую операцию «И» к адресу и маске:

11000000.10101000.01100100.00000101
11111111.11111111.11111111.00000000
11000000.10101000.01100100.00000000

При переводе результата в десятичный формат получим

192.168.100.0 – номер сети

Для вычисления номера узла применим побитовую операцию «И» к адресу и инвертированной маске

11000000.10101000.01100100.00000101
00000000.00000000.00000000.11111111
00000000.00000000.00000000.00000101

При переводе результата в двоичный формат получим

0.0.0.5 – номер узла.

В данном примере мы фактически использовали стандартный адрес класса С. Этот же адрес можно представить в виде

192.168.100.5/24,

где /24 означает, что под маску отведено 24 разряда.

Приведем пример использования нестандартной маски:

адрес: 215.17.125.177,

маска 255.255.255.240.

Представим адрес и маску в двоичном виде:

11010111.00010001.01111101.10110001 – адрес,

11111111.11111111.11111111.11110000 – маска.

Аналогично предыдущему примеру получим адрес сети:

11010111.00010001.011111101.10110000 – 215.17.125.176

и адрес узла:

00000000.00000000.00000000.00000001 – 0.0.0.1.

Соответственно, маршрутизатор накладывает маску на адрес принятого пакета, определяет адрес сети, находит этот адрес в таблице маршрутизации, и определяет, куда этот пакет необходимо переправить.

Использование маски переменной длины позволяет при проектировании сети разбить ее на нужное количество подсетей. Например, провайдер выделил для сети статический адрес 213.59.30.0/24. Все адреса блока имеют одинаковый префикс, состоящий из 24 разрядов. Если последние 8 разрядов использовать для адресации узлов, то все узлы будут находиться в одной сети. Если администратору необходимо разделить общую сеть на 4 подсети, он может увеличить маску на два разряда. Маска в этом случае будет иметь вид:

11111111.11111111.11111111.11000000 – 255.255.255.192.

Тогда адресация подсетей будет следующая:

подсеть 1 – 11010101.00111011.00011110.00000000 – 213.59.30.0/26

подсеть 2 – 11010101.00111011.00011110.01000000 – 213.59.30.64/26

подсеть 3 – 11010101.00111011.00011110.10000000 – 213.59.30.128/26

подсеть 4 – 11010101.00111011.00011110.11000000 – 213.59.30.192/26.

Соответственно, если «занять» из номера узла 3 разряда, можно организовать $2^3 = 8$ подсетей, если 4 – $2^4 = 16$, и так далее.

В настоящее время имеется достаточно много программных продуктов, позволяющих автоматизировать процесс разделения на подсети, то есть вычислять значения маски. На рисунке 4.5 представлен интерфейс одного из таких продуктов – LAN Calculator.

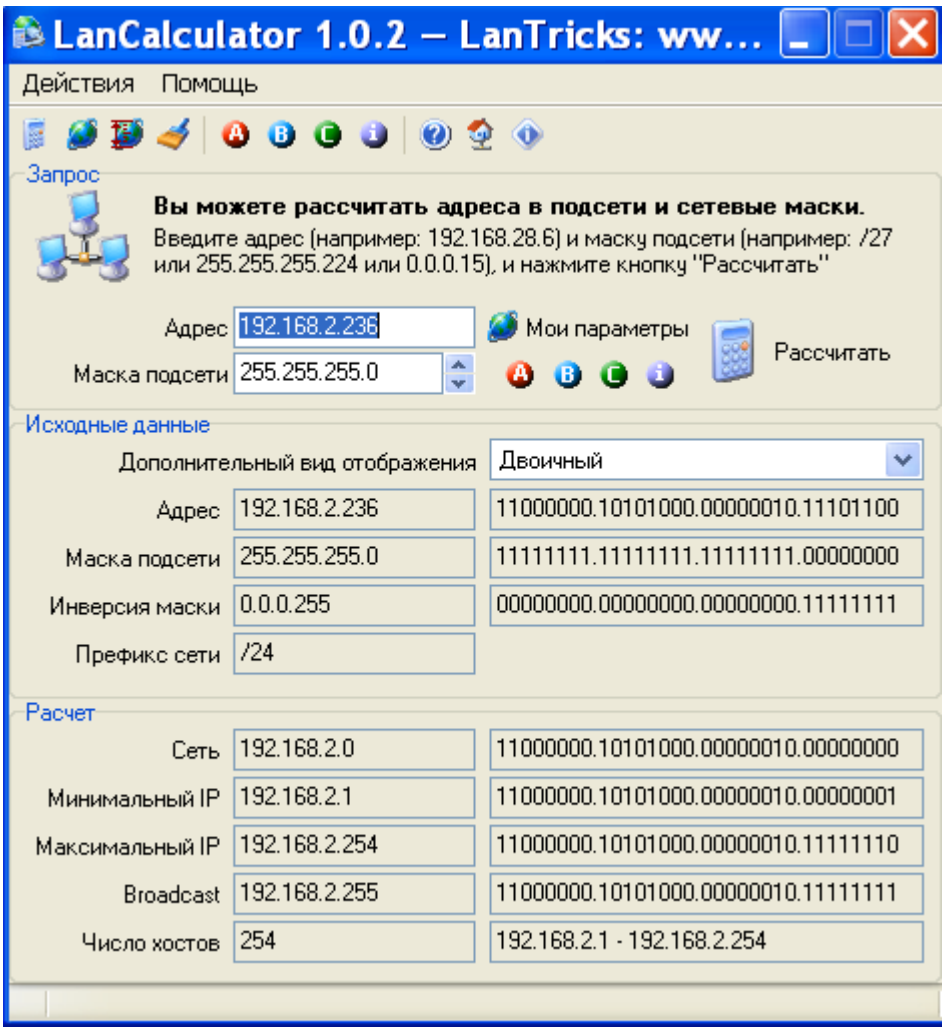

Рисунок 4.5 – Интерфейс программы LAN Calculator

5 Протоколы маршрутизации

5.1 Статическая маршрутизация

При небольшом количестве подсетей, как правило, используется статическая маршрутизация. Рассмотрим в общем виде конфигурирование сети с использованием статической маршрутизации.

Предположим, что структура сети имеет вид, показанный на рисунке 5.1.

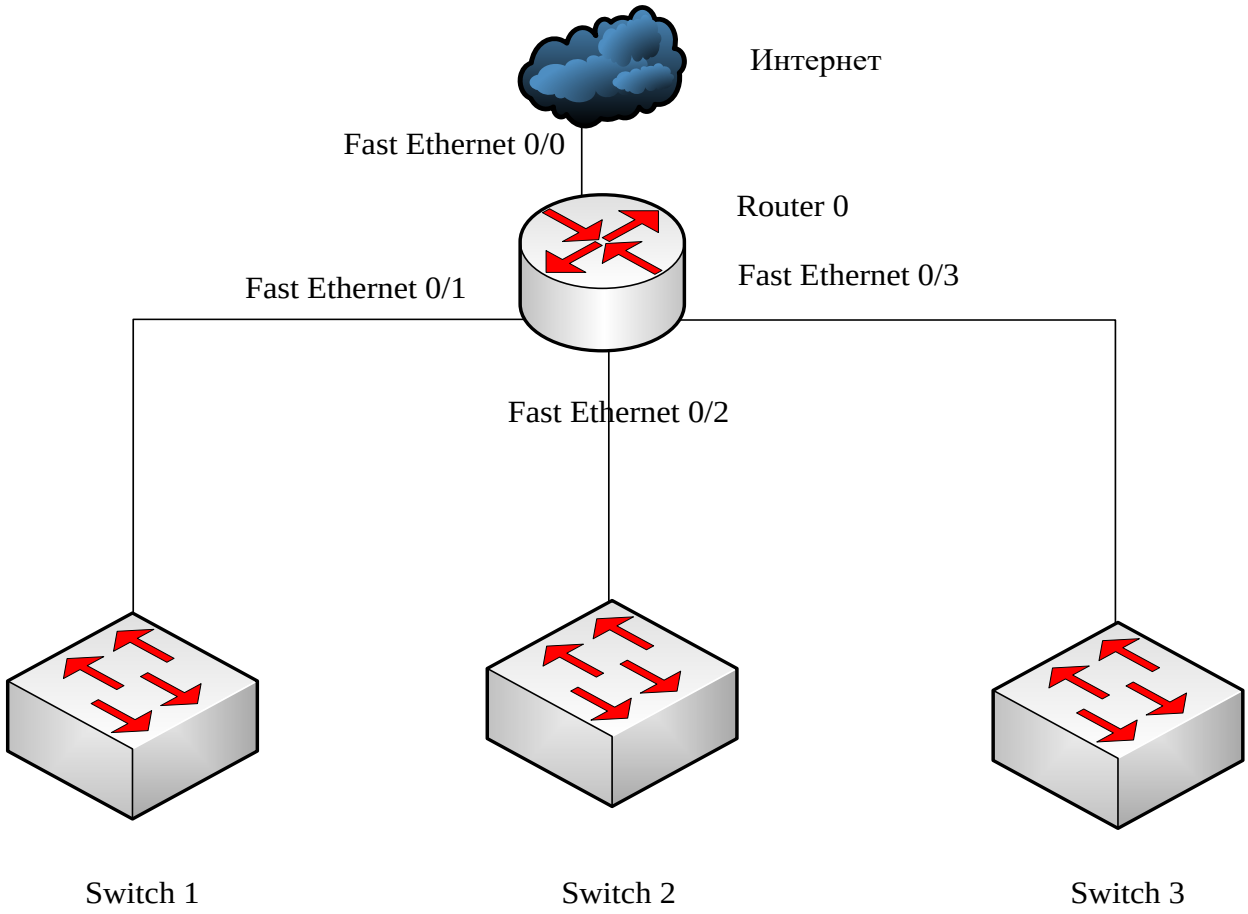

Рисунок 5.1 – Структура сети

Из рисунка видно, что сеть состоит из трех подсетей (это могут быть, например, три отдела предприятия). Разделение на подсети осуществляется с использованием маршрутизатора Router 0, через него же осуществляется доступ к сети Интернет. Каждая подсеть содержит коммутатор второго уровня.

Предположим, что для адресации сети будет использоваться адрес 192.168.1.0/24.

Предположим, что каждая подсеть может содержать до 24 конечных узлов, плюс адрес интерфейса маршрутизатора, плюс два специальных адреса (для номера сети и широковещания), следовательно, под адресацию узлов в каждой подсети необходимо отвести 5 разрядов ($2^5 = 32$). Оставшиеся 3 разряда четвертого байта можно использовать для адресации подсетей. Тогда маска подсети будет иметь вид:

11111111.11111111.11111111.11100000

или в десятичном формате:

255.255.255.224.

Тогда в нашей сети можно выделить $2^3 = 8$ подсетей, из которых используем только три, а остальные можно оставить в резерве для будущего развития сети.

Подсетям назначим следующие адреса:

- 192.168.1.32/27;
- 192.168.1.64/27;
- 192.168.1.96/27.

Для конфигурирования статической маршрутизации в нашем примере портам маршрутизатора необходимо назначить сетевые адреса из диапазона адресного пространства перечисленных выше подсетей. Соответственно, порт Fast Ethernet, входящий в первую подсеть, получает адрес 192.168.1.33/27, во вторую – адрес 192.168.1.65/27, в третью – 192.168.1.97/27.

Компьютерам подсетей также необходимо задать соответствующие сетевые настройки. Этот процесс можно автоматизировать с применением протокола DHCP, который будет рассмотрен ниже, или сконфигурировать конечные узлы вручную. В состав минимальных настроек узла входят: IP-адрес, маска подсети, а также адрес шлюза по умолчанию. В качестве шлюза

по умолчанию в нашем примере для каждой из подсетей будет выступать маршрутизатор Router 0, точнее, его интерфейс, включенный в подсеть.

Например, если конечные узлы работают под управлением ОС Windows, для конфигурирования необходимо зайти во вкладку «Подключение по локальной сети – Свойства» и выбрать пункт «Протокол Интернета (TCP/IP)», рисунок 5.2.

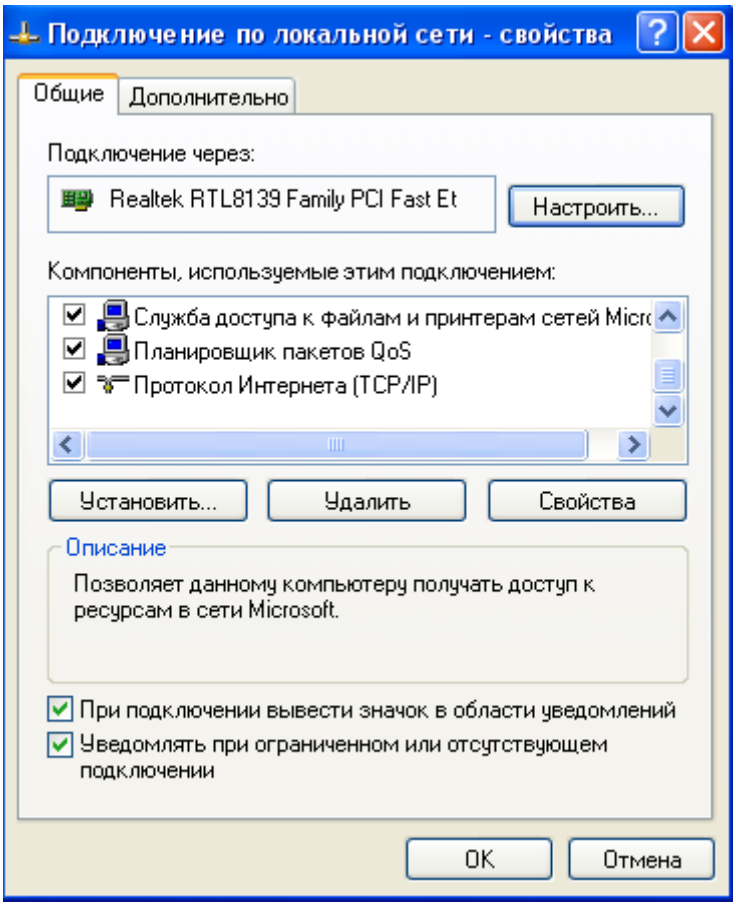

Рисунок 5.2 – Конфигурирование конечного узла

В появившемся окне необходимо выделить пункт «Использовать следующий IP-адрес» и в соответствующие поля внести минимальную конфигурацию, рисунок 5.3.

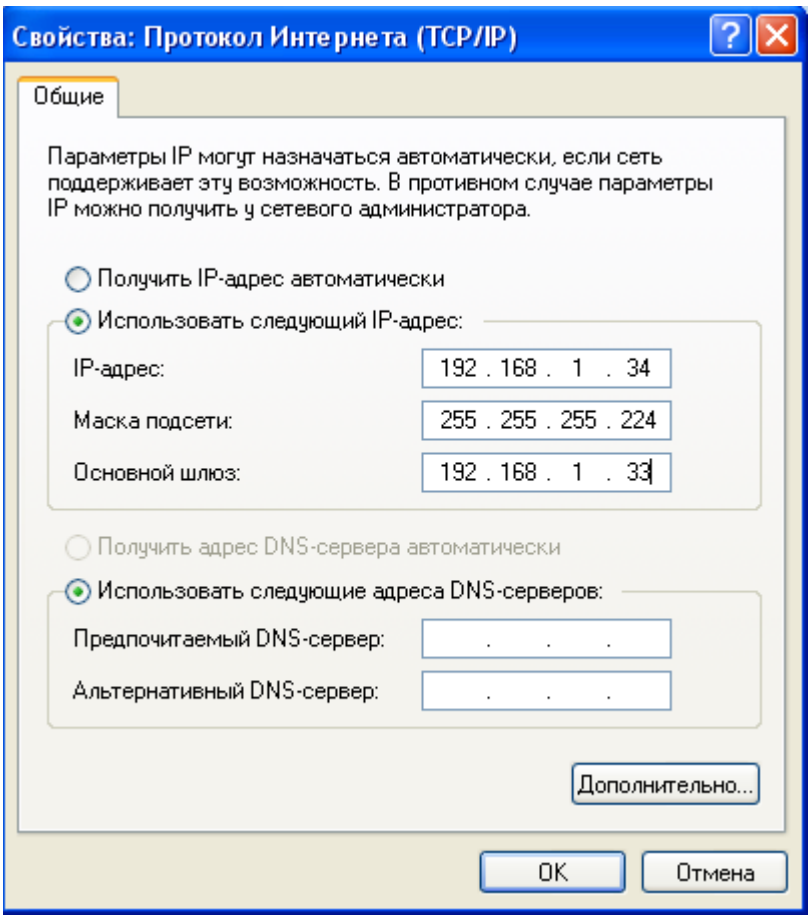

Рисунок 5.3 – Ручная настройка сетевых параметров

Таким образом, при статической маршрутизации необходимо лишь назначить интерфейсам маршрутизатора необходимые адреса.

5.2 Классификация алгоритмов и протоколов динамической маршрутизации

Статическая маршрутизация, рассмотренная выше, может использоваться только в сетях относительно небольшого масштаба. При значительном масштабе сети задача ручного ведения таблиц маршрутизации становится слишком трудоемкой и сопряжена с большим количеством ошибок, допускаемых при конфигурировании сетевых устройств.

Поэтому в настоящее время используются алгоритмы и основанные на них протоколы динамической маршрутизации. Перед рассмотрением

непосредственно самих протоколов маршрутизации целесообразно привести классификацию алгоритмов маршрутизации.

В общем случае под алгоритмом маршрутизации понимается набор правил, регламентирующих процедуры обмена служебной информацией между маршрутизаторами с целью заполнения их таблиц.

Напомним, что здесь рассматривается одношаговая маршрутизация, при которой маршрутизатор пересылает пакет в следующую подсеть и не «заботится» о его дальнейшем продвижении.

Различают следующие виды динамической маршрутизации [3, 14]:

- локальная;
- распределенная;
- централизованная.

При локальной маршрутизации маршрутизатор принимает решение о дальнейшем продвижении пакета только на основании информации о состоянии своих портов и очередей пакетов. Очевидно, что при локальной маршрутизации не учитывается ни расстояние до узла назначения, ни состояние каналов связи, в связи с чем локальная маршрутизация в настоящее время практически не применяется.

При распределенной маршрутизации происходит обмен служебной информацией между маршрутизаторами. Алгоритмы распределенной маршрутизации можно разделить на дистанционно-векторные алгоритмы (Distance Vector Algorithms) и алгоритмы состояния связей (Link State Algorithms).

В дистанционно-векторных алгоритмах каждый маршрутизатор широковещательно распространяет вектор расстояний (метрик) от самого себя до всех известных ему подсетей. В качестве метрики в таких алгоритмах обычно используется количество промежуточных маршрутизаторов, через которые должен пройти пакет, чтобы достигнуть подсети назначения. Именно такой тип метрики был приведен для примера в рассмотренных выше таблицах маршрутизации. Очевидно, что наименьшей метрике

соответствует кратчайшее расстояние до сети назначения, и маршрут с минимальной метрикой считается оптимальным.

Получив такой вектор от своего «соседа», каждый маршрутизатор добавляет к нему сведения обо всех известных ему подсетях (в том числе о тех, к которым он подключен непосредственно, а также о тех, о которых ему стало известно ранее от других «соседей») и снова рассылает вектор по сети.

Таким образом, в результате обмена векторами каждый маршрутизатор в конце концов получит информацию обо всех подсетях, входящих в составную сеть, а также о расстояниях до них.

Алгоритмы состояния связей обеспечивают каждый маршрутизатор информацией, достаточной для построения точного графа связей составной сети. Все маршрутизаторы работают на основании одинаковых графов, что делает процесс маршрутизации более устойчивым к изменениям конфигурации. Широковещательная рассылка используется здесь только при изменениях состояния связей, что происходит в надежных сетях не так часто.

Для того чтобы определить, в каком состоянии находятся линии связи, подключенные к его портам, маршрутизатор периодически обменивается короткими пакетами со своими ближайшими соседями. Этот трафик также широковещательный, но он циркулирует только между соседями и поэтому не так «засоряет» сеть, как при использовании дистанционно-векторных алгоритмов.

При централизованном алгоритме используется выделенный маршрутизатор (часто называемый контроллером), собирающий информацию о состоянии сети и управляющий процессом маршрутизации.

Централизованные алгоритмы пока не нашли широкого применения в современных сетях, хотя интенсивные исследования и разработки в данном направлении ведутся. Например, в последнее время наметился переход к концепции так называемых SDN-сетей (Software-defined Networking, дословно – программно-конфигурируемые сети).

Наиболее широкое применение пока еще находят распределенные алгоритмы маршрутизации и, соответственно, основанные на них протоколы маршрутизации, хотя будущее, несомненно за централизованными алгоритмами и SDN-сетями. В настоящем пособии не будем останавливаться на математических принципах, лежащих в основе распределенных алгоритмов, а сами протоколы рассмотрим более подробно в следующих параграфах данной главы.

5.3 Протокол RIP

Протокол RIP (Routing Information Protocol) является одним из первых протоколов маршрутизации и относится к дистанционно-векторным протоколам. Существует две версии RIP – первая версия (RIPv1) использует маршрутизацию на основе классов и описана в RFC 1058, вторая версия (RIPv2) использует бесклассовую маршрутизацию и описана в RFC 1388 [11].

Применение дистанционно-векторной маршрутизации накладывает ограничения на размер составной сети. При этом вводится понятие максимального диаметра сети [3] – максимальное расстояние, на которое может быть передан пакет, после превышения которого пункт назначения считается недостижимым. Для протоколов RIP обеих версий максимальный диаметр сети составляет 15 маршрутизаторов, соответственно, маршрут с метрикой 16 считается недостижимым.

Для рассмотрения процедур, предусмотренных протоколом RIP, рассмотрим тот же пример составной сети, представленный на рисунке 5.4, заимствованным из [9].

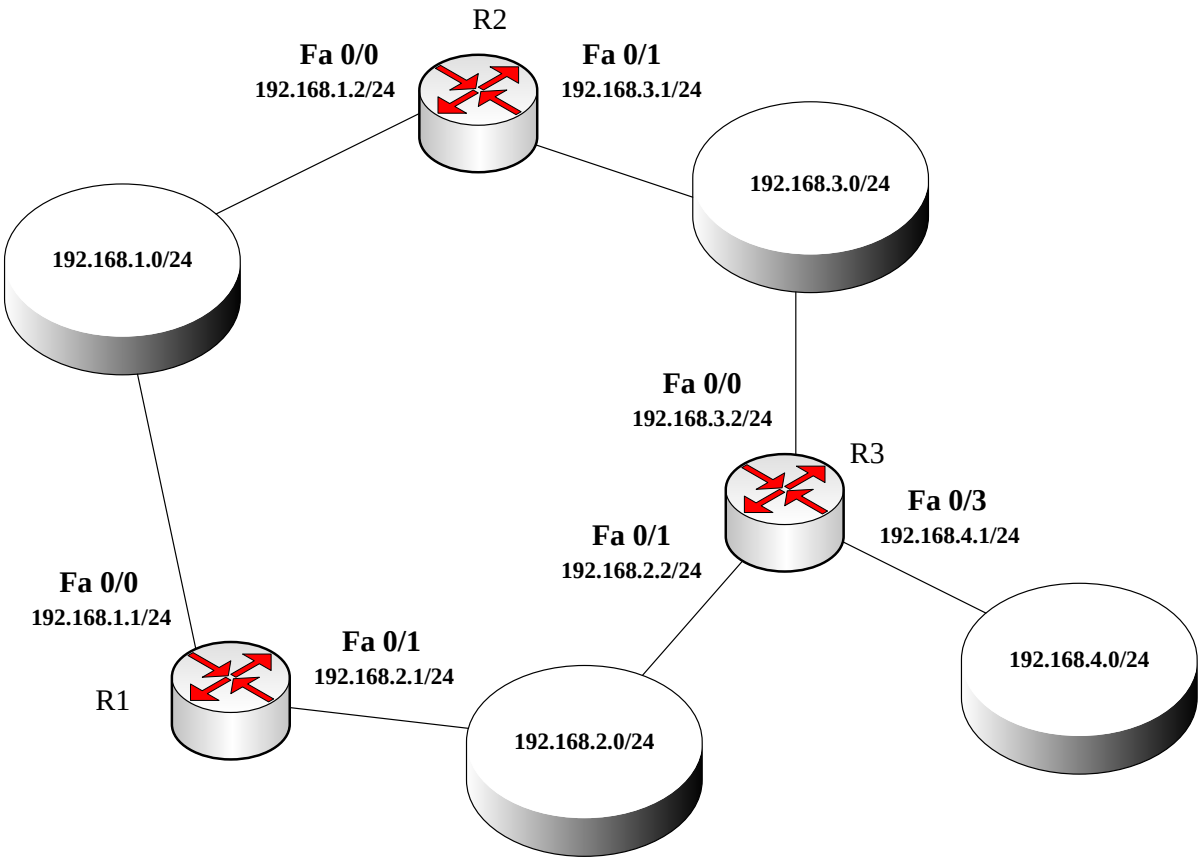

Рисунок 5.4 – Пример составной сети

На рисунке 5.4 представлены три маршрутизатора R1 – R3, у каждого из которых обозначены порты Fast Ethernet (Fa) с назначенными IP-адресами. Адреса портов, как и ранее, соответствуют адресам подсетей, в которые они входят.

На первом этапе протокола RIP создаются минимальные таблицы маршрутизации, которые содержат только адреса непосредственно подключенных подсетей.

Минимальную таблицу маршрутизатора R1 представим в виде таблицы 5.1.

Минимальные таблицы маршрутизаторов R2 и R3 имеют аналогичный вид и представлены в таблицах 5.2 и 5.3, соответственно.

Таблица 5.1 – Минимальная таблица маршрутизатора R1

Адрес сети назначения	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
192.168.1.0	-	192.168.1.1	1
192.168.2.0	-	192.168.2.1	1

Таблица 5.2 – Минимальная таблица маршрутизатора R2

Адрес сети назначения	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
192.168.1.0	-	192.168.1.2	1
192.168.3.0	-	192.168.3.1	1

Таблица 5.3 – Минимальная таблица маршрутизатора R3

Адрес сети назначения	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
192.168.3.0	-	192.168.3.2	1
192.168.2.0	-	192.168.2.2	1
192.168.4.0	-	192.168.4.1	1

На следующем этапе каждый из маршрутизаторов рассылает минимальную таблицу своим «соседям». Для этого используется UDP-сегмент с номером порта 520. В этом сегменте содержатся сведения о сетях, имеющихся в минимальной таблице, и расстояниях (метриках) до них.

Например, «соседями» для маршрутизатора R1 являются маршрутизаторы R2 и R3. Поэтому им передаются сообщения примерно следующего вида:

- сеть 192.168.1.0, метрика 1;

- сеть 192.168.2.0, метрика 1.

Аналогичным образом свою минимальную таблицу маршрутизатор R2 передает маршрутизаторам R1 и R3, а маршрутизатор R3 – маршрутизаторам R1 и R2.

После получения информации от своих «соседей» маршрутизатор обрабатывает ее – увеличивает значение принятой метрики на единицу и запоминает порт, на который пришло данное сообщение, а также адрес маршрутизатора (точнее, его порта), передавшего сообщение. Эта информация заносится в таблицу маршрутизации (в которой уже имеются минимальные записи).

Например, после приема RIP-сообщений от маршрутизаторов R2 и R3 таблица маршрутизатора R1 примет вид, представленный в таблице 5.4.

Таблица 5.4 – Таблица маршрутизатора R1

Адрес сети назначения	Адрес порта следующего маршрутизатора	Адрес выходного порта	Метрика
192.168.1.0	-	192.168.1.1	1
192.168.2.0	-	192.168.2.1	1
192.168.1.0	192.168.1.2	192.168.1.1	2
192.168.3.0	192.168.1.2	192.168.1.1	2
192.168.3.0	192.168.2.2	192.168.2.1	2
192.168.2.0	192.168.2.2	192.168.2.1	2
192.168.4.0	192.168.2.2	192.168.2.1	2

Нетрудно заметить, что строки 3 и 4 были заполнены в результате получения информации от маршрутизатора R2, а строки 5-8 – в результате получения информации от маршрутизатора R3.

Затем маршрутизатор сравнивает принятую информацию с той, которая содержалась в его минимальной таблице. В нашем примере информация о сети 192.168.1.0 содержится как в первой, так и в третьей строках, однако в строке 3 метрика больше, следовательно, эта строка удаляется. Аналогичным образом удаляется строка 6. В строках 4 и 5 содержатся данные о разных маршрутах к одной и той же сети – 192.168.3.0 – и с одним и тем же значением метрики. Поэтому в таблице сохраняется та запись, которая появилась раньше (например, строка 4).

После этого рассмотренные выше процедуры повторяются, только «соседям» рассылаются уже не минимальные таблицы, а таблицы с данными, полученными от других маршрутизаторов. Правило обработки полученной информации и внесения новых данных в таблицу остается прежним – запись о новом маршруте к уже известной сети производится в том случае, если метрика нового маршрута меньше метрики имеющегося маршрута.

В нашем простейшем примере новых записей в таблицу внесено не будет. Тем не менее, маршрутизаторы будут продолжать рассылку своих таблиц каждые 30 секунд, чем обеспечивается корректировка таблиц в случае изменения состояния сети.

Одним из важнейших понятий алгоритмов маршрутизации является время сходимости. Считается, что алгоритм «сошелся», когда все маршрутизаторы имеют согласованную информацию о доступных маршрутах. Время сходимости протокола RIP достаточно велико, поэтому в данном протоколе возможны возникновения петель маршрутизации, что приводит к «зацикливанию» пакетов. В настоящее время эта проблема решается путем введения дополнительных мер (например, использования метода «расщепления горизонта») и ограничением максимального значения метрики.

С другой стороны, данное ограничение не позволяет использовать RIP в крупных сетях. Кроме того, сама логика работы RIP приводит к существенному «засорению» сети служебным трафиком, так как таблицы передаются маршрутизаторами в полном объеме независимо от состояния сети.

5.4 Протокол OSPF

Протокол OSPF (Open Shortest Path First – выбор первого кратчайшего пути) относится к протоколам состояния связей и описан в RFC 1247 и RFC 2328 [11].

Протокол OSPF позволяет либо задавать метрики произвольно, либо использовать метрику по умолчанию. В качестве метрики по умолчанию используется величина, обратно пропорциональная пропускной способности канала, через который проходит маршрут. Такое значение метрики позволяет наиболее оптимально загрузить сеть. Например, если к пункту назначения имеются два маршрута, один из которых более длинный (например, через два промежуточных маршрутизатора), но с высокой пропускной способностью (например, 1 Гбит/с), а другой – более короткий (через один промежуточный маршрутизатор), но с низкой пропускной способностью (10 или 100 Мбит/с), то меньшим значением метрики будет с высокой вероятностью характеризоваться первый маршрут.

Конкретное значение метрики по умолчанию в протоколе OSPF вычисляется как время передачи по каналу одного бита информации, деленное на 10 наносекунд. Например, в канале Fast Ethernet один бит передается за 10 наносекунд (10^{-8} с), соответственно, метрика будет равна 1. Для канала 10 Мбит/с метрика составит 10.

В протоколе OSPF каждый маршрутизатор строит описание сети в виде графа. В графе вершинами являются маршрутизаторы и подсети, а ребрами – связи между ними. Для отыскания оптимального пути на графе используется итерационный алгоритм Дийкстры, обладающий, с одной стороны, быстрой сходимостью, а с другой – вычислительной сложностью, значительно возрастающей при укрупнении сети. Поэтому при реализации протокола OSPF составная сеть разбивается на области (Area), в каждой из которых существует назначенный маршрутизатор (Designated Router – DR) и запасной назначенный маршрутизатор (Backup Designated Router – BDR). Соответственно, маршрутизаторы строят графы состояния связей внутри своей области, что уменьшает вычислительную сложность нахождения оптимальных (кратчайших) путей. В простейшем случае используется одна нулевая область (Area 0).

В отличие от протокола RIP, в протоколе OSPF не используются широковещательные рассылки для обновления сведений о состоянии связей. Вместо этого используются рассылки на Multicast-адрес 224.0.0.6, предназначенные назначенному и запасному назначенному маршрутизаторам, и рассылки на Multicast-адрес 224.0.0.5, предназначенные для остальных маршрутизаторов. Соответственно, при активизации на маршрутизаторе протокола OSPF, он автоматически становится членом группы многоадресной рассылки с адресом 224.0.0.5.

Для вычисления оптимальных маршрутов и занесения их в таблицу маршрутизации используется, как указывалось выше, алгоритм Дийкстры, в качестве исходных данных для которого используется информация, хранящаяся в базе данных. Эта база содержит две таблицы [11]:

- таблица соседних устройств;
- таблица топологии.

Для упрощенного описания процедур протокола OSPF рассмотрим сначала виды сообщений, которыми обмениваются между собой маршрутизаторы:

- сообщение Hello;
- сообщение описания базы данных Data Base Description (DBD);
- сообщение запроса Link-State Request (LSR);
- сообщение обновлений о состоянии связей Link-State Update (LSU);
- сообщение подтверждения Link-State Acknowledgment (LSAck).

Сообщения DBD и LSU могут содержать в себе одно или несколько сообщений Link-State Advertisement (LSA) – объявлений о состоянии связи одного топологического элемента (маршрутизатора, подсети или суммарного маршрута).

Все перечисленные сообщения инкапсулируются непосредственно в IP-пакет. Протоколы транспортных уровней TCP и UDP для передачи сообщений OSPF не используются.

Рассмотрим в несколько упрощенном виде процедуры протокола OSPF, предполагая, что в подсетях используется технология Ethernet (в соединениях маршрутизаторов «точка-точка» и при использовании других технологий канального уровня эти процедуры несколько отличаются).

При инициализации на маршрутизаторе протокола OSPF данный маршрутизатор сначала должен обнаружить своих «соседей». Это обнаружение производится путем обмена сообщениями Hello, в результате чего заполняется таблица соседних устройств, о которой было сказано выше. Зачастую этот этап называют установлением соседских отношений.

Каждый из маршрутизаторов OSPF характеризуется двумя параметрами:

- идентификатор (Router ID);

- приоритет (priority).

По умолчанию приоритет маршрутизатора имеет значение 1 и может быть изменен в пределах от 0 до 255. В качестве идентификатора может быть использован адрес одного из интерфейсов маршрутизатора, кроме того, он также может быть установлен вручную.

Таблица соседних устройств маршрутизатора представлена в таблице 5.5.

Таблица 5.5 – Структура таблицы соседства маршрутизатора OSPF

Идентификатор соседа (Neighbor ID)	Приоритет соседа (Priority)	Состояние (State)	Время до разрыва (Dead Time)	Адрес соседа (Address)	Интерфейс (Interface)

В поле «Состояние» заносится информация о состоянии соседских отношений. Таких состояний может быть семь:

- нерабочее (Down);

- инициализация (Init);

- двунаправленные отношения (Two-Way);

- выборы DR и BDR (Exstart);

- обмен (Exchange);

- загрузка (Loading);

- полные соседские отношения (Full).

Время до разрыва (Dead Time) – это временной интервал, по истечении которого будут разорваны соседские отношения, если от соседа не поступит ни одного сообщения OSPF.

Адрес соседа (Address) – это адрес сетевого уровня соседнего маршрутизатора.

Интерфейс (Interface) – собственный выходной интерфейс в направлении к соседу.

В процессе установления соседских отношений путем обмена сообщениями Hello маршрутизатор заполняет таблицу 5.5. При обмене сообщениями Hello, например, между двумя маршрутизаторами, после внесения в таблицу каждым из них сведений о своем соседе между ними устанавливаются двунаправленные отношения (Two-Way), соответствующая запись появляется в столбце 3 таблицы 5.5.

После установления двунаправленных отношений производится выбор DR и BDR на основании значений идентификаторов и приоритета. При этом сначала рассматриваются приоритеты, и в качестве DR выбирается маршрутизатор с наивысшим приоритетом. При равенстве приоритетов в качестве DR выбирается маршрутизатор с наивысшим значением идентификатора.

После заполнения таблицы соседства маршрутизаторы обмениваются известной им топологической информацией. Для передачи этой информации используется сообщение описания базы данных Data Base Description (DBD). Как указывалось выше, DBD содержит в себе один или несколько LSA. Сообщение LSA переносит информацию только об одном топологическом элементе – маршрутизаторе, подсети или суммарном маршруте. На каждый

LSA отсылается подтверждение LSAck, чем обеспечивается гарантированная доставка топологической информации.

В процессе обмена сообщениями DBD маршрутизаторы находятся в состоянии обмена (Exchange), что отображается в третьем столбце таблицы 5.5.

Важно, что сообщение DBD переносит не всю топологическую информацию, а только список своей таблицы топологии (фактически, список топологических элементов, сведения о которых имеются у него в таблице). Когда маршрутизатор принимает сообщение DBD, он проверяет, содержит ли его таблица информацию о тех топологических элементах, список которых был получен в этом сообщении. Если обнаруживается элемент, сведений о котором у данного маршрутизатора нет, он посылает запрос Link-State Request (LSR), в ответ на который сосед пересылает сообщение обновлений о состоянии связей Link-State Update (LSU), содержащий LSA с полной топологической информацией о запрашиваемом элементе. Данная информация заносится в таблицу топологий.

Таким образом, в результате обмена таблицы топологической информации у маршрутизаторов оказываются синхронизированы, что отражается состоянием Full в столбце 3 таблицы 5.5.

После перехода в это состояние маршрутизаторы запускают алгоритм Дийкстры, в результате реализации которого заполняются таблицы маршрутизации.

Помимо обработки сообщений LSA, для пополнения своих таблиц маршрутизаторы производят их ретрансляцию.

В процессе нормального функционирования каждый маршрутизатор генерирует сообщение Hello с интервалом в 10 секунд. Если от какого-либо маршрутизатора не поступает это сообщение, это говорит об изменении состояния связи. Маршрутизатор в этом случае вносит изменение в свою таблицу топологии и рассылает это изменение с использованием сообщений LSA. Соответственно, каждый маршрутизатор, приняв данное сообщение,

вносит коррективы в свои таблицы и рассылает сообщение дальше. То же самое происходит при появлении в сети нового маршрутизатора после получения от него сообщения Hello.

Внеся изменения в свои таблицы топологии, каждый из маршрутизаторов применяет алгоритм Дийкстры для отыскания оптимальных маршрутов и заносит их в таблицы маршрутизации.

Таким образом, если в сети не происходит изменений, маршрутизаторы обмениваются только сообщениями Hello, а достаточно объемные обновления, передаваемые сообщениями LSA, производятся только при изменении состояния сети. Однако для повышения надежности маршрутизаторы все-таки обмениваются своими базами, но это происходит достаточно редко – по умолчанию раз в 30 минут.

Такое снижение нагрузки на сеть является несомненным достоинством протокола OSPF. Однако применение маршрутизатором достаточно сложного алгоритма Дийкстры требует использования высокопроизводительных процессоров.

Дополнительное снижение нагрузки на сеть при использовании протокола OSPF обеспечивается тем, что в отношении Full маршрутизатор находится только с DR и BDR, с остальными маршрутизаторами поддерживается отношение Two-Way.

Кроме того, протокол OSPF способен распараллеливать передачу данных по нескольким маршрутам, если они имеют одинаковую метрику.

Наконец, как указывалось выше, протокол OSPF поддерживает разделение составной сети на области (Area). Обмен топологической информацией происходит только внутри области, что также снижает нагрузку на сеть.

5.5 Протокол EIGRP

Помимо рассмотренного выше дистанционно-векторного протокола RIP, в свое время широко использовался протокол IGRP (Interior Gateway

Routing Protocol – протокол маршрутизации внутреннего шлюза). Данный протокол также являлся дистанционно-векторным и применялся исключительно в оборудовании Cisco Systems. Затем вышла усовершенствованная версия протокола – EIGRP (Enhanced Interior Gateway Routing Protocol), по существу, превратившая его в гибридный протокол маршрутизации. EIGRP используется не только в оборудовании Cisco, но и в оборудовании других производителей. Так как в настоящее время повсеместно вместо IGRP используется протокол EIGRP, именно его и рассмотрим в этом параграфе.

В отличие от рассмотренных выше протоколов маршрутизации, EIGRP использует достаточно сложную обобщенную метрику, параметры которой, помимо всего прочего, могут настраиваться при конфигурировании сети.

Для вычисления значения обобщенной метрики в EIGRP используются частные значения метрик, к которым относятся:

- пропускная способность канала между получателем и отправителем (BW);

- задержка (D);

- надежность (R);

- нагрузка (L).

Формула для вычисления обобщенной метрики имеет вид [9]

$$M = \left(k1 \cdot BW + \frac{k2 \cdot BW}{256 - L} + k3 \cdot D \right) \cdot \frac{k5}{R + k4},$$

где $k1 ... k5$ - весовые коэффициенты.

При этом частные метрики можно разделить на два вида - статические (не зависящие от состояния сети BW и D) и динамические (зависящие от состояния сети R и L).

По умолчанию в протоколе весовым коэффициентам присвоены значения:

$$k1 = k3 = 1,$$

$$k2 = k4 = k5 = 0.$$

Соответственно, значение обобщенной метрики по умолчанию вычисляется по формуле:

$$M = BW + D.$$

Как следует из последней формулы, по умолчанию протокол учитывает только статические метрики, определяемые типом канала связи и интерфейсом. Сами статические метрики рассчитываются по формулам:

$$BW = \left(\frac{10^7}{bw} \right) \cdot 256,$$

где bw - полоса пропускания, заданная на интерфейсе, кбит/с;

$$D = \left(\frac{d}{10} \right) \cdot 256,$$

где d - задержка, определяемая типом интерфейса, мкс.

Для стандартных интерфейсов значения d определяются следующим образом – для Fast Ethernet $d = 100$ мкс, для Ethernet $d = 1000$ мкс и т.д.

При этом метрика составного маршрута определяется исходя из пропускной способности самого «медленного» канала, входящего в данный маршрут, и суммарной задержки всех интерфейсов, через который проходит маршрут.

В результате обмена маршрутной информацией маршрутизаторы EIGRP подобно маршрутизаторам OSPF формируют базу данных, в которую входят таблица соседства и таблица топологии. Однако алгоритм, по которому вычисляется оптимальный маршрут, качественно другой. Алгоритм, используемый EIGRP, называется DUAL (Diffusing Update Algorithm – алгоритм диффузионного обновления).

Для пополнения своей базы данных маршрутизаторы EIGRP обмениваются между собой пятью типами пакетов:

- пакет обновлений Update;
- пакет запроса Query;
- пакет ответа на запрос Reply;

- пакет приветствия Hello;

- пакет подтверждения Ack.

Данные пакеты аналогично протоколу OSPF используют групповой адрес 224.0.0.10 или индивидуальные адреса соседних маршрутизаторов. Так как протокол EIGRP является внутренним (Interior), он функционирует только в пределах так называемой автономной системы (понятие автономной системы более подробно будет рассмотрено в следующем параграфе). Соответственно, перечисленные выше пакеты содержат номер автономной системы. Маршрутизатор EIGRP, приняв пакет, обрабатывает его только в том случае, если его передал маршрутизатор, относящийся к этой же автономной системе.

Аналогично маршрутизатору OSPF, маршрутизатор EIGRP при инициализации должен установить соседские отношения. Эти отношения устанавливаются путем обмена пакетами Hello, в результате которого каждый маршрутизатор, устанавливающий соседские отношения, заполняет свою таблицу соседства. Поля таблицы соседства маршрутизаторов EIGRP представлены в таблице 5.6.

Таблица 5.6 – Таблица соседства

Номер соседа H	Адрес соседа Address	Собственный интерфейс Interface	Время удержания Holdtime	Время отношений Uptime	Счетчик очереди Count	Последовательный номер Seq Num	Таймер цикла SRTT	Таймер повторной передачи RTO

В отличие от протокола OSPF, в EIGRP используется порядковый номер соседнего маршрутизатора (H). В поле Address указывается его сетевой адрес, а в поле Interface – собственный интерфейс, через который доступен сосед.

Время удержания (Holdtime) – это интервал времени, по истечении которого, в случае отсутствия EIGRP-пакетов от соседнего

маршрутизатора, он считается недостижимым. По умолчанию данный интервал равен 15 с.

Время отношений (Uptime) – это интервал времени, прошедший с момента установления соседских отношений.

Счетчик очереди (Count) – это число пакетов, находящихся в очереди передачи.

Последовательный номер (Seq Num) – номер последнего EIGRP-пакета, принятого от соседа. Используется как для контроля порядка приема пакетов, так и для передачи пакета подтверждения.

Таймер цикла (SRTT) – временной интервал, необходимый для отправления соседу пакета и получения от него ответа. Если в течение этого интервала ответ не приходит, пакет отправляется повторно.

Таймер повторной передачи (RTO) – интервал, в течение которого ожидается подтверждение от соседа о приеме им пакета. Если в течение этого интервала подтверждения не приходит, пакет отправляется повторно.

В ответ на пакет Hello маршрутизатор отправляет пакет Update по индивидуальному адресу маршрутизатора, от которого был принят пакет Hello (при этом в пакете Update содержится подтверждение получения пакета Hello, что снижает необходимый объем пакетов подтверждений Ack).

Таким образом, в результате такого обмена маршрутизаторы EIGRP заполняют свои таблицы соседства. После установления соседских отношений маршрутизаторы обмениваются между собой своими таблицами топологии.

Рассмотрим состав таблицы топологии.

В первой колонке таблицы представлен статус маршрута. Символ P (Passive) обозначает пассивный, то есть готовый к использованию маршрут. Символ A (Active) обозначает активный маршрут, то есть маршрут, по которому не закончен расчет по алгоритму DUAL. Как следует из рисунка 3.15, все маршруты в данном примере пассивные.

После указания состояния маршрута указываются подсеть назначения и число преемников (Successors). Под преемником понимается первичный маршрут, который после вычисления передается в таблицу маршрутизации.

FD (Feasible Distance) обозначает выполнимое расстояние (метрику) до сети назначения по данному маршруту. Выполнимым расстоянием называется полная метрика маршрута, которая вычисляется как сумма заявленного расстояния (то есть полученного от соседа) и расстояния до соседа.

После via указывается источник маршрута. Например, запись via Connected означает, что подсеть подключена непосредственно к данному маршрутизатору. Запись via 156.92.15.5 (284160/281600) означает, что маршрут был анонсирован маршрутизатором с адресом выходного интерфейса 156.92.15.5 с заявленным состоянием 281600, выполнимое расстояние составляет 284160.

Наконец, последняя запись означает выходной интерфейс данного маршрутизатора.

Кроме понятия преемника, в протоколе EIGRP существует понятие вероятного преемника – резервного маршрута, который задействуется при отказе маршрутизатора-преемника. Вероятный преемник не заносится в таблицу маршрутизации, но хранится в таблице топологии.

Протокол EIGRP по умолчанию способен распараллеливать передачу данных по маршрутам с равной метрикой, а после соответствующей настройки – и по маршрутам с различными метриками.

Имея в своей базе таблицу топологий, каждый из маршрутизаторов в соответствии с алгоритмом DUAL рассчитывает оптимальные маршруты и заносит их в таблицу маршрутизации.

Детальное рассмотрение таблиц соседства, топологии и маршрутизации будет дано в следующей главе.

5.6 Протокол BGP

Рассмотренные выше протоколы маршрутизации относились к так называемым протоколам внутренних шлюзов. Под протоколом внутреннего шлюза понимается протокол, предназначенный для работы внутри автономной системы.

Под автономной системой (AS – Autonomous System) понимается совокупность сетей, работающих под единым административным управлением [3].

Например, в сети Интернет термин AS используется для описания крупных логически объединенных сетей, каждая из которых идентифицируется определенным числом.

С точки зрения большой сети, объединяющей несколько AS, каждая автономная система представляется как единый объект. В каждой AS действуют протоколы внутреннего шлюза, более того, этих протоколов может быть несколько. Область AS, в которой работает один из протоколов внутренней маршрутизации, называется доменом маршрутизации. Соответственно, возникает задача объединения автономных сетей между собой, что осуществляется с использованием протоколов внешнего шлюза (EGP – Exterior Gateway Protocol).

Одним из самых распространенных протоколов внешнего шлюза является протокол BGP (Border Gateway Protocol – протокол пограничного шлюза). С 1994 года действует четвертая версия протокола (RFC 1771), которую и рассмотрим в этом параграфе.

Автономные системы объединяются между собой с помощью так называемых пограничных маршрутизаторов. Сама автономная система может принадлежать к различным типам. В частности, автономная система может быть тупиковой, то есть связанной только с одной другой автономной системой. Если имеется связь с несколькими AS, автономная система называется многопортовой. Многопортовая AS, способная передавать через себя трафик других автономных систем, называется транзитной [15]. Пример

объединения нескольких автономных систем между собой представлен на рисунке 5.5.

На рисунке представлено шесть автономных систем, объединенных с использованием пограничных маршрутизаторов. Маршрутизаторы, находящиеся внутри автономных систем и использующие протоколы внутреннего шлюза, на рисунке не показаны. Задачей пограничных маршрутизаторов является обеспечение передачи данных между автономными системами, для чего и используется протокол BGP.

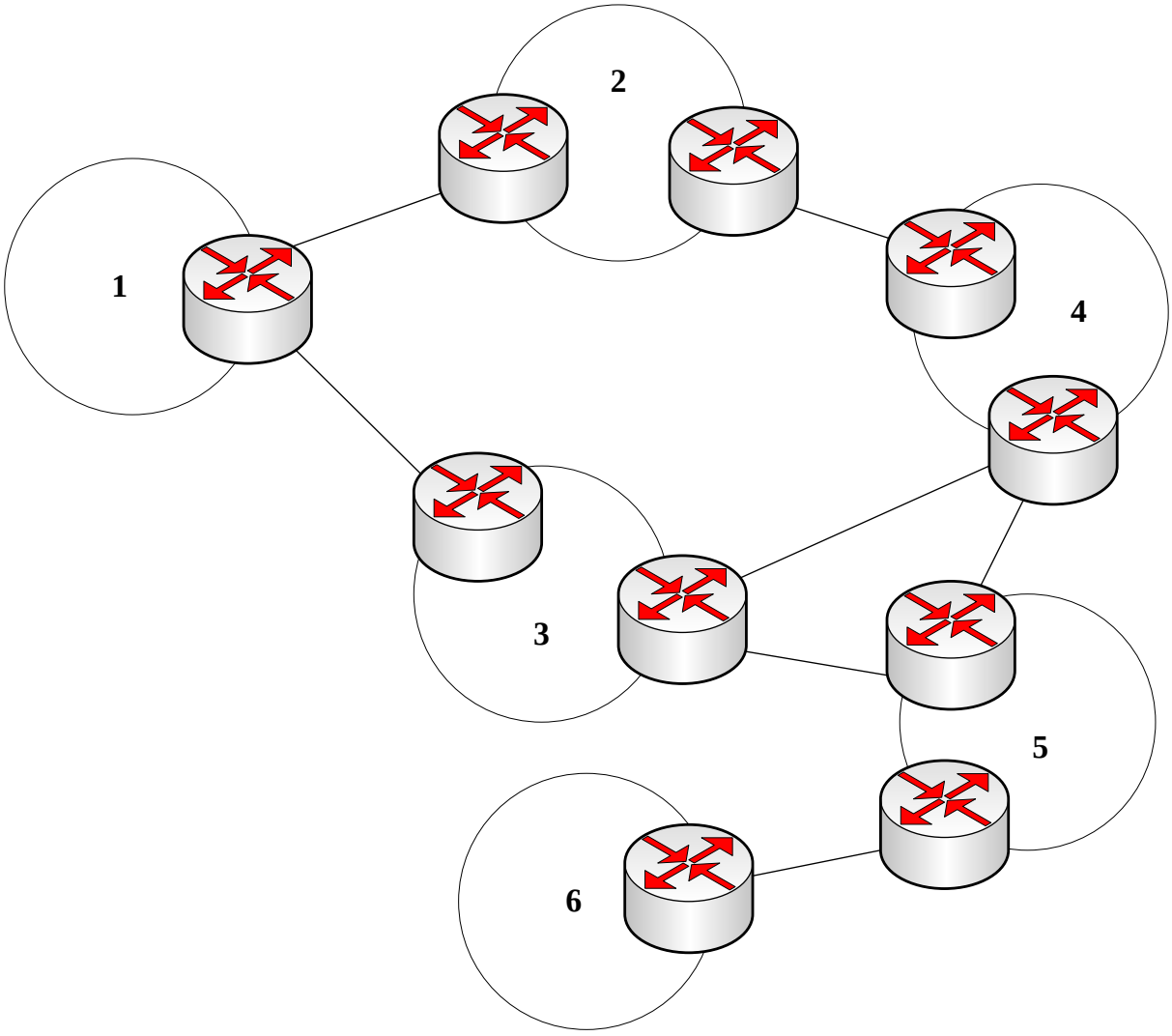

Рисунок 5.5 – Объединение автономных систем

Протоколы внешнего шлюза, в том числе и BGP, принципиально отличаются от протоколов внутреннего шлюза. Основным отличием является

то, что для расчета маршрута используется не столько метрика, зависящая от технических параметров, сколько соображения нетехнического характера, например, экономические соображения. Эти соображения формализуются в виде так называемой политики автономной системы. Таким образом, ни дистанционно-векторные, ни протоколы состояния связей не могут быть использованы в качестве протоколов внешнего шлюза.

Рассмотрим неприменимость протоколов внутреннего шлюза для объединения автономных систем на удачном примере, приведенном в [15].

Предположим, что пограничные маршрутизаторы используют какой-либо протокол состояния связей. При передаче данных из AS1 в AS5 был рассчитан оптимальный маршрут AS1-AS2-AS4-AS5. Тогда передача данных из AS2 в AS5 будет осуществляться по маршруту AS2-AS4-AS5, так как все маршрутизаторы одинаково интерпретируют метрики связей (например, пропускную способность) и используют одни и те же математические процедуры для вычисления маршрутов. Однако может оказаться так, что AS2 выгоднее передавать свой трафик по маршруту AS1-AS3-AS5, так как у AS4 невыгодный тариф за пропуск транзитного трафика. Соответственно, происходит зацикливание, так как каждая AS по своему интерпретирует понятие метрики.

Если же использовать в такой же ситуации дистанционно-векторный протокол, то, например, AS6 для передачи данных в AS2 выберет маршрут AS6-AS5-AS4-AS2. Однако может так случиться, что AS4 по каким-либо причинам нетехнического характера не пропускает через себя трафик AS6. Так как информация обо всем маршруте в дистанционно-векторных протоколах не передается, AS2 окажется недостижима, несмотря на то, что существует маршрут AS6-AS5-AS3-AS1-AS2.

Из данного примера следует, что для передачи данных между автономными системами необходимо использовать какой-либо другой подход. Именно такой подход и реализуется протоколом BGP, который, аналогично дистанционно-векторным протоколам, предполагает рассылку

(анонсирование) маршрутизаторами своим соседям векторов, но не расстояний, а путей (Path Vector). В отличие от вектора расстояний, вектор путей содержит не только адрес сети назначения и расстояние (метрику) до нее, но и ряд атрибутов, описывающих данный маршрут. При этом атрибутов должно быть достаточно для того, чтобы маршрутизатор мог выбрать наилучший путь с точки зрения политики своей автономной системы.

Например, одним из атрибутов пути является атрибут AS_PATH, содержащий в себе список номеров автономных систем, через которые достижима сеть назначения. Данный атрибут может быть использован для вычисления метрики маршрута – числа промежуточных AS, для обнаружения маршрутных петель – одна и та же AS не должна встречаться в маршруте дважды, а также для определения маршрутной политики – маршрут не должен включать неприемлемые AS. При анонсировании маршрута маршрутизатор BGP добавляет в атрибут AS_PATH номер своей автономной системы.

Протокол BGP может быть как внешним, так и внутренним. Внутренний BGP (iBGP) необходим для обмена маршрутной информацией между маршрутизаторами BGP внутри автономной системы. Например, при передаче данных из AS1 через AS2 (рисунок 5.5) в пределах AS2 должны взаимодействовать два маршрутизатора. Внешний BGP (eBGP) используется для обмена маршрутной информацией между различными автономными системами.

Важно, что при использовании iBGP маршрутизатор не добавляет в атрибут AS_PATH номер своей автономной системы, так как в противном случае после добавления такого же номера на «выходе» AS пограничным маршрутизатором этот номер появится в атрибуте дважды, что будет расценено как образование маршрутной петли.

Подобно некоторым рассмотренным выше протоколам внутреннего шлюза, маршрутизаторы BGP должны установить между собой соседские отношения. Для этого они используют надежное TCP-соединение, порт 179.

При этом маршрутизаторы разных автономных систем должны быть соединены непосредственно, для маршрутизаторов, относящихся к одной AS, такого ограничения нет, так как они могут обмениваться информацией с использованием протоколов внутреннего шлюза.

Данные, которыми обмениваются между собой соседи в рамках открытого TCP-соединения, называются BGP-сообщениями.

Протокол предусматривает использование следующих сообщений:

- сообщение Open;
- сообщение Keepalive;
- сообщение Update;
- сообщение Notification.

Сообщение Open посылается маршрутизатором после установления TCP-соединения. Ответом на Open является сообщение Keepalive, если второй маршрутизатор согласен стать BGP-соседом. В противном случае посылается сообщение Nnotification с кодом, поясняющим причину отказа, и соединение разрывается.

Сообщение Keepalive, помимо описанной выше функции подтверждения согласия на установление соседских отношений, используется для периодического контроля работоспособности соседа.

Сообщение Update предназначено для анонсирования соседу маршрутов. В процессе инициализации протокола BGP маршрутизатор с использованием сообщения Update передает данные обо всех анонсируемых маршрутах, а впоследствии – только об изменениях, то есть данные об удаленных или добавленных маршрутах.

После установления соседских отношений и приема сообщений Update маршрутизатор BGP принимает решение о том, какой путь использовать к сети назначения. К конкретной сети определяется только один путь, который и заносится в таблицу маршрутизации.

Процесс выбора наилучшего маршрута является итерационным и предусматривает последовательное сравнение значений атрибутов, о которых

говорилось ранее. Помимо упомянутого атрибута AS_PATH, BGP использует еще ряд атрибутов, которые здесь подробно рассматривать не будем. Исходя из их значений, а также принятой в данной AS маршрутной политики, маршрутизатор выбирает наилучший маршрут.

Так как маршрутизаторы BGP обычно связывают достаточно крупные автономные системы, для сокращения количества записей в таблице маршрутизации широко используется CIDR – бесклассовая междоменная маршрутизация. Данная технология позволяет агрегировать маршруты, то есть объединять несколько маршрутов с одинаковым префиксом в один. Для агрегированных маршрутов также применяются специальные атрибуты.

6 Конфигурирование маршрутизаторов и коммутаторов третьего уровня

6.1 Настройка статической маршрутизации

В отличие от коммутаторов, маршрутизаторы для своего правильного функционирования должны быть правильно сконфигурированы. Как минимум, необходимо интерфейсам маршрутизатора присвоить адреса, как это было показано в параграфе 5.1. Если адреса назначаются администратором и конфигурируются администратором вручную, такие адреса называются статическими. Рассмотрим сначала статическую маршрутизацию.

Рассмотрим пример, представленный на рисунке 6.1.

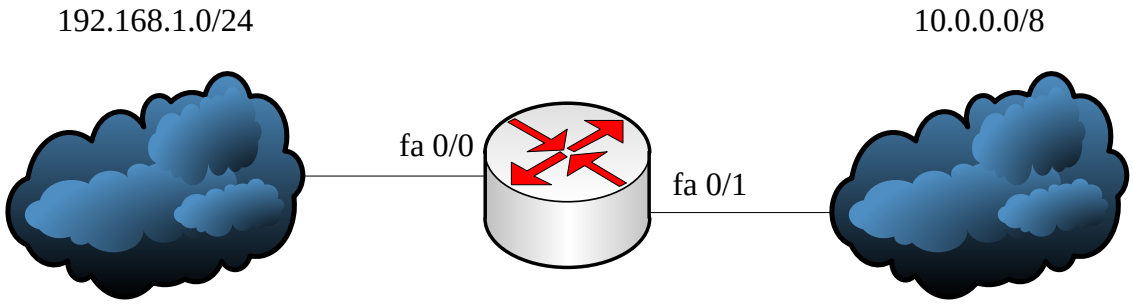

Рисунок 6.1 – Пример использования маршрутизатора

На рисунке представлены две сети, соединенных между собой маршрутизатором. Для обеспечения продвижения пакетов между подсетями портам маршрутизатора необходимо назначить адреса в соответствии с адресами подсоединенных сетей.

Для этого добавим в рабочую область Cisco Packet Tracer один маршрутизатор (например, Cisco 2811). Зададим его интерфейсам IP-адреса из диапазонов, приведенных выше сетей. Интерфейсу FastEthernet 0/0 присвоим IP-адрес 192.168.1.1 с маской 24, а интерфейсу FastEthernet 0/1 IP-

адрес 10.10.10.1 с маской 8. Для этого на маршрутизаторе в режиме конфигурирования выполним следующие команды:

Router(config)#interface fastEthernet 0/0

Router(config-if)#ip address 192.168.1.1 255.255.255.0

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#interface fastEthernet 0/1

Router(config-if)#ip address 10.10.10.1 255.0.0.0

Router(config-if)#no shutdown

Поясним значение данных команд. Команда interface fastEthernet 0/0 служит для того, чтобы перейти к конфигурированию интерфейса fastEthernet 0/0 маршрутизатора (в тот момент когда она выполняется, указатель ввода Router(config) меняется на Router(config-if), это свидетельствует о том, что осуществлен переход из режима глобального конфигурирования маршрутизатора в режим конфигурирования выбранного интерфейса). Команда ip address 192.168.1.1 255.255.255.0 назначает указанному выше интерфейсу IP-адрес 192.168.1.1 с маской 255.255.255.0. Команда no shutdown включает выбранный интерфейс (по умолчанию все интерфейсы маршрутизатора в отличие от коммутатора находятся в отключенном состоянии).

После того, как будут применены данные команды, необходимо выйти из режима конфигурации и выполнить команду show ip route. Данная команда отобразит содержимое таблицы маршрутизации данного маршрутизатора.

Таблица маршрутизации на данный момент пуста (рисунок 6.2). На экране отобразилась лишь справочная информация о сокращениях, используемых для обозначения маршрутов.

Теперь подключим к маршрутизатору сети с указанными на рисунке 6.1 адресациями. Для упрощения рассмотрим частный случай, когда в каждой из сетей располагается по одному компьютеру. Сначала подключим к

интерфейсу FastEthernet 0/0 компьютер с IP адресом 192.168.1.100/24, в качестве основного шлюза укажем ему 192.168.1.1.

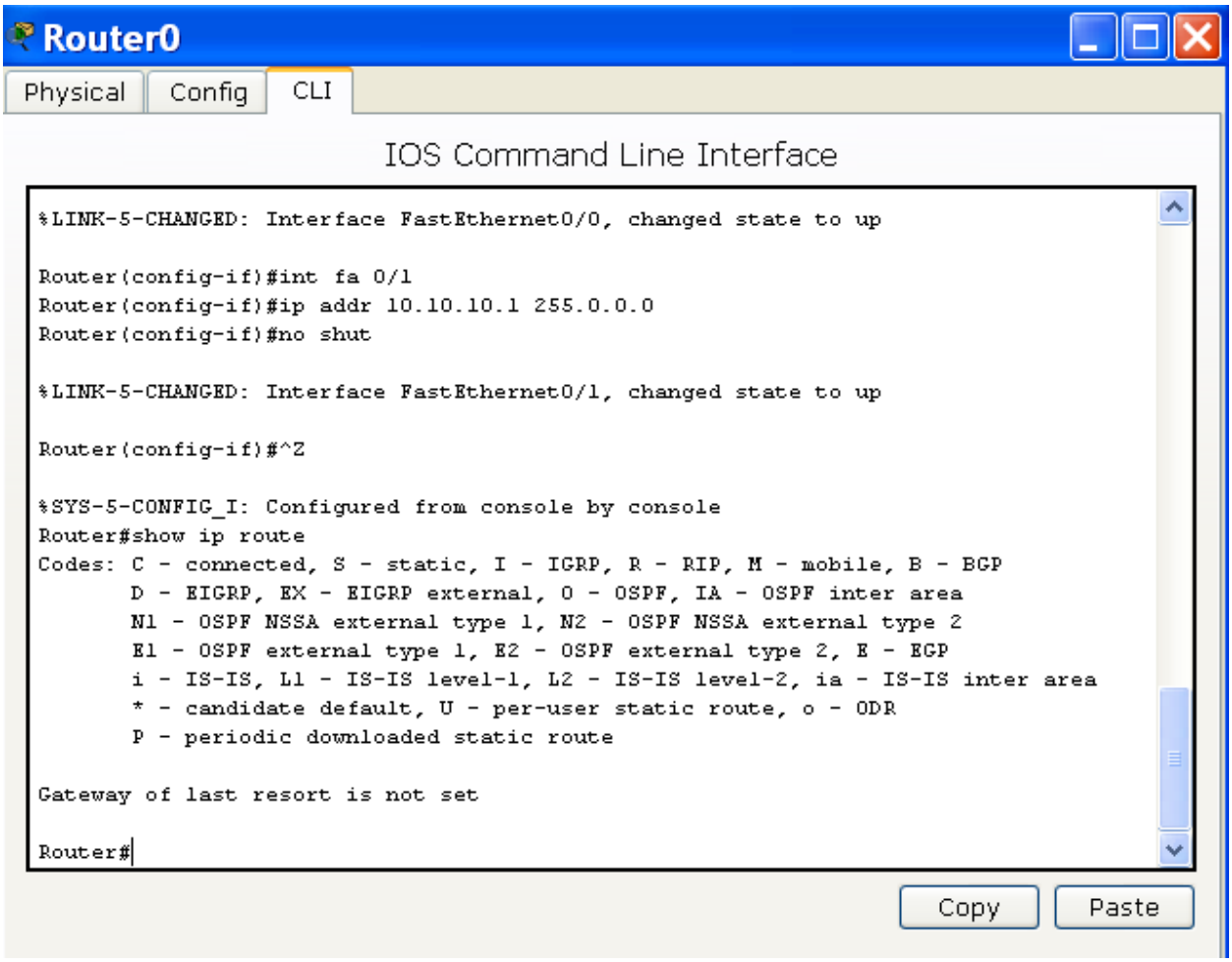

Рисунок 6.2 – Результат выполнения команды show ip route

В Cisco Packet Tracer это будет выглядеть так, как показано на рисунке 6.3 (используется кроссированное подключение).

Рисунок 6.3 – Соединение компьютера с маршрутизатором

После этого не будем подключать сразу второй компьютер, а перейдем на маршрутизатор и посмотрим содержимое его таблицы маршрутизации, рисунок 6.3.

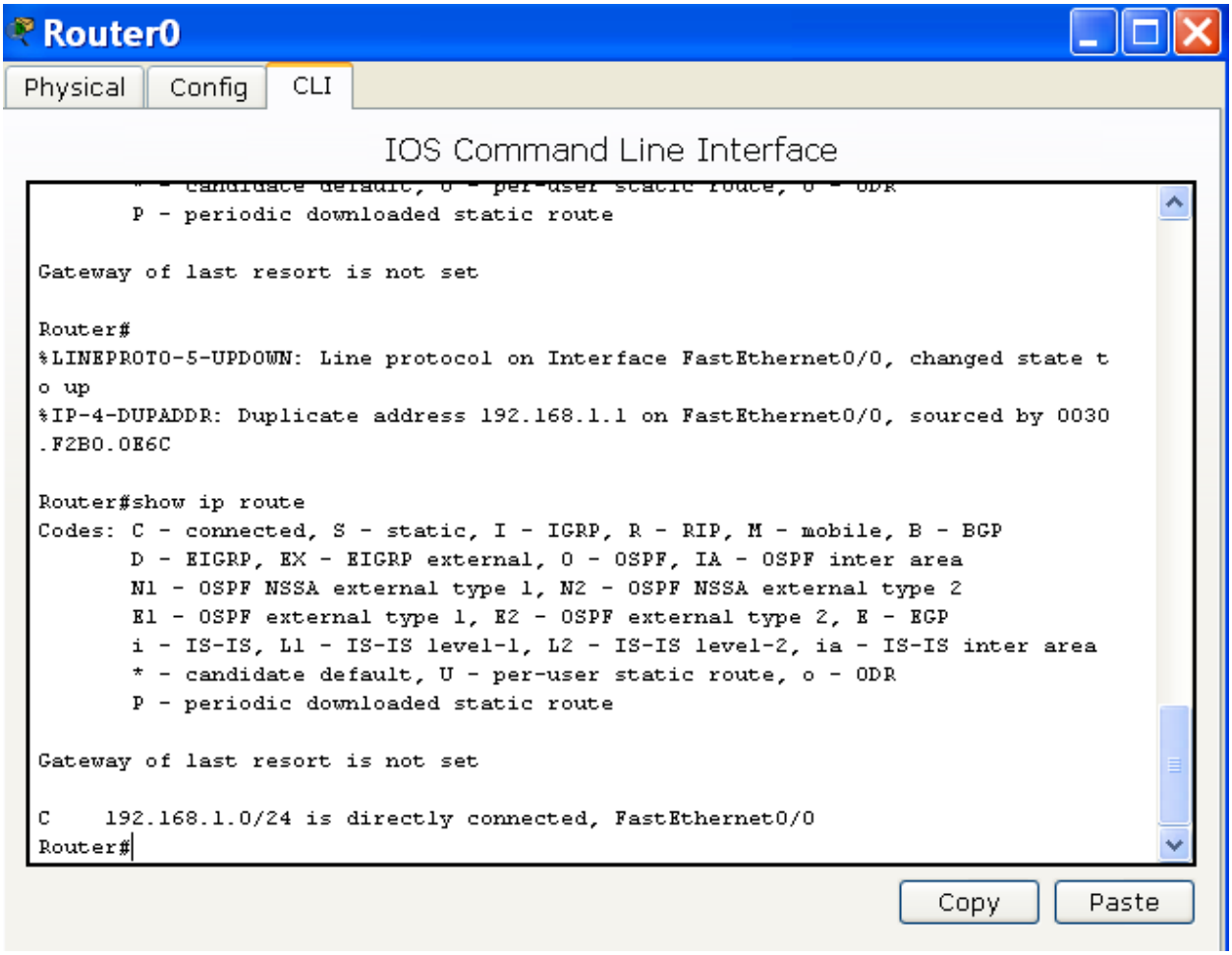

Рисунок 6.3 – Таблица маршрутизации после подключения к нему сети 192.168.1.0

Как можно заметить, в таблице появилась информация о подключенной непосредственно к маршрутизатору сети, выглядящая как "C 192.168.1.0/24 is directly connected, FastEthernet0/0". Буква C обозначает, что данная сеть подключена непосредственно к данному маршрутизатору (Connected), 192.168.1.0/24 – это адрес данной сети, а FastEthernet0/0 – это интерфейс, через который она доступна.

Теперь подключим в Packet Tracer к маршрутизатору второй компьютер. Компьютеру зададим IP адрес 10.10.10.100/8, в качестве основного шлюза укажем 10.10.10.1.

После этого снова просмотрим содержимое таблицы маршрутизации, в ней появилась информация и о второй подключенной сети. Проверим доступность компьютеров одной сети из другой. Для этого необходимо перейти в консоль компьютера с адресом 10.10.10.100 и выполнить в ней команду **ping 192.168.1.100**. Для трассировки статического маршрута можно выполнить команду **tracert**, рисунок 6.4.

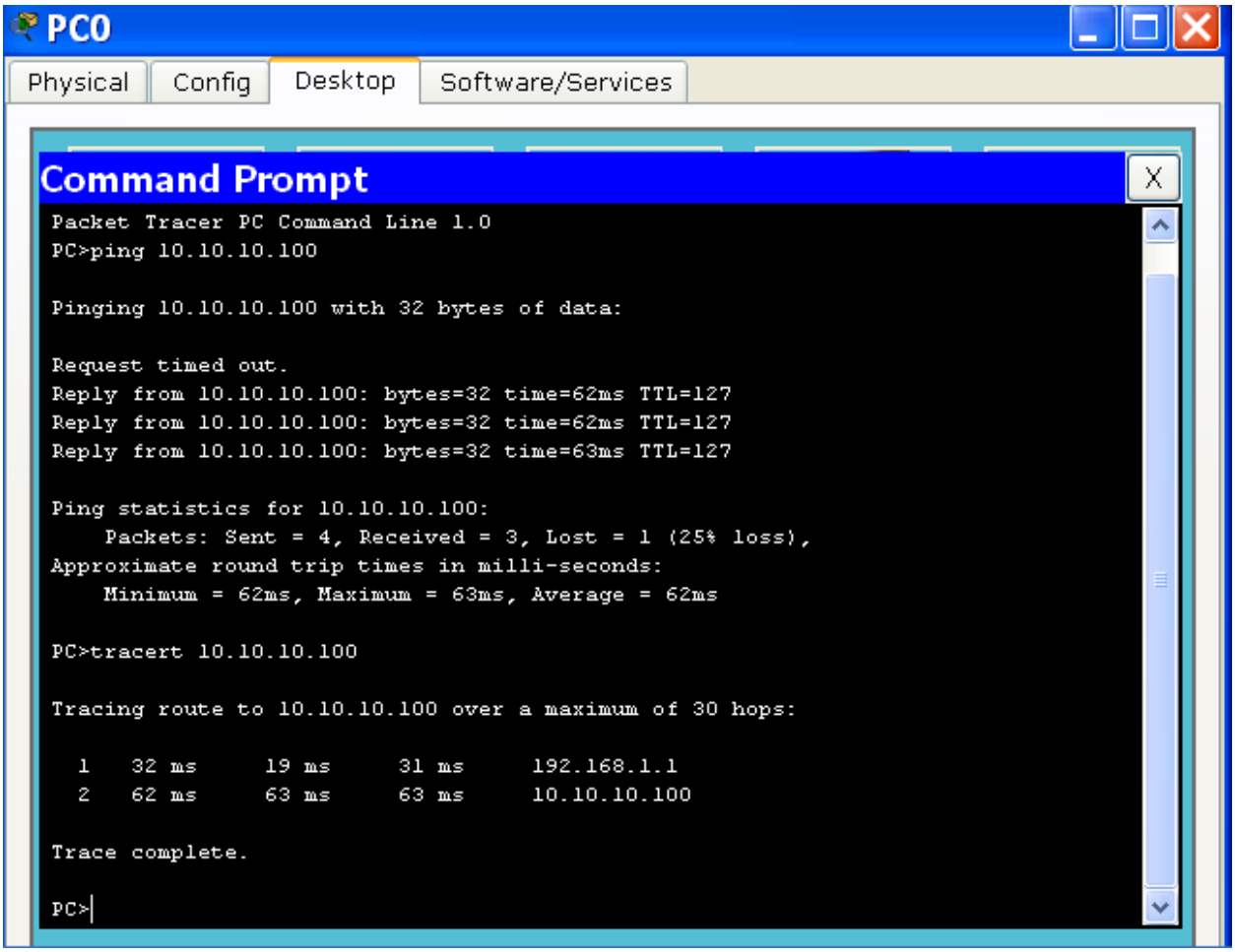

Рисунок 6.4 – Результат трассировки маршрута

Из рисунка видно, что пакет от рабочей станции с адресом 192.168.1.100 поступает на интерфейс маршрутизатора с адресом 192.168.1.1,

продвигается во вторую подсеть, и принимается рабочей станцией с адресом 10.10.10.100.

Таким образом, если имеется всего один маршрутизатор, то достаточно всего лишь сконфигурировать его интерфейсы, и он будет выполнять маршрутизацию между сетями, подключенными к нему. Ситуация несколько усложняется, если в сети есть несколько маршрутизаторов. Например, составная сеть имеет вид, представленный на рисунке 6.5.

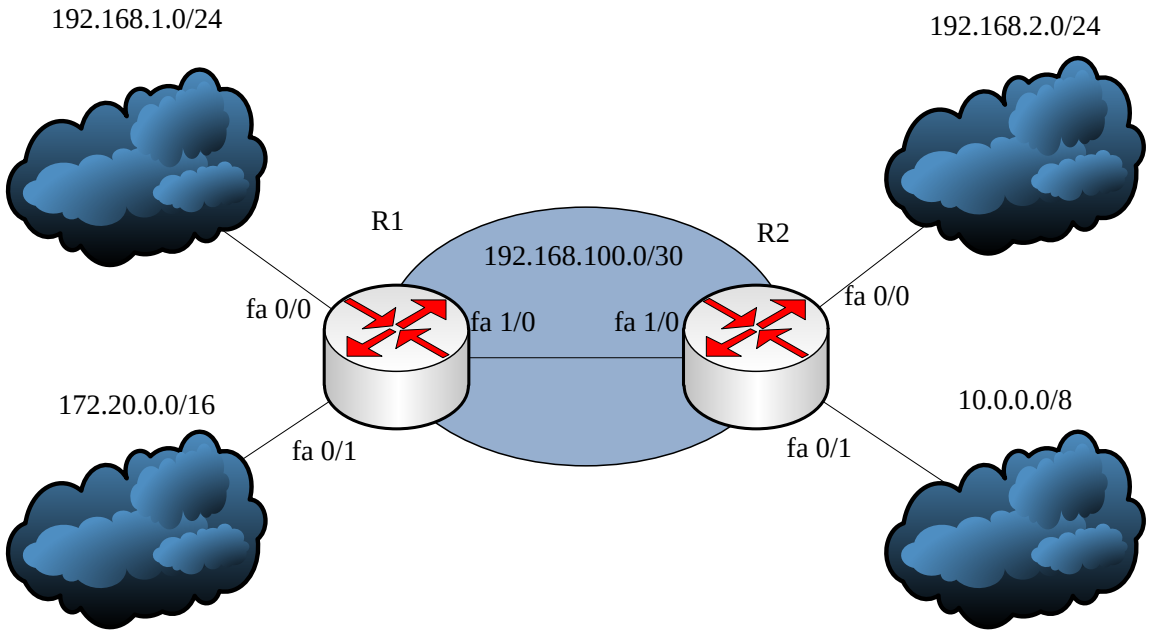

Рисунок 6.5 – Пример составной сети

Отдельно следует отметить, что соединение маршрутизаторов «точка-точка» тоже является полноправной сетью со своей адресацией. Так как в этой сети всего два узла, она называется вырожденной и использует маску, состоящую из 30-и единиц (255.255.255.252). Несложно заметить, что с такой маской для адресации узлов остается только 2 разряда, что, с учетом запрета на адреса, состоящие только из единиц или только из нулей, дает возможность сформировать только два адреса.

Как было указано выше, после непосредственного подключения сетей к маршрутизатору и назначения его интерфейсам IP-адресов, в его таблице

появляется информация о маршрутах к этим сетям. Так маршрутизатор R1 будет иметь в своей таблице информацию о сетях 192.168.1.0/24, 172.20.0.0/16 и 192.168.100.0/30, следовательно, передача пакетов между этими сетями возможна. Аналогично обстоят дела с маршрутизатором R2 (у него есть информация о сетях 192.168.2.0/24, 10.0.0.0/8 и 192.168.100.0/30). Очевидно, что в этом случае компьютеры сети 10.0.0.0/8 не будут доступны из сети 192.168.1.0/24.

Смоделируем данную ситуацию в Cisco Packet Tracer. Для начала соберем схему, показанную на рисунке 6.6.

Рисунок 6.6 – Схема составной сети

На данной схеме компьютер PC0 имеет IP адрес – 192.168.1.100/24, PC1 – 172.20.20.100/16, PC2 – 192.168.2.100/24, PC3 – 10.10.10.100/8. Интерфейсы маршрутизатора, к которым подключены компьютеры, имеют такие же адреса, как и сами компьютеры, только в четвертом октете стоит 1. Например, интерфейс, к которому подключен ПК с адресом 192.168.1.100/24, имеет IP-адрес 192.168.1.1/24. В качестве шлюза по умолчанию у каждого компьютера указан интерфейс маршрутизатора, к которому он подключен. После настройке интерфейсов маршрутизаторов, необходимо сохранить их конфигурации, выполнив команду **write memory**, выполняемую из привилегированного режима. Далее соединим маршрутизаторы между собой. Чтобы это сделать, потребуется добавить к маршрутизатору интерфейсную плату. В данном случае добавим к маршрутизатору плату NM-1FE-TX (NM – Network module, 1FE – содержит один порт Fast Ethernet, TX – поддерживает

10/100MBase-TX). Чтобы это сделать, необходимо перейти к окну конфигурации маршрутизатора, выключить его, щелкнув по кнопке питания, после чего «перетянуть» необходимую интерфейсную плату в разъем маршрутизатора.

После того, как карта добавлена, нужно еще раз щелкнуть по тумблеру маршрутизатора, чтобы включить его. В его интерфейсах должен добавиться еще один – FastEthernet1/0. Повторим аналогичные действия со вторым маршрутизатором, зададим интерфейсу FastEthernet1/0 маршрутизатора R1 IP-адрес 192.168.100.1 с маской 255.255.255.252, а маршрутизатору R2 – 192.168.100.2 с аналогичной маской. После этого соединим интерфейсы FastEthernet1/0 этих маршрутизаторов между собой.

Проверим доступность компьютеров одной сети из других. Если все сделано верно, то картина доступности должна быть следующая. Допустим, с компьютера с IP-адресом 172.20.20.100 поступает эхо-запрос на интерфейс с IP-адресом 192.168.100.2, маршрутизатора R2. Эхо-ответ получен не будет.

Разберем причину этого. Запрос поступает на интерфейс маршрутизатора R1 с адресом 172.20.20.1, маршрутизатор направляет его на порт маршрутизатора R2 с адресом 192.168.100.2. То есть запрос достигает пункта назначения. Однако маршрутизатор R2 отправить ответ не может, так как в его таблице отсутствуют сведения о сети 172.20.20.0/16.

В нашей простейшей сети самым простым способом настроить маршрутизацию является добавление маршрута по умолчанию. Для того, чтобы это сделать, нужно выполнить на маршрутизаторе R1 в режиме конфигурирования следующую команду:

ip route 0.0.0.0 0.0.0.0 192.168.100.2

На маршрутизаторе R2:

ip route 0.0.0.0 0.0.0.0 192.168.100.1

В следующих командах первые 4 цифры обозначают IP-адрес сети назначения, следующие 4 цифры обозначают её маску, а последние 4 цифры – это IP-адрес интерфейса следующего маршрутизатора (next hop), на

который необходимо передать пакеты, чтобы попасть в данную сеть. Если мы указываем в качестве адреса сети 0.0.0.0 с маской 0.0.0.0, то данный маршрут становится маршрутом по умолчанию, и все пакеты, адреса назначения которых прямо не указаны в таблице маршрутизации, будут отправлены на адрес, указанный в нем.

Посмотрим, как это выглядит на конкретном примере. Допустим, мы хотим послать запрос с использованием утилиты ping с компьютера с адресом 192.168.1.100 на компьютер с IP-адресом 10.10.10.100. В качестве шлюза по умолчанию на компьютере с адресом 192.168.1.100 установлен адрес интерфейса маршрутизатора 192.168.1.1. Сначала компьютер будет искать в свой таблице маршрутизации непосредственно адрес 10.10.10.100, после того как он его не найдет, он начнет искать в таблице маршрутизации маршрут к сети 10.0.0.0/8. После того, как данный маршрут так же не будет обнаружен, ICMP-пакеты будут отправлены на адрес по умолчанию, то есть на интерфейс маршрутизатора с адресом 192.168.1.1. Получив пакет, маршрутизатор просмотрит адрес его назначения – 10.10.10.100, и также попытается обнаружить его в свой таблице маршрутизации. Когда этот адрес обнаружен не будет, маршрутизатор попробует найти в свой таблице маршрут к сети 10.0.0.0/8. Когда он не обнаружит и его, пакет будет отправлен, используя маршрут по умолчанию. Соответственно, ICMP-пакет будет передан на интерфейс с адресом 192.168.100.2 маршрутизатора R2. Этот маршрутизатор попробует обнаружить в свой таблице маршрутизации маршрут к адресу 10.10.10.100. Когда этот адрес обнаружен не будет, маршрутизатор R2 будет искать маршрут к сети 10.0.0.0/8. Информация о данной сети содержится в таблице маршрутизации, и маршрутизатор «знает», что для того, чтобы попасть в данную сеть, необходимо отправить пакеты на интерфейс FastEthernet0/1, непосредственно к которому подключена данная сеть. Так как в нашем примере вся сеть 10.0.0.0/8 представляет из себя всего один компьютер, то пакеты сразу же попадают в место назначения – компьютер с IP-адресом 10.10.10.100. При отсылке ответных ICMP-пакетов,

все происходит аналогичным образом, только адресом назначения уже будет являться 192.168.1.100/24.

Очевидно, далеко не всегда можно обойтись указанием только маршрутов по умолчанию. В более сложных сетевых конфигурациях может потребоваться прописывать маршрут для каждой из сетей в отдельности. Рассмотрим процедуру конфигурирования таких статических маршрутов. Для этого сначала удалим из таблицы маршрутизации все статически добавленные маршруты, используя команду

no ip route xxxx(адрес сети) yyyy(маска) zzzz(адрес интерфейса)

В конечном итоге таблицы маршрутизации должны содержать только информацию о непосредственно подключенных к ним сетях.

Теперь необходимо добавить к каждому из маршрутизаторов маршруты к двум сетям, которые ему неизвестны (к сетям, подключенным к соседнему маршрутизатору). На маршрутизаторе R1 выполним:

ip route 192.168.2.0 255.255.255.0 192.168.100.2

ip route 10.0.0.0 255.0.0.0 192.168.100.2

На маршрутизаторе R2 выполним:

ip route 192.168.1.0 255.255.255.0 192.168.100.1

ip route 172.20.0.0 255.255.0.0 192.168.100.1

Если все сделано верно, связность всех сетей будет обеспечена, что нетрудно проверить и использованием утилит **ping** и **traceroute**.

6.2 Настройка динамической маршрутизации

В случае, если сеть достаточно крупная, одних статических маршрутов будет явно недостаточно, необходимо настроить динамическую маршрутизацию. В этом случае маршрутизаторы смогут обмениваться между собой маршрутной информацией и таким образом самостоятельно заполнять свои таблицы маршрутизации.

Настройка динамической маршрутизации зависит от того протокола, в соответствии с которым будут работать маршрутизаторы. Рассмотрим

сначала конфигурирование наиболее простого протокола маршрутизации – RIP, рассмотренного в параграфе 5.3.

Конфигурирование протокола RIP состоит из трех основных этапов [14]:

- разрешение маршрутизатору исполнять протокол RIP;

- выбор версии протокола RIP;

- задание сетевых адресов и интерфейсов, которые должны включаться в пакеты обновления данных маршрутизации.

Первый этап реализуется путем выполнения на маршрутизаторе команды

router rip

Выбор версии протокола (этап 2) реализуется путем выполнения субкоманды

version

Напомним, что существуют две версии протокола RIP. Версия 1 основана на классовой адресации, версия 2 способна использовать маски переменной длины в IP-адресации.

Наконец, для реализации третьего этапа используется субкоманда

network

Данная команда должна использоваться для идентификации только тех сетевых адресов, которые непосредственно подключены к конфигурируемому маршрутизатору и предназначены для включения в пакеты обновления маршрутной информации. В пакеты обновления будут включаются только те интерфейсы, которые имеют IP-адреса, принадлежащие идентификационной сети [14].

Приведем пример, взятый из [5]. Допустим, что имеется маршрутизатор, у которого интерфейсы имеют следующие адреса:

131.108.4.5/16;

131.108.6.9/16;

172.16.3.6/16.

Исполнение команды

network 131.108.0.0

приведет к тому, что объявления с маршрутной информацией будут посылаться только с данными о подсетях сети 131.108.0.0 и только в интерфейсы, которые адресуются в сети 131.108.0.0. Чтобы включить в пакеты обновления маршрутной информации интерфейс, находящийся в адресном пространстве 172.16.0.0, необходимо выполнить команду

network 172.16.3.6

Соберем в Cisco Packet Tracer сеть, состоящую из двух маршрутизаторов, аналогичную той, которую мы рассмотрели в предыдущем параграфе (рисунок 6.5). Назначим интерфейсам маршрутизаторов те же сетевые адреса, которые мы использовали в предыдущем параграфе.

Произведем конфигурирование маршрутизатора Router0, рисунок 6.7.

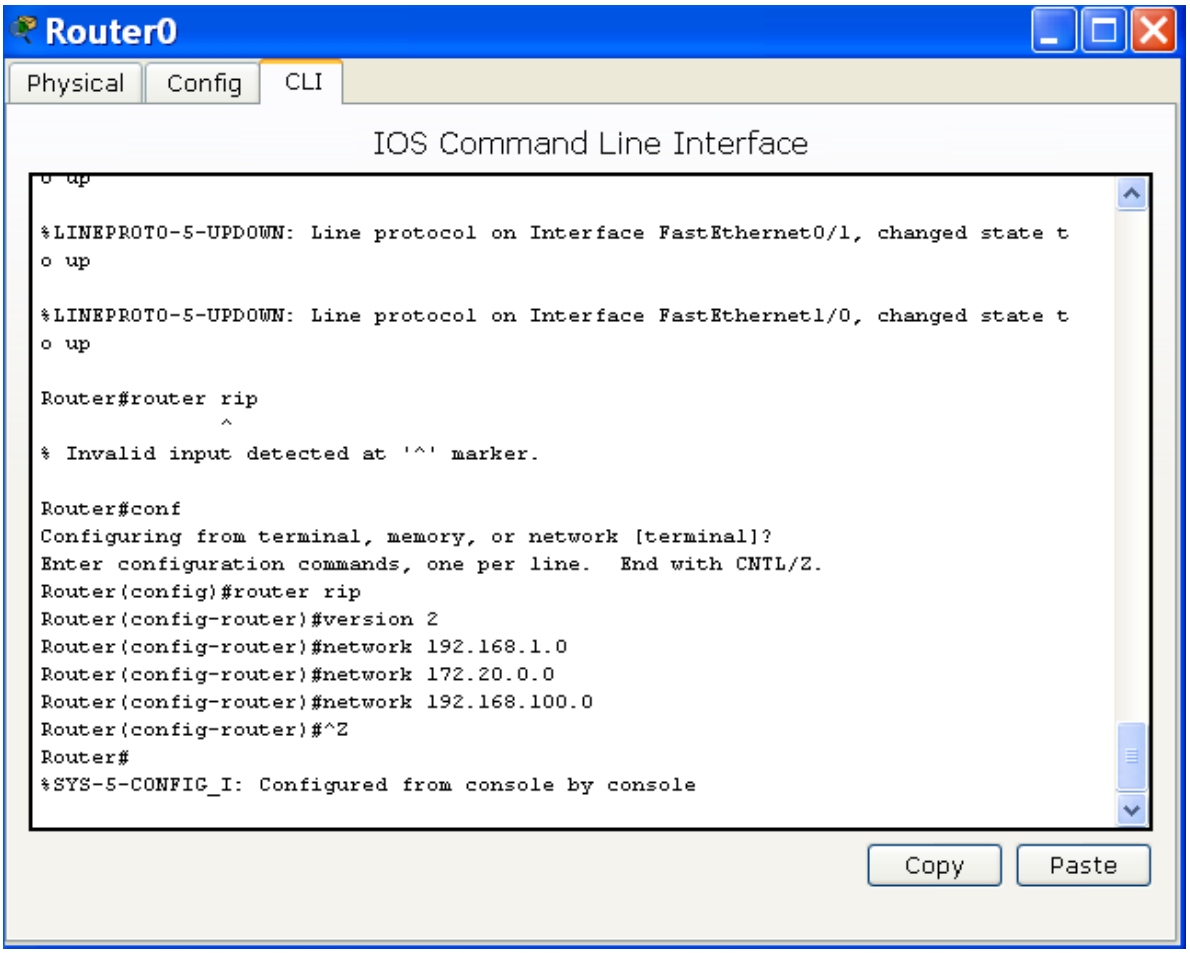

Рисунок 6.7 – Конфигурирование маршрутизатора

Аналогично произведем конфигурирование маршрутизатора Router1, рисунок 6.8.

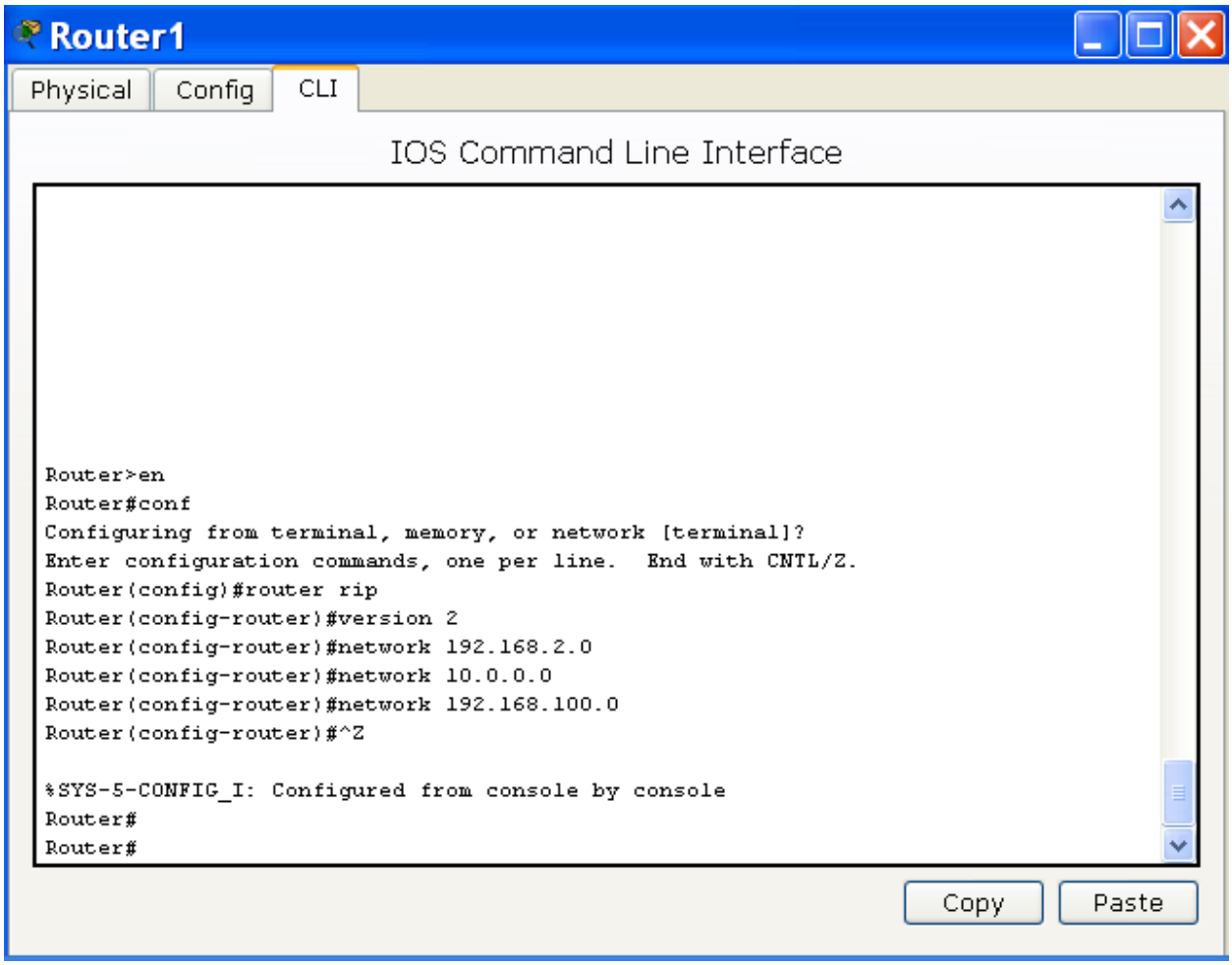

Рисунок 6.8 – Конфигурирование маршрутизатора Router1

Посмотрим таблицу маршрутизации маршрутизатора Router0, рисунок 6.9.

Из рисунка видно, что в таблице появились две записи, помеченные как R (что означает протокол динамической маршрутизации RIP). Из таблицы следует, что в сеть 10.0.0.0 маршрут проходит через порт маршрутизатора Router1 192.168.100.2 через собственный выходной порт fast Ethernet 1/0.

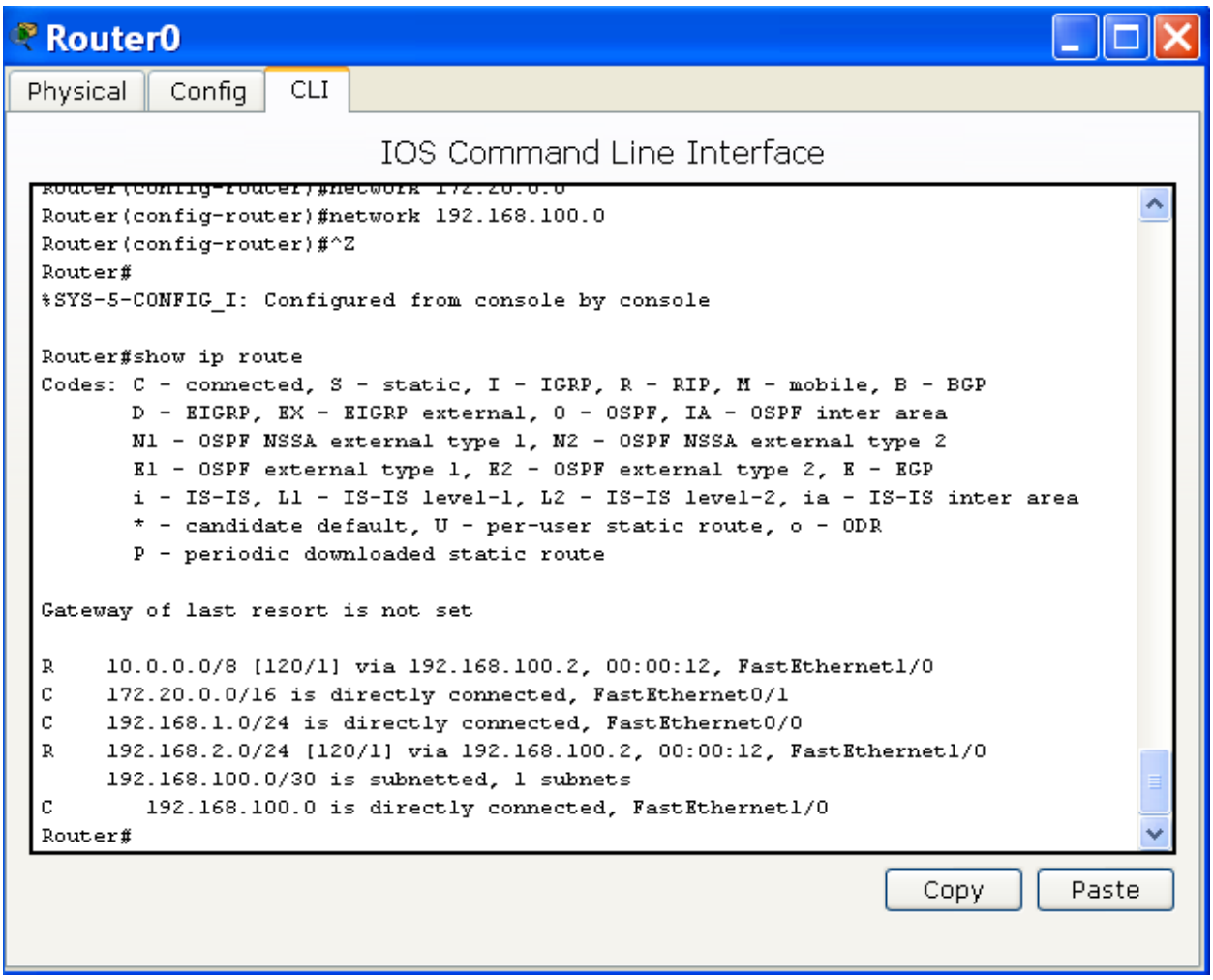

Рисунок 6.9 – Таблица маршрутизации Router0

Таким образом, маршрутизация в составной сети настроена.

Протокол RIP имеет ряд недостатков, которые более подробно были рассмотрены в параграфе 5.3. Современные маршрутизаторы, как правило, могут реализовывать несколько протоколов маршрутизации – OSPF, BGP, IGRP, EIGRP, и т.д. Рассмотрим конфигурирование протокола EIGRP на примере сети, изображенной на рисунке 6.10.

Из рисунка видно, что маршрутизаторы связаны друг с другом с использованием вырожденной подсети «точка-точка», кроме того, к каждому из них подключена своя подсеть. Например, маршрутизатор R1 связан с подсетью 192.168.1.0/24 с использованием интерфейса с адресом 192.168.1.1/24. С маршрутизатором R0 он связан через вырожденную подсеть 156.92.15.4/30 с использованием интерфейса с адресом 156.92.15.6/30, а с

маршрутизатором R2 – через вырожденную подсеть 156.92.15.8/30 с использованием интерфейса с адресом 156.92.15.6/30.

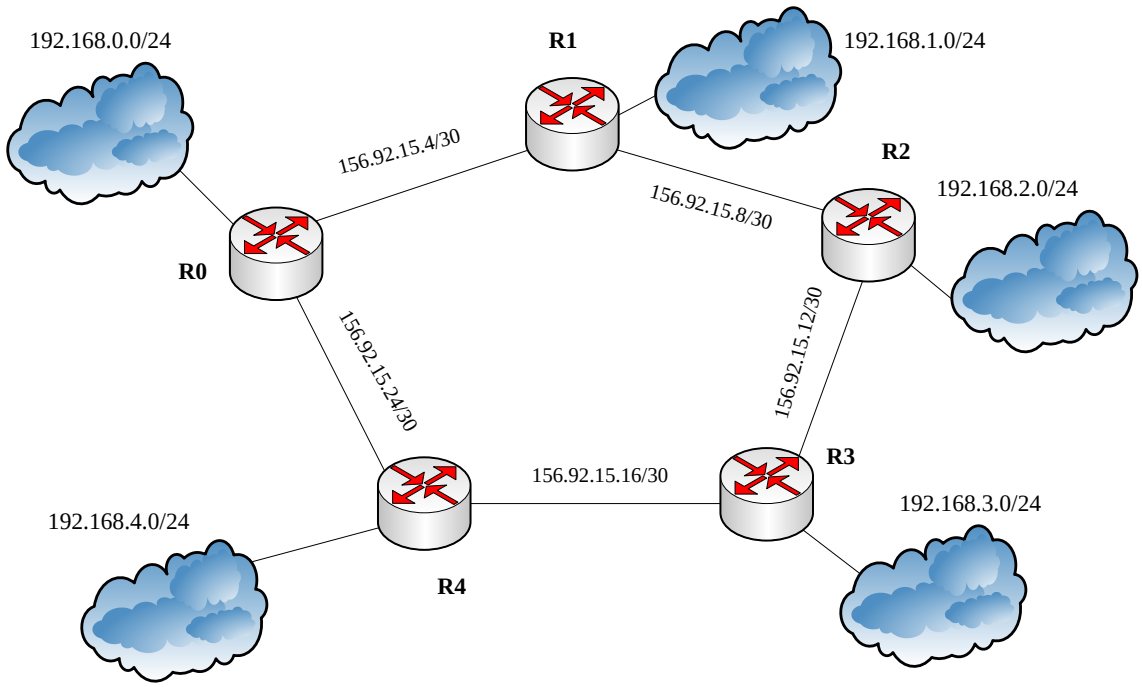

Рисунок 6.10 – Пример составной сети

Для включения на маршрутизаторе протокола EIGRP используется команда

Router eigrp <AS>

Здесь AS – номер автономной системы, который должен быть одинаковым на всех маршрутизаторах, обменивающихся сообщениями протокола EIGRP.

Далее необходимо указать, о каких подключенных сетях маршрутизатору необходимо рассылать информацию. Осуществляется это командой

network <адрес сети> <специальная маска>

Под специальной маской (ориг. – wildcardmask) понимается четырехбайтовое двоичное число, указывающее с использованием своих нулевых разрядов, какая часть разрядов адреса сети должна быть обработана. Например, если в вышеуказанной команде имеется сеть 192.168.1.0 и

используется специальная маска 0.0.255.255, то информация будет рассылаться обо всех сетях, адрес которых начинается с 192.168.

Конфигурирование протокола EIGRP на маршрутизаторе R0 иллюстрируется рисунком 6.11.

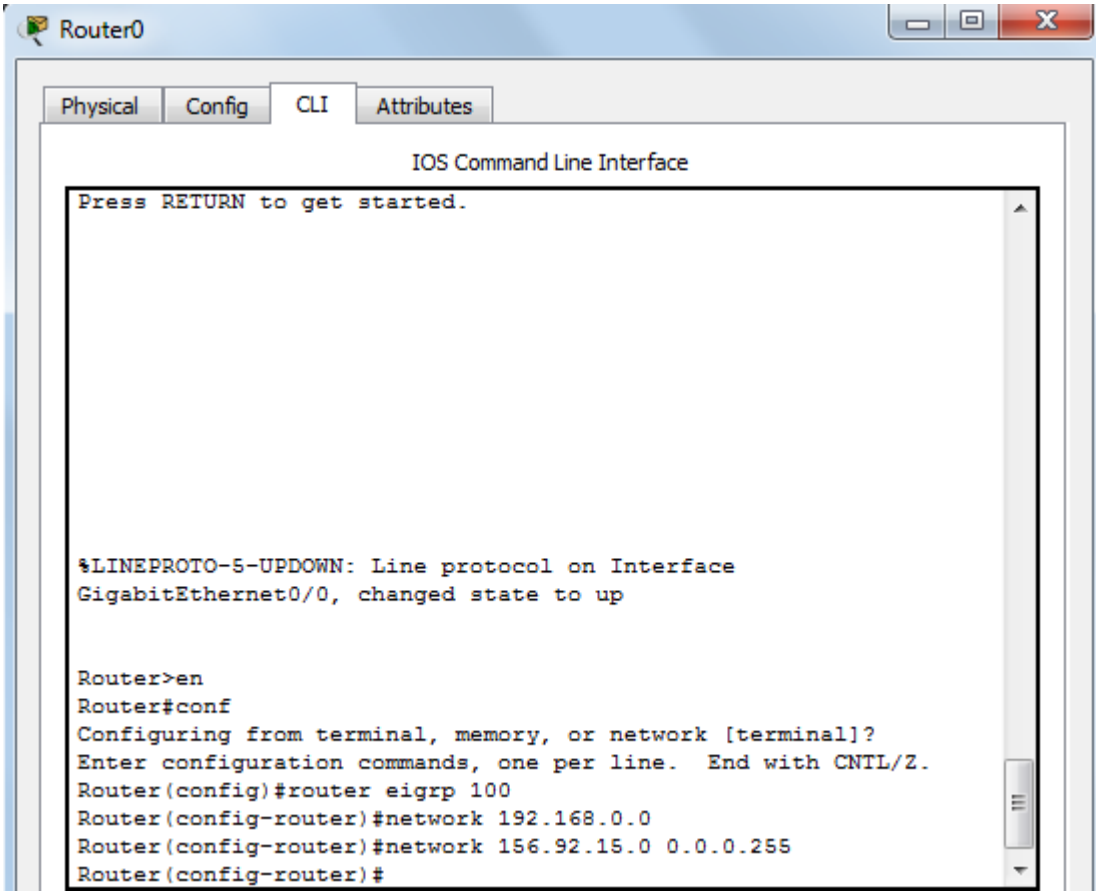

Рисунок 6.11 – Конфигурирование маршрутизатора R0

Аналогичным образом конфигурируются остальные маршрутизаторы сети, показанной на рисунке 6.10.

Вид таблицы соседства маршрутизатора R3 представлен на рисунке 6.12 (используется команда **show ip eigrp neighbor**).

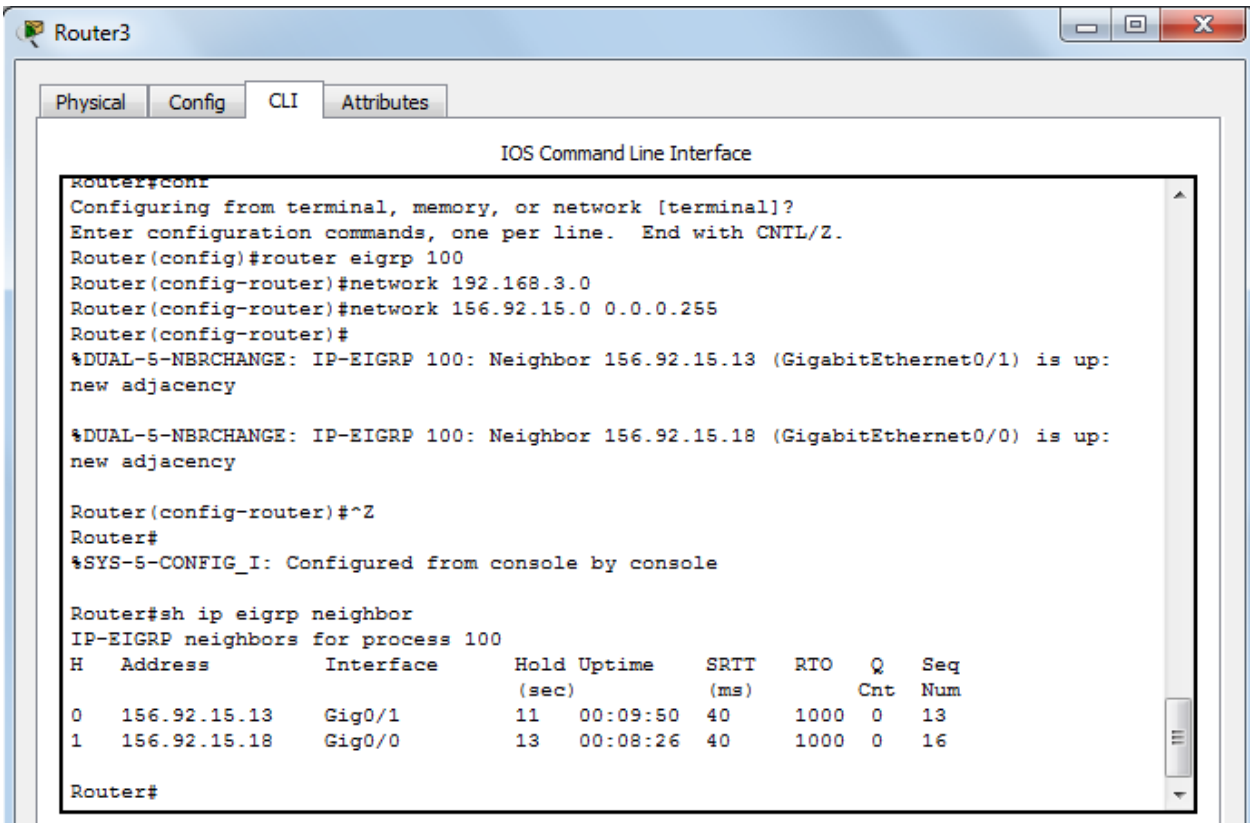

Рисунок 6.12 – Таблица соседства маршрутизатора R3

В ответ на пакет Hello маршрутизатор отправляет пакет Update по индивидуальному адресу маршрутизатора, от которого был принят пакет Hello (при этом в пакете Update содержится подтверждение получения пакета Hello, что снижает необходимый объем пакетов подтверждений Ack).

Таким образом, в результате такого обмена маршрутизаторы EIGRP заполняют свои таблицы соседства. После установления соседских отношений маршрутизаторы обмениваются между собой своими таблицами топологии. Таблица топологии того же маршрутизатора R3 представлена на рисунке 6.13 (используется команда **show ip eigrp topology**).

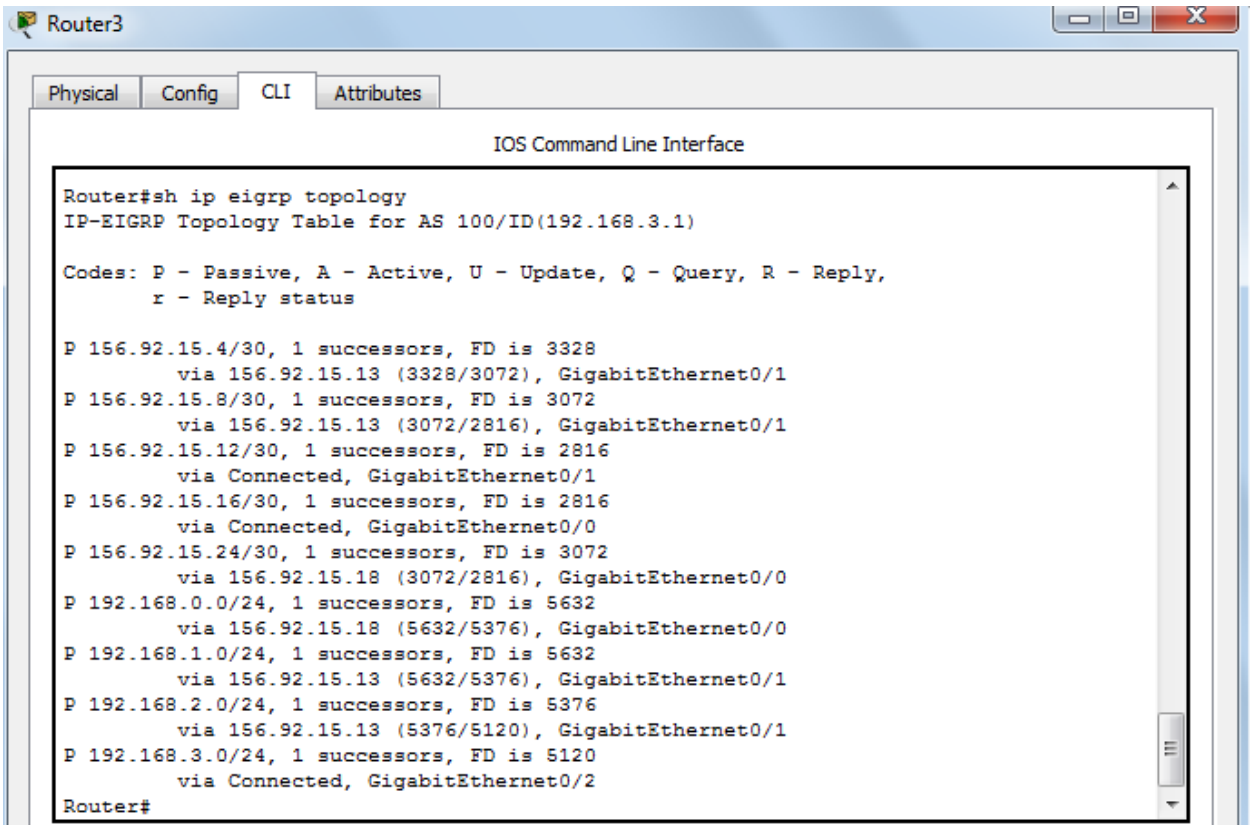

Рисунок 6.13 – Таблица топологии маршрутизатора Cisco

Рассмотрим еще раз состав таблицы топологии, теперь уже на конкретном примере.

В первой колонке таблицы представлен статус маршрута. Символ P (Passive) обозначает пассивный, то есть готовый к использованию маршрут. Символ A (Active) обозначает активный маршрут, то есть маршрут, по которому не закончен расчет по алгоритму DUAL. Как следует из рисунка 3.15, все маршруты в данном примере пассивные.

После указания состояния маршрута указываются подсеть назначения и число преемников (Successors). Под преемником понимается первичный маршрут, который после вычисления передается в таблицу маршрутизации.

FD (Feasible Distance) обозначает выполнимое расстояние (метрику) до сети назначения по данному маршруту. Выполнимым расстоянием называется полная метрика маршрута, которая вычисляется как сумма заявленного расстояния (то есть полученного от соседа) и расстояния до соседа.

После via указывается источник маршрута. Например, запись via Connected означает, что подсеть подключена непосредственно к данному маршрутизатору. Запись via 156.92.15.13 (3328/3072) означает, что маршрут был анонсирован маршрутизатором с адресом выходного интерфейса 156.92.15.13 с заявленным состоянием 3328, выполнимое расстояние составляет 3072.

Наконец, последняя запись означает выходной интерфейс данного маршрутизатора.

Кроме понятия преемника, в протоколе EIGRP существует понятие вероятного преемника – резервного маршрута, который задействуется при отказе маршрутизатора-преемника. Вероятный преемник не заносится в таблицу маршрутизации, но хранится в таблице топологии.

Протокол EIGRP по умолчанию способен распараллеливать передачу данных по маршрутам с равной метрикой, а после соответствующей настройки – и по маршрутам с различными метриками.

Имея в своей базе таблицу топологий, каждый из маршрутизаторов в соответствии с алгоритмом DUAL рассчитывает оптимальные маршруты и заносит их в таблицу маршрутизации. Пример таблицы маршрутизации, соответствующей таблице топологии (рисунок 6.13) представлен на рисунке 6.14.

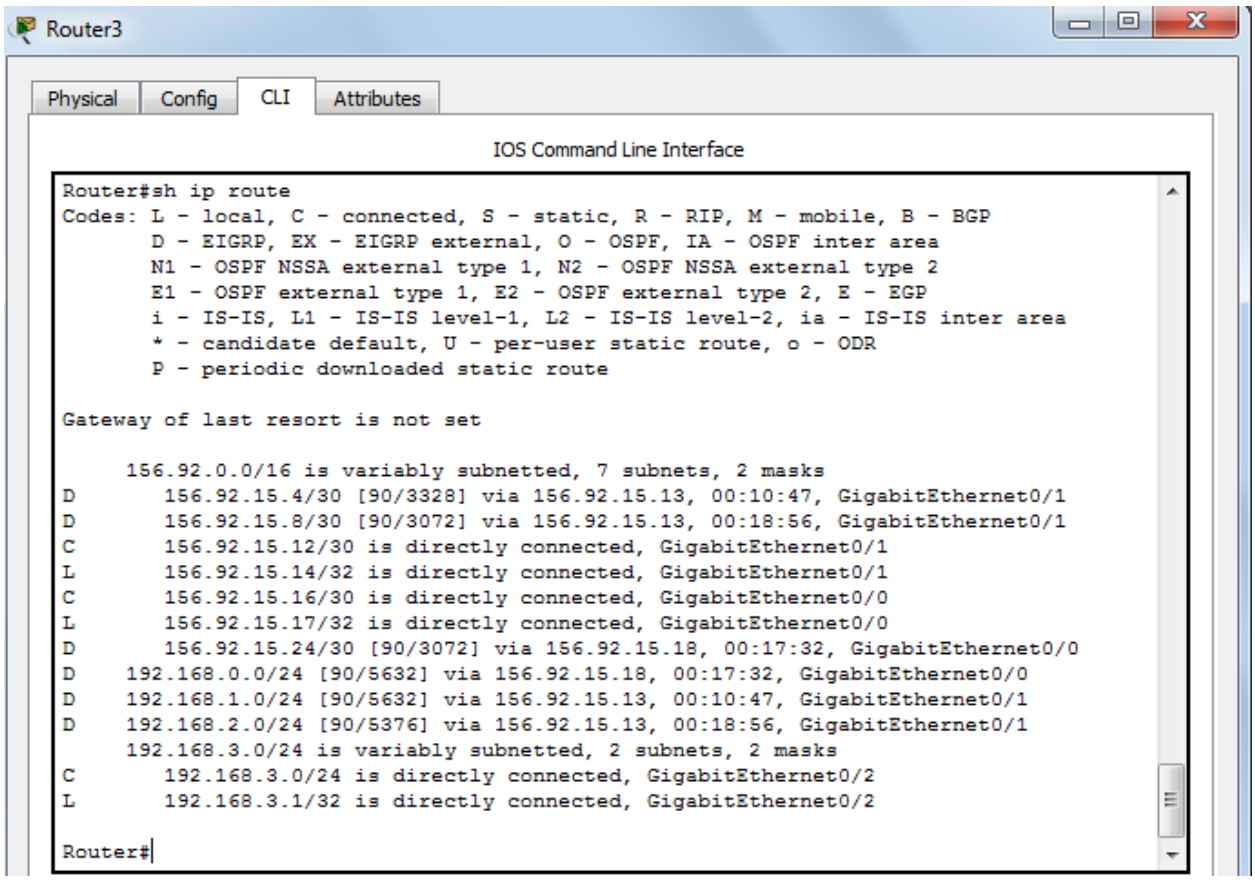

Рисунок 6.14 – Таблица маршрутизации маршрутизатора EIGRP

6.3 Объединение виртуальных сетей средствами третьего уровня

Как указывалось в параграфе 2.5, практически все современные коммутаторы поддерживают технологию VLAN – виртуальных локальных сетей. При этом трафик различных VLAN на канальном уровне полностью изолирован друг от друга, и для передачи данных из одной виртуальной сети в другую необходимы средства сетевого уровня. Особенности использования таких средств (маршрутизаторов и коммутаторов третьего уровня) для объединения VLAN рассмотрим в этом параграфе.

Рассмотрим сначала использование маршрутизаторов. Предположим, что имеется сеть, состоящая из коммутатора второго уровня, на котором организованы две VLAN. Для последующего объединения этих VLAN между собой один из свободных портов коммутаторов соединим с маршрутизатором, как это показано на рисунке 6.15.

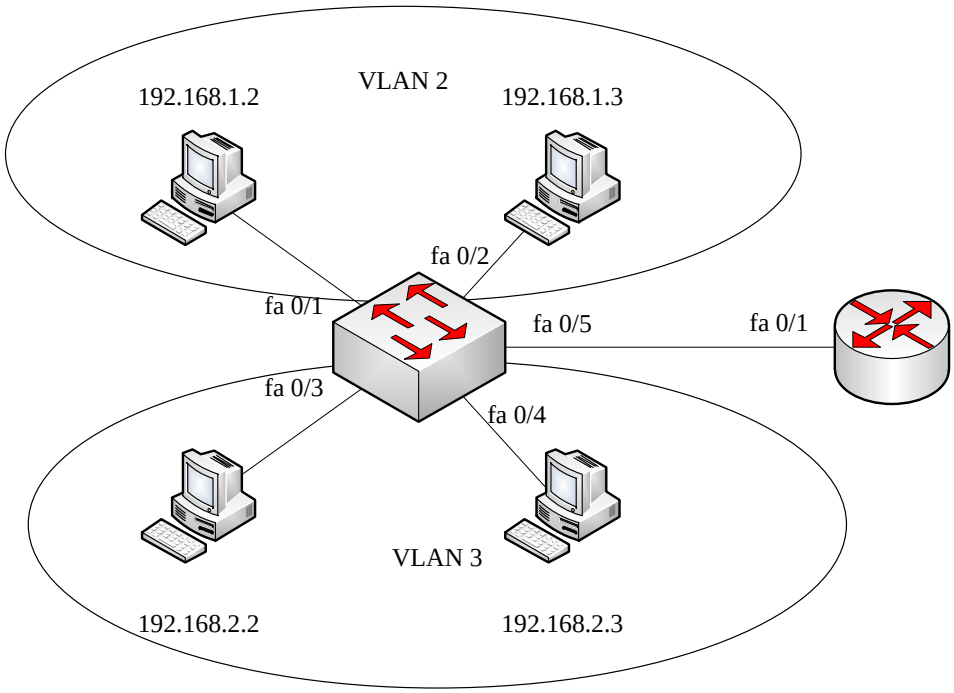

Рисунок 6.15 – Схема сети

Коммутатор сконфигурируем таким образом, чтобы порты fa 0/1 и fa 0/2 находились в VLAN 2, а порты fa 0/3 и fa 0/4 – в VLAN 3 (используются команды, рассмотренные в параграфе 2.5). Порт fa 0/5 переведем в магистральный (Trunk) режим.

Очевидно, что на данном этапе трафик виртуальных сетей будет полностью изолирован друг от друга. Для объединения VLAN между собой будем использовать интерфейс маршрутизатора. Так как интерфейс один, а подсетей две, разобьем интерфейс маршрутизатора на два субинтерфейса. Команды конфигурирования маршрутизатора будут иметь следующий вид:

Router(config)#interface fastEthernet 0/1

Router(config-if)#no shutdown

Router(config)#interface fastEthernet 0/1.2

Router(config-subif)#encapsulation dot1Q 2

Router(config-subif)#ip address 192.168.1.1 255.255.255.0

Router(config)#interface fastEthernet 0/1.3

Router(config-subif)#encapsulation dot1Q 3

Router(config-subif)#ip address 192.168.2.1 255.255.255.0

Router(config-subif)#exit

Видно что с использованием данных команд каждому из субинтерфейсов был присвоен свой IP-адрес, при этом в компьютерах каждой VLAN этот адрес должен быть прописан в качестве адреса шлюза по умолчанию. Команда encapsulation dot1Q 3 указывает маршрутизатору, что используется стандарт 802.1.Q, а также трафик какой виртуальной сети он должен обрабатывать.

Рассмотрим теперь использование для объединения виртуальных сетей коммутаторов третьего уровня.

Коммутатор третьего уровня – это, образно говоря, коммутатор с возможностями маршрутизации. По умолчанию это устройство работает как коммутатор, но при соответствующих настройках может использоваться и как маршрутизатор.

Рассмотрим сеть, представленную на рисунке 6.16.

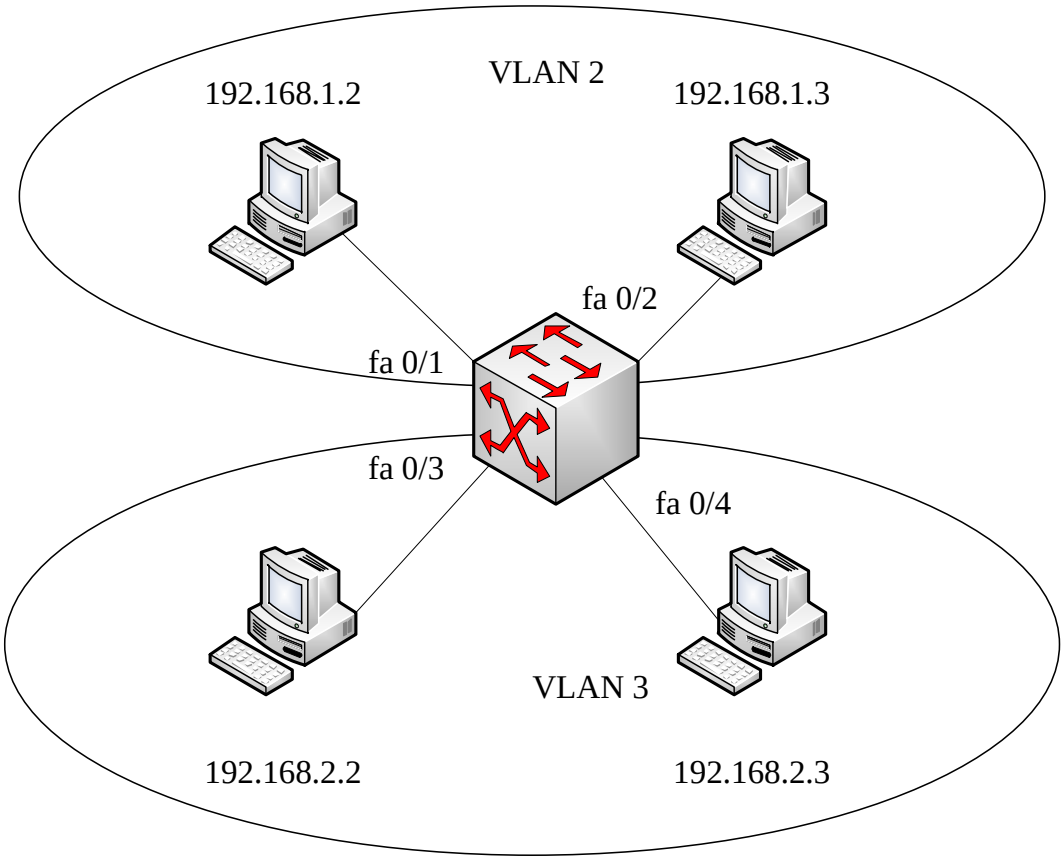

Рисунок 6.16 – Схема сети

Сначала, как и на обычном коммутаторе второго уровня, организуем две виртуальные сети и привяжем к ним порты:

Switch(config)#vlan 2

Switch(config-vlan)#name net_2

Switch(config)#vlan 3

Switch(config-vlan)#name net_3

Switch(config)# interface fastEthernet 0/1

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 2

Switch(config)#interface fastEthernet 0/2

Switch(config-if)#switchport mode access

Switch(config-if)#switchport access vlan 3

Для того, чтобы перейти к маршрутизации, интерфейсам необходимо задать IP-адреса. В коммутаторах третьего уровня IP-адреса даются не физическим, а логическим портам, в качестве которых выступают VLAN. Таким образом, в нашем случае необходимо войти в режим конфигурирования логического интерфейса VLAN, после чего задать им нужные адреса:

Switch(config)#interface vlan 2

Switch(config-if)#no shutdown

Switch(config-if)#ip address 192.168.1.1 255.255.255.0

Switch(config)#interface vlan 3

Switch(config-if)#no shutdown

Switch(config-if)#ip address 192.168.2.1 255.255.255.0

Switch(config)#ip routing

Как можно заметить принципы маршрутизации между VLAN на коммутаторе третьего уровня и на маршрутизаторе достаточно схожи, только в случае использования коммутатора необходимо присваивать IP-адреса не субинтерфейсам, а VLAN-интерфейсам (объединяющим группы физических

интерфейсов коммутатора), кроме того, необходимо дополнительно включать маршрутизацию между VLAN с использованием команды ip routing.

7 Технологии транспортного уровня

7.1 Структура сегментов протоколов TCP и UDP

В параграфе 1.3 отмечалось, что поток данных, сформированный протоколом прикладного уровня, передается на нижележащий транспортный уровень. На транспортном уровне основными (но не единственными) протоколами являются протокол управления передачей (Transmission Control Protocol – TCP) и протокол дейтаграмм пользователя (User Datagram Protocol – UDP). На транспортном уровне формируется сегмент (TCP- или UDP-сегмент) с указанием номера порта, идентифицирующего создавший поток прикладной протокол. После инкапсуляции сегмента в пакет на уровне межсетевого взаимодействия он передается с использованием технологий, описанных в предыдущих главах. Соответственно, в данной главе рассмотрим технологии транспортного уровня.

Протокол UDP, описанный в RFC 768 [11], предоставляет прикладным процессам простейшие услуги транспортного уровня. К функциям, выполняемым UDP, относятся мультиплексирование и демультиплексирование потоков различных прикладных процессов на основе номеров портов, а также контроль ошибок при передаче сегментов. При этом при возникновении ошибки она только диагностируется с использованием контрольной суммы, но не исправляется, сегмент с обнаруженной ошибкой попросту отбрасывается. Протокол UDP, как указывалось в первой главе, относится к протоколам без установления соединения и не гарантирует доставку сегментов по назначению. Однако это обстоятельство делает UDP более «быстродействующим» протоколом по сравнению с TCP, так как отсутствуют временные затраты на ожидание подтверждений на переданные сегменты [3].

С учетом этих особенностей сегмент UDP (UDP-дейтаграммы) имеет достаточно простую структуру и содержит следующие поля:

- номер порта отправителя;

- номер порта получателя;

- длина сегмента;

- контрольная сумма.

Протокол TCP (RFC 793) ориентирован на соединение и гарантирует доставку сегментов. Для обеспечения этого в протоколе имеется ряд механизмов, для рассмотрения которых необходимо сначала рассмотреть структуру TCP-сегмента. Структура заголовка сегмента приведена на рисунке 7.1, заимствованном из [15].

<table>
<tr><td colspan="8">1-й байт</td><td colspan="8">2-й байт</td><td colspan="8">3-й байт</td><td colspan="8">4-й байт</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="16">Source Port</td><td colspan="16">Destination Port</td></tr>
<tr><td colspan="32">Sequence Number</td></tr>
<tr><td colspan="32">Acknowledgement number</td></tr>
<tr><td colspan="4">Data offset</td><td colspan="6">Reserved</td><td colspan="6">Control bits</td><td colspan="16">Window</td></tr>
<tr><td colspan="16">Checksum</td><td colspan="16">Urgent pointer</td></tr>
<tr><td colspan="24">Options</td><td colspan="8">Padding</td></tr>
</table>

Рисунок 7.1 – Структура заголовка TCP-сегмента

Рассмотрим более подробно поля, входящие в заголовок TCP-сегмента.

Source Port и Destination Port – это номера портов источника и получателя сегмента.

Sequence Number – номер последовательности – это номер в потоке первого байта данного сегмента. Все передаваемые в потоке байты нумеруются, начиная с начального порядкового номера последовательности (Initial Sequence Number – ISN) для того, чтобы можно было организовать передачу подтверждения.

Acknowledgement number (AckN) – номер байта, на который отправляется подтверждение.

Data offset – длина заголовка – содержит длину заголовка сегмента, измеряемую в 32-битных словах.

Reserved – резервное поле – неиспользуемое в настоящее время поле, заполняется нулями.

Control bits – биты управления, часто называемые флагами. В TCP-сегменте используются следующие флаги:

- указатель срочности (Urgent Pointer – UGR) – устанавливается в единицу, если используется поле Urgent pointer (см. ниже);

- подтверждение (Acknowledgement – Ack) – устанавливается в единицу, если задействуется поле Acknowledgement number;

- выталкивание (Push – PSH) – в настоящее время практически не используется;

- сброс (Reset – RST) – используется для отмены TCP-соединения.

Синхронизация (Synchronize – SYN) – устанавливается в единицу при инициировании TCP-соединения.

Завершение (Finished – FIN) – используется для завершения TCP-соединения.

Window – размер окна – используется для управления потоком, более подробно это поле будет рассмотрено в параграфе 7.3.

Checksum – контрольная сумма – используется для обнаружения ошибок в принятых сегментах.

Urgent pointer – указывает на смещение в байтах от текущего порядкового номера байта до байта, имеющего статус срочности. Используется совместно с управляющим битом UGR.

Options – дополнительные поля, расширяющие возможности стандартного заголовка.

Padding – выравнивание – поле, используемое для доведения размера заголовка до целого числа 32-разрядных слов.

7.2 Установление и завершения соединений в протоколе TCP

Как указывалось выше, TCP является протоколом, ориентированным на установление соединения. Поэтому аналогично другим таким же протоколам

перед передачей данных должно быть установлено логическое соединение (не путать с соединением в сети с коммутацией каналов, параграф 1.1) между клиентом и сервером (TCP является клиент-серверным протоколом). Установление такого соединения основано на процедуре трехэтапного приветствия (three-way handshake). Кратко рассмотрим эту процедуру.

1. Клиент отсылает серверу сегмент с установленным в единицу управляющим битом SYN и начальным номером последовательности ISN, начиная с которого в передаваемом потоке будут нумероваться байты. В качестве ISN используется сгенерированное клиентом случайное число.

2. Сервер, получив от клиента сформированный на первом этапе сегмент, сохраняет начальный номер ISN, генерирует свой ISN, начиная с которого он будет нумеровать отправляемые в рамках этого соединения байты, и отправляет свой сегмент с установленными в единицу управляющими битами SYN и Ack. При этом в поле номер последовательности Sequence Number помещается сгенерированный ISN, а в поле номер подтверждения Acknowledgement number – номер принятого от клиента ISN, увеличенный на единицу (ISN+1). Таким образом, на данном этапе одновременно устанавливается соединение с клиентом и отправляется подтверждение на принятый сегмент.

3. Клиент, приняв от сервера сегмент, сохраняет полученный от него начальный номер ISN и отправляет серверу сегмент с установленным в единицу битом Ack, подтверждая прием, после чего соединение считается установленным (ESTABLISHED).

После установления соединения между клиентом и сервером осуществляется информационный обмен, при этом прием данных подтверждается (квитируется) с использованием бита Ack и номера подтверждения Acknowledgement number. Важно понимать, что IP-пакеты, переносящие TCP-сегменты, могут передаваться по сети различными путями с использованием рассмотренных выше принципов маршрутизации, несмотря на то, что передаются они в рамках одного TCP-соединения.

После передачи очередного сегмента узел (сервер или клиент) запускает таймер Retransmission TimeOut – RTO, по истечении которого, если не поступило подтверждение, сегмент считается утерянным. Протокол TCP является дуплексным, то есть в рамках одного соединения регламентируется процедура обмена данными в обе стороны. Каждая сторона одновременно выступает и как отправитель, и как получатель. У каждой стороны есть пара буферов: один – для хранения принятых сегментов, другой – для сегментов, которые только еще предстоит отправить. Кроме того, имеется буфер для хранения копий сегментов, которые были отправлены, но квитанции о получении которых еще не поступили. Если на сегмент не поступило подтверждение в течение времени, определяемого таймером, он посылается повторно.

Если после повторной посылки TCP-пакета опять не будет получено его подтверждение за интервал времени RTO, то попытки послать TCP-пакеты будут повторены (до 12 раз), но каждый раз с экспоненциально возрастающим значением RTO. Только после неудачи всей серии повторных посылок связь между участниками TCP-соединения будет считаться аварийно закрытой.

Для корректного завершения соединения TCP также предусматривает определенную процедуру, которая включает в себя четыре этапа:

- одна из сторон, инициирующая завершение соединения, посылает сегмент с установленным в единицу управляющим битом FIN;

- вторая сторона подтверждает прием этого сегмента путем передачи своего сегмента с установленным битом Ack. После этого один из каналов двухстороннего соединения считается закрытым;

-вторая сторона посылает FIN-сегмент;

- первая сторона подтверждает прием FIN-сегмента, после чего второй канал двухстороннего соединения также считается закрытым.

7.3 Обеспечение надежности доставки в протоколе TCP

Как указывалось выше, для обеспечения надежности доставки сегментов протокол TCP использует квитанции (подтверждения). Механизм квитирования был известен достаточно давно, скорее всего, до появления первых спецификаций протокола TCP, а системы передачи информации, использующие этот механизм, назывались системами с обратной связью.

Основным недостатком таких систем являлось то, что повышение надежности передачи обеспечивалось за счет снижения скорости. Например, если пункт А посылает сообщение пункту В, то для передачи следующего сообщения ему необходимо дождаться подтверждения. Поэтому в протоколе TCP используется модификация механизма квитирования, получившая название «метод скользящего окна».

Суть метода сводится к следующему. Отправитель (назовем его условно узлом А) отправляет не одно сообщение, а сразу несколько, не дожидаясь прихода квитанций. Одновременно эти сообщения сохраняются в буфере узла А на тот случай, если квитанции получены не будут в течение интервала RTO, и их придется передать повторно. Количество сообщений, которые узел А имеет право передать без квитанций, называется размером окна. Например, если размер окна составляет 7 сообщений, узел А начинает последовательную передачу сообщений. Если к моменту передачи 5-го сообщения приходит квитанция на 1-е, оно стирается из буфера, соответственно, узел А имеет право передать сообщения со 2-го по 8-е. Таким образом, при приеме каждой квитанции окно как бы «скользит», что и определило название метода. Если же после отправки сообщений, число которых составляет размер окна, так и не пришла квитанция на первое сообщение, передача приостанавливается. Если при этом с момента передачи первого сообщения окна истек интервал RTO, сообщения выбираются из буфера и передаются повторно.

Особенностью реализации метода скользящего окна в протоколе TCP является то, что нумеруются не сообщения (сегменты), а байты. Как

указывалось в предыдущем параграфе, нумерация байтов ведется, начиная со сгенерированного случайного числа ISN+1. Например, если номер первого байта сегмента имеет номер 28750, а размер сегмента составляет 1460 байт, то номер первого байта следующего сегмента будет 28750+1460=30210. Соответственно, эти номера, помимо обеспечения надежности, используются для правильного формирования и последующей передачи потока прикладному уровню. Кроме того, принимающая сторона по номерам байтов может идентифицировать повторный прием сегмента или потерю сегмента.

В качестве квитанции получатель отправляет сегмент в поле, номер подтверждения которого помещает номер последнего байта в принятом сегменте, увеличенный на единицу. Поэтому один и тот же сегмент одновременно переносит и пользовательские данные, и квитанцию.

Пример передачи сегментов и квитанций в несколько упрощенном виде приведен на рисунке 7.2, заимствованным из [9].

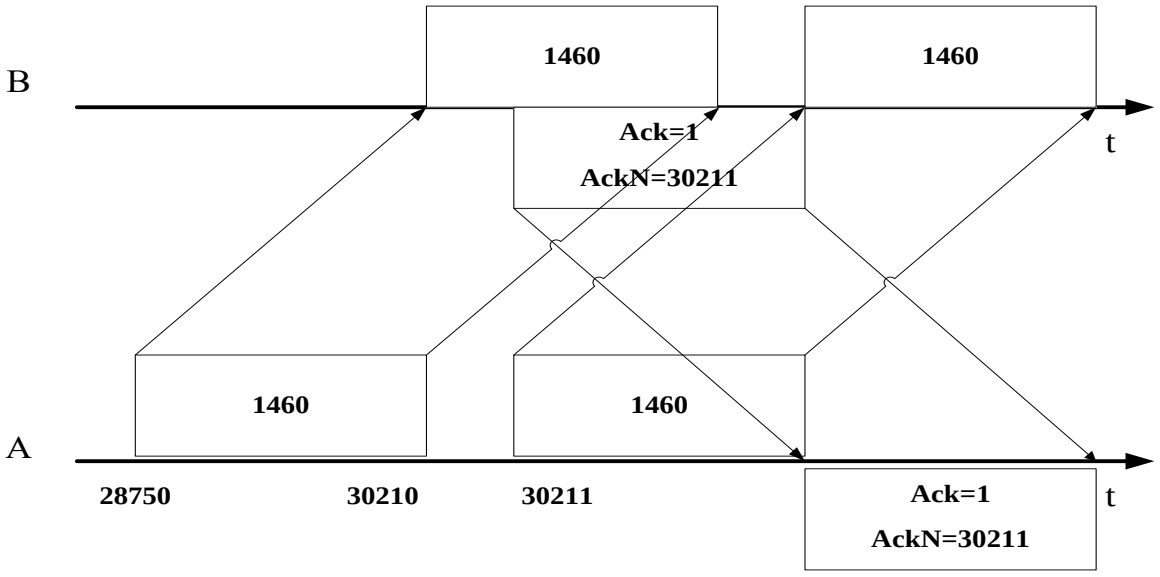

Рисунок 7.2 – Пример передачи сегментов и квитанций

На рисунке показана передача сегментов от узла А к узлу В (напомним, что на практике передача ведется в обоих направлениях). Узел В, приняв сегмент с начальным номером последовательности 28750 и с размером 1460 байтов, во встречном направлении (к узлу А) отправляет свой сегмент. Для

подтверждения в этом сегменте устанавливается в единицу бит Ack, а в поле номер подтверждения AckN заносится 30211. По существу это означает, что узел B ожидает следующий сегмент с номером последовательности, равным 30211. При этом узел A отправляет следующий сегмент не дожидаясь приема квитанции, а предыдущий сегмент, на который еще не была получена квитанция, хранится до ее приема в буфере.

Очевидно, что при размере окна W байт после приема подтверждения на N байт узел может посылать сегменты до тех пор, пока не будет отправлено $W + N$ байт.

Таким образом, протокол TCP гарантирует доставку сегментов до узла назначения. Однако может случиться так, что один из сегментов, из числа переданных в пределах окна, по каким-либо причинам не был доставлен получателю. В этом случае сегмент будет отправлен повторно, но на это может потребоваться недопустимо много времени. Поэтому протокол TCP не подходит для передачи потокового трафика.

7.4 Управление потоком в протоколе TCP

Как указывалось выше, протокол TCP обеспечивает не только гарантированную доставку сегментов, но и обеспечивает управление их потоком. Необходимость управления потоком в сети обусловлена следующим. При передаче сегментов с использованием рассмотренного выше метода скользящего окна необходимо определить несколько параметров. В первую очередь к таким параметрам относится размер окна W и интервал RTO. Очевидно, что задать заранее эти параметры нельзя, так как неизвестно, как далеко находятся друг от друга отправитель и получатель сегментов, через какие каналы связи будет передаваться сегмент, через какое количество маршрутизаторов и т.д. Другими словами, эти параметры должны быть различны для различных TCP-соединений. Более того, может оказаться так, что переданные сегменты успешно достигают пункта назначения, но объема свободной буферной памяти у этого узла окажется недостаточно для

хранения всех принятых сегментов. Тогда часть сегментов окажется просто отброшенной, что приведет к их повторной передаче. Очевидно, что это приведет к задержке передачи данных, а также к увеличению нагрузки на сеть за счет повторной передачи сегментов.

Из этого следует, что указанные выше параметры при установлении TCP-соединения должны назначаться динамически и учитывать особенности устанавливаемого соединения.

Рассмотрим данный процесс, естественно, в несколько упрощенном виде. При установлении соединения (параграф 7.2) каждая из сторон выделяет некоторый объем буферной памяти и передает в своем сегменте количество байт, которое она готова принять. Эта информация передается в том же сегменте, в котором передается начальный номер последовательности ISN, в поле заголовка сегмента Window (рисунок 7.1). Данное количество байт в протоколе TCP получило название «окно приема». Соответственно, узел A, получив значение окна приема, равное W, при передаче следующих сегментов следит за тем, чтобы количество отправленных, но не подтвержденных байт не превышало значения окна приема.

Однако из рассмотренной структуры заголовка TCP-сегмента следует, что значение окна приема передается во всех сегментах, а не только в тех, которые используются для установления соединения. Следовательно, окно приема можно изменить при передаче любого сегмента, то есть практически в любой момент. Например, если узел B (получатель) перегружен, он может передать узлу A сегмент со значением окна приема, равным нулю. Это приведет к тому, что узел A прекратит передачу сегментов до тех пор, пока не примет от узла B сегмент с ненулевым значением окна приема.

Кроме окна приема, необходимо в процессе установления соединения назначить значение еще одного важного параметра – размера сегмента. Этот размер определяется исходя из того, что в процессе передачи сегмента его нежелательно фрагментировать. Из этого следует, что сегмент должен целиком «помещаться» в PDU нижних уровней – в пакет и в кадр. Размер

сегмента, устанавливаемый в протоколе TCP по умолчанию, получил название MSS – Maximum Segment Size, значение которого устанавливается равным 536 байтам. Такое значение, в свою очередь, обусловлено значением IP-пакета по умолчанию – 576 байт.

Справедливости ради следует отметить, что величиной MSS можно управлять. Например, если промежуточными подсетями на пути передачи данных являются подсети Ethernet с размером кадра 1500 байт, желательно установить значительно больший MSS, например, 1460 байт. Это приведет к уменьшению накладных расходов на передачу данных, так как размер заголовка остается прежним, а количество данных, переносимых сегментом, увеличивается.

Вернемся к рассмотрению окна приема. При установлении соединения, когда клиент и сервер не уверены в качестве соединяющих их каналов связи, они устанавливают значение окна приема в один MSS. В процессе получения сегментов от партнера по установлению соединения значение окна приема увеличивается экспоненциально до некоторого порога, после чего возрастает линейно, но только при условии, что на сегменты приходят подтверждения. Как только подтверждение не приходит, размер окна приема уменьшается вдвое с каждым неподтвержденным сегментом. При получении подтверждения окно опять увеличивается и т.д.

Таким образом, в процессе информационного обмена клиент и сервер как бы подстраиваются под конкретные условия TCP-соединения. Для хранения данных о состоянии соединения создается база данных, называемая блоком управления передачей – TCB (Transmission Control Block). TCB хранит значения переменных, характеризующих состояние соединения.

В заключение отметим, что на компьютере под управлением ОС Windows текущие соединения и их состояния можно просмотреть с использованием команды netstat. Соединение может находиться в одном из следующих состояний:

- LISTEN – ожидание запроса на соединение;

- SYN-SENT – ожидание ответного запроса на соединение (когда собственный сегмент для установления соединения отправлен);

- SYN-RESEIVED – ожидание ответа после обмена запросами на соединение;

- ESTABLISHED – соединение установлено;

- FIN-WAIT – ожидание запроса на завершение соединения;

- CLOSING – ожидание ответа на запрос о завершение соединения;

- TIME-WAIT – ожидание для уверенности в том, что противоположная сторона получила подтверждение на свой запрос на завершение соединения;

- CLOSED – соединение завершено.

8 Вспомогательные протоколы и службы стека TCP/IP

8.1 Протоколы ARP и RARP

Как указывалось выше, по мере продвижения пакета по составной сети адреса канального уровня (локальные адреса) изменяются от одной подсети к другой, адреса сетевого уровня (IP-адреса) остаются неизменными. Маршрутизатор, декапсулировав пакет из принятого кадра, определяет по таблице маршрутизации интерфейс, на который его нужно продвинуть. На этом интерфейсе пакет инкапсулируется в кадр канальной технологии (например, Ethernet) и передается по подсети следующему маршрутизатору или конечному узлу. Таким образом, для передачи пакета через подсеть необходимо знать локальный адрес порта следующего маршрутизатора или конечного узла. Иначе говоря, по IP-адресу порта следующего маршрутизатора или конечного узла необходимо определить его локальный адрес (MAC-адрес, если подсеть использует Ethernet).

Эта задача решается с использованием протокола ARP (Address Resolution Protocol – протокол разрешения адреса).

В соответствии с данным протоколом все сетевые устройства содержат в своей памяти ARP-таблицу, связывающую сетевые и локальные адреса. Просмотреть содержимое этой таблицы на конечном узле, работающем под управлением ОС Windows, можно с использованием команды arp –a, рисунок 8.1.

Из рисунка 8.1 следует, что IP-адресу 192.168.0.1 соответствует MAC-адрес bc:f6:85:bf:d9:d2, в данном примере это адрес шлюза по умолчанию. Так как это таблица конечного узла, она содержит только одну запись, соответствующую маршруту по умолчанию. У маршрутизаторов таких записей несколько, однако их ARP-таблицы не могут содержать всех необходимых записей – их оказалось бы недопустимо много.

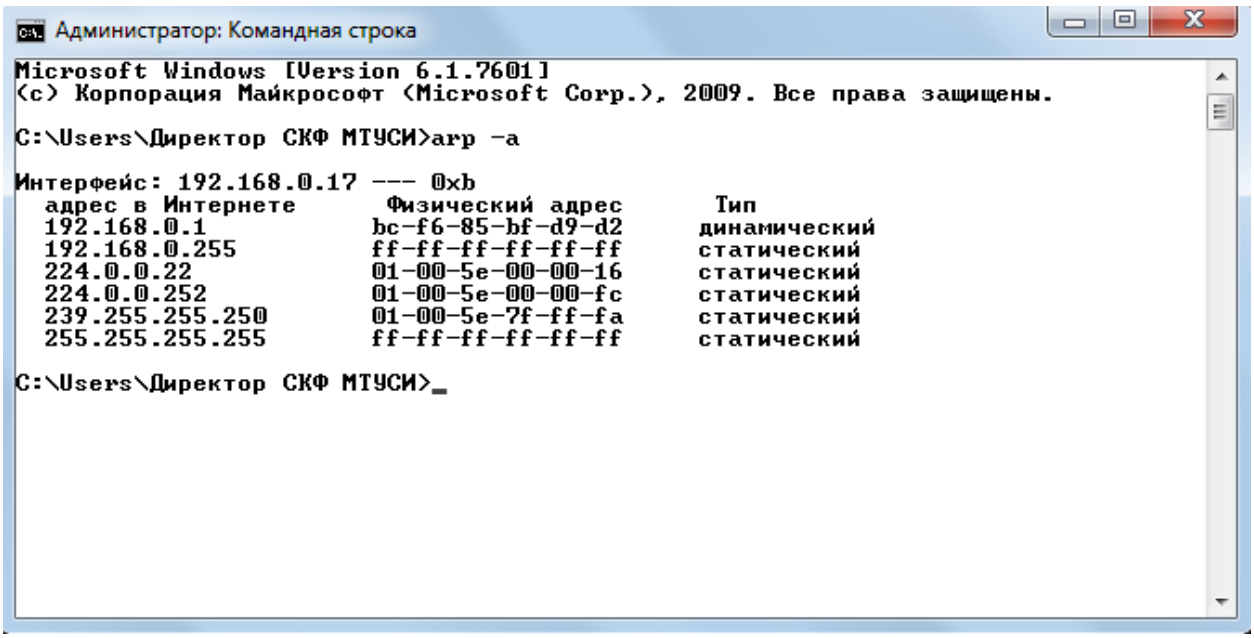

Рисунок 8.1 – Просмотр ARP-таблицы в ОС Windows

Поэтому при необходимости передачи пакета через подсеть протокол IP сначала обращается к ARP-таблице и пытается найти там нужный локальный адрес по заданному IP-адресу. Если такой записи в таблице нет, протокол ARP формирует ARP-запрос, инкапсулирует его в кадр канальной технологии (например, Ethernet), и широковещательно передает в пределах подсети. Запрос содержит IP-адрес порта назначения. Приняв запрос, устройство, чей IP-адрес совпадет с IP-адресом, содержащимся в запросе, формирует ARP-ответ, содержащий локальный адрес. Полученная в результате такого обмена информацией пара IP-адрес – локальный адрес записывается в ARP-таблицу (кэшируется), так как велика вероятность передачи следующего пакета по этому же адресу.

Необходимо отметить, что сетевые устройства могут пополнять свои ARP-таблицы не только путем приема ARP-ответов, но и путем приема широковещательных ARP-запросов, так как в них содержатся и сетевой, и локальный адреса отправителя запроса.

В ряде практических случаев возникает обратная задача – по локальному адресу определить IP-адрес. Например, при старте «тонкого клиента» – бездисковой рабочей станции – в начальный момент не знающей

своего IP-адреса, но знающей локальный адрес своего сетевого адаптера. Такую задачу решает реверсивный ARP – RARP. Информационный обмен между запросами и ответами в RARP аналогичен ARP.

8.2 Протокол DHCP

В рассмотренных в предыдущих главах примерах конфигурирования сетевых устройств конечным узлам необходимо было задать минимальную сетевую конфигурацию – IP-адрес, маску подсети и адрес шлюза по умолчанию. При этом данная информация вносилась вручную. Внесение этой информации является трудоемким рутинным процессом, особенно если число конечных узлов в сети достаточно велико (напомним, что конечные узлы – это не только компьютеры, а также любые другие оконечные устройства – IP-телефоны, IP-видеокамеры, принтеры, и т.д.). В результате возникает необходимость автоматизации процесса раздачи конечным узлам сетевых настроек.

Для решения этой задачи был разработан протокол DHCP (Dynamic Host Configuration Protocol). Реализация протокола зависит от канальной технологии, на базе которой построена сеть. Рассмотрим работу протокола DHCP в сети Ethernet (RFC 2131 и RFC 2132).

DHCP является клиент-серверным протоколом, при этом в качестве клиента выступает конечный узел, а в качестве сервера – DHCP-сервер. В соответствии с моделью взаимодействия клиент-сервер конечный узел запрашивает требуемые настройки, а сервер возвращает результат запроса. На практике реализация протокола DHCP не требует от сервера высокой производительности, поэтому DHCP-сервер обычно организуется либо на базе имеющегося в сети сервера приложений, либо на базе маршрутизатора.

Рассмотрим работу протокола на примере сети, представленной на рисунке 8.2.

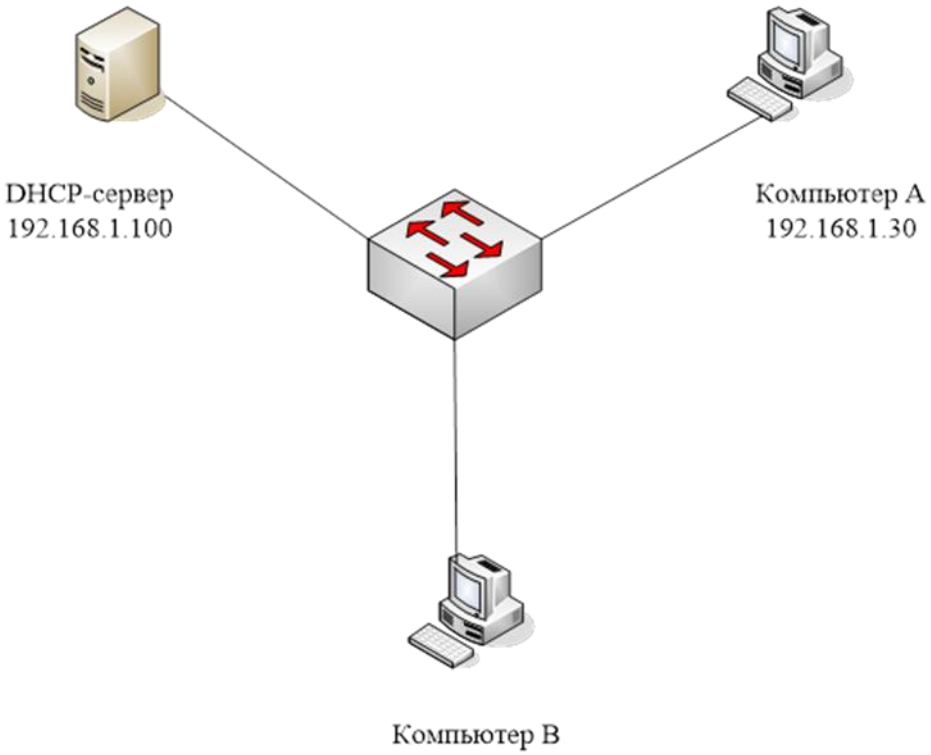

Рисунок 8.2 – Пример сети с DHCP-сервером

Как видно из рисунка, в данной сети имеется DHCP-сервер, коммутатор второго уровня и компьютер, который должен получить от сервера сетевые настройки. Если компьютер работает, например, на базе ОС Windows, для автоматической настройки сетевых параметров с использованием протокола DHCP необходимо под правами администратора зайти на вкладку «Свойства используемого подключения», дважды щелкнуть на пункте «Протокол Интернета (TCP/IP)», и в отобразившемся окне выбрать пункт «Получить IP-адрес автоматически» (рисунок 8.3). Использование DHCP в ОС Linux иллюстрируется рисунком 8.4.

В этом случае компьютер выступает в качестве клиента, который будет пытаться получить у сервера сетевые настройки. Для этого он формирует запрос на широковещательный адрес 255.255.255.255, а в качестве своего IP-адреса указывает адрес 0.0.0.0. Данный кадр называется DHCP DISCOVER, в нем указывается MAC-адрес узла, сформировавшего данный запрос.

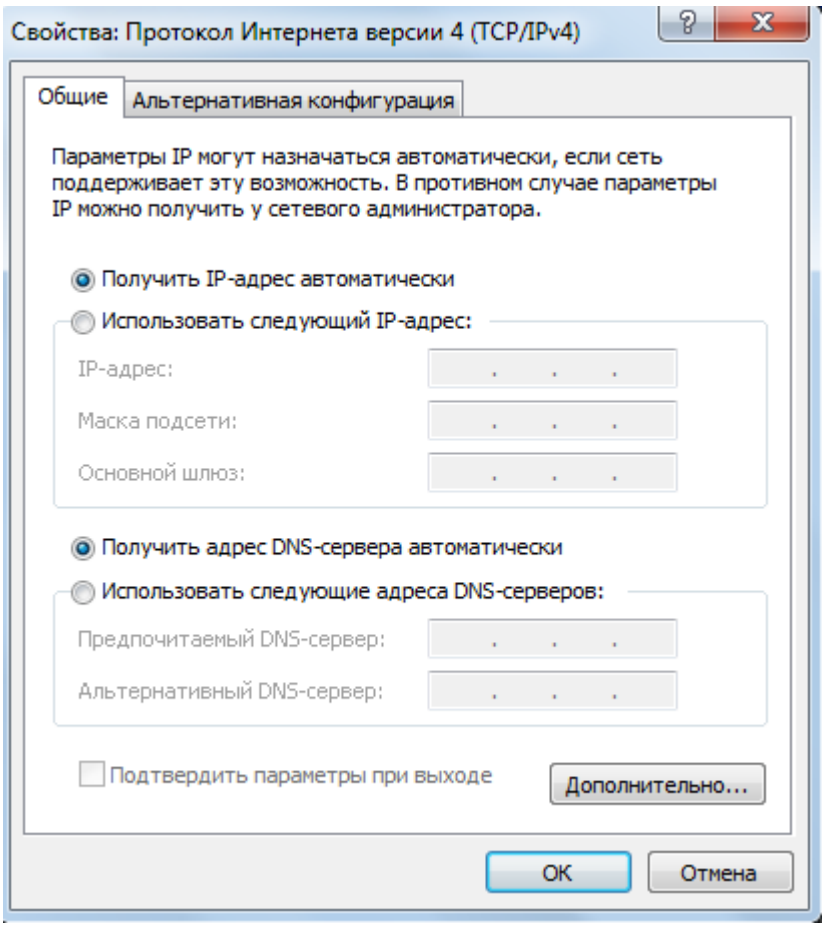

Рисунок 8.3 – Автоматическое получение сетевых настроек в ОС Windows

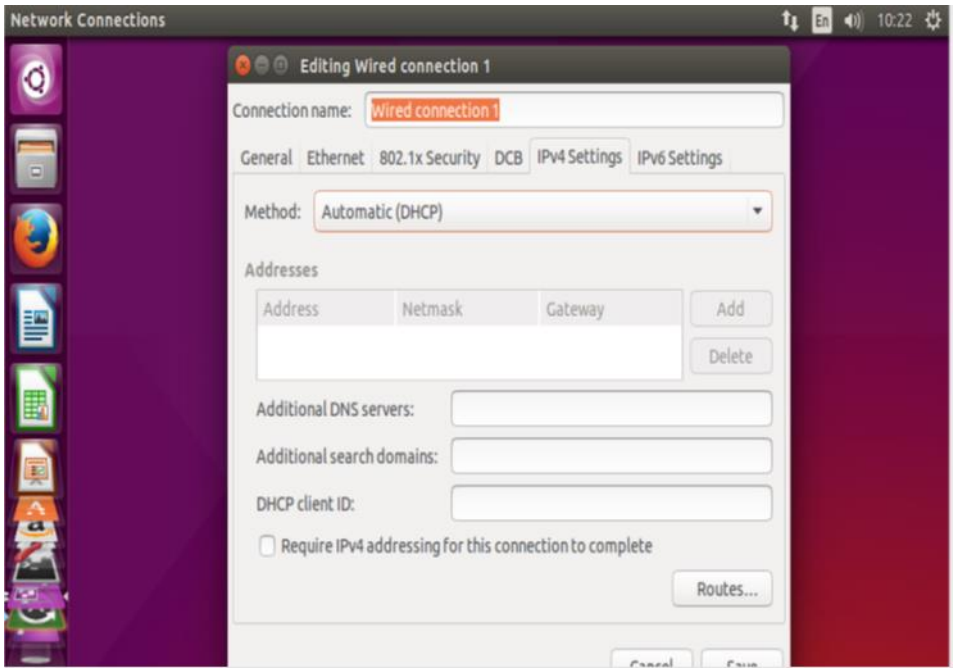

Рисунок 8.4 – Автоматическое получение сетевых настроек в ОС Linux

Так как переданный кадр-запрос является широковещательным, его получают все узлы сети, включая и DHCP-сервер, то есть в нашем примере запрос получают как компьютер А с адресом 192.168.1.30, так и сервер с адресом 192.168.1.100. Компьютер А запрос игнорирует, так как он не является DHCP-сервером.

Сервер, приняв запрос, выбирает в соответствии со своими настройками подходящую конфигурацию, и отправляет ее в кадре-ответе, который называется DHCP OFFER. В отличие от DHCP DISCOVER, сообщение DHCP OFFER является адресным, то есть пересылается не широковещательно, а на MAC-адрес узла, приславшего запрос.

Компьютер А, приняв сообщение DHCP OFFER, запоминает содержащиеся в нем сетевые настройки, и отсылает серверу сообщение DHCP REQUEST, имеющее такую же структуру, как и DHCP DISCOVER, но с указанием IP-адреса DHCP-сервера. Это сообщение также рассылается широковещательно (так как в сети может быть несколько DHCP-серверов).

Сервер, приняв сообщение DHCP REQUEST со своим IP-адресом, подтверждает это ответным сообщением DHCP ACK, и помечает выделенный IP-адрес как задействованный. Рабочая станция, получив сообщение DHCP ACK, применяет сохраненные ранее настройки.

Таким образом, в результате обмена служебными сообщениями протокола DHCP конечные узлы получают необходимые для дальнейшей работы сетевые настройки. Процесс обмена этими сообщениями иллюстрируется рисунком 8.5.

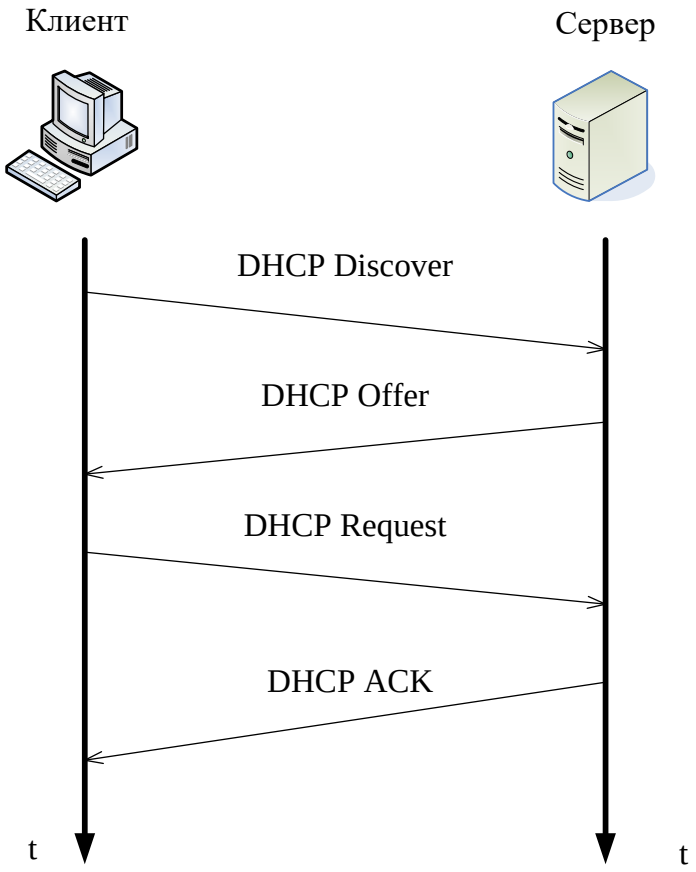

Рисунок 8.5 – Временная диаграмма, иллюстрирующая обмен сообщениями протокола DHCP

В свою очередь, DHCP-сервер также нуждается в конфигурировании – по меньшей мере, на нем должен быть задан пул адресов, которые он должен выдавать рабочим станциям, и время аренды, на которое выдается IP-адрес. Если DHCP-сервер организуется на базе маршрутизатора, объединяющего несколько подсетей, при конфигурировании протокола DHCP необходимо указать интерфейс, который будет использоваться для раздачи сетевых настроек в пределах подсети.

8.3 Конфигурирование DHCP на маршрутизаторе

Рассмотрим пример настройки DHCP-сервера, развернутого на базе маршрутизатора Cisco. Используя Cisco Packet Tracer, построим простейшую сеть, представленную на рисунке 8.6.

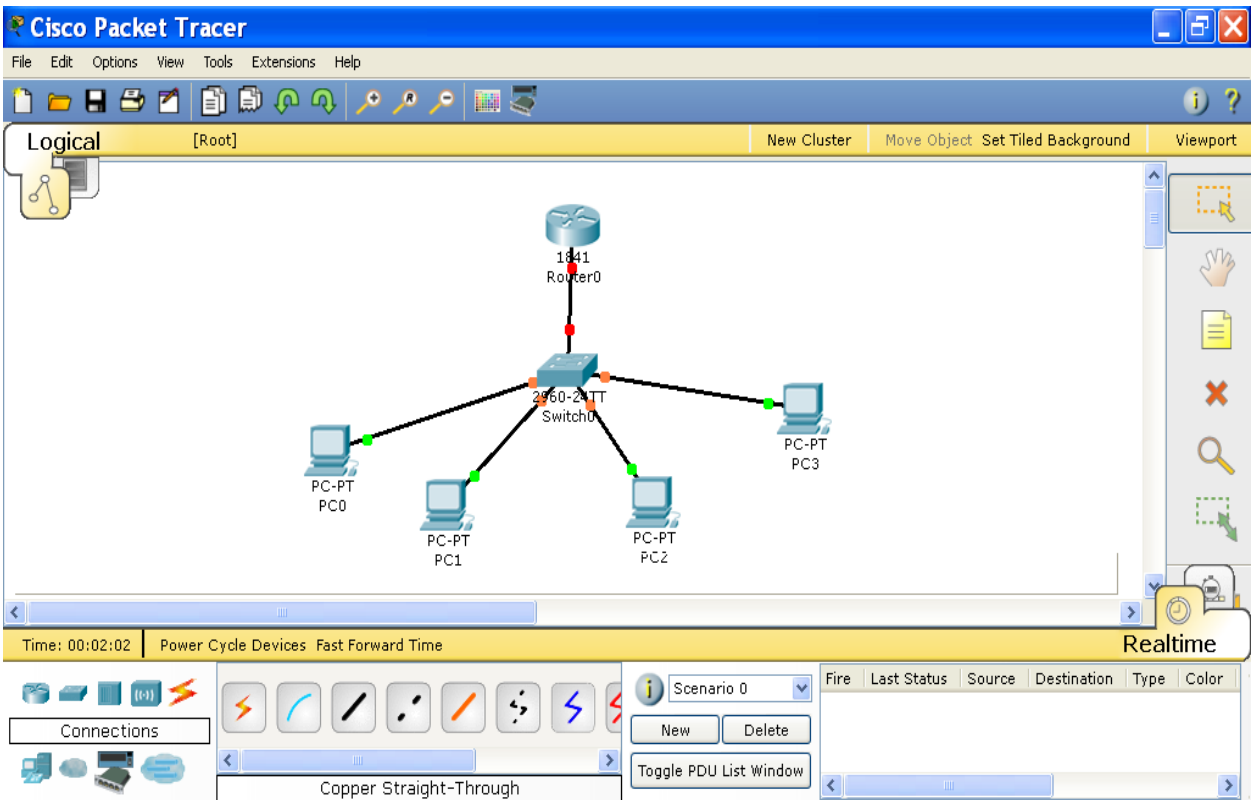

Рисунок 8.6 – Пример сети

На рабочих станциях необходимо выбрать пункт DHCP на вкладке IP Configurations. На реальном компьютере, работающем, например, под управлением ОС Windows, необходимо выбрать режим автоматического получения настроек, как это показано на рисунке 8.3.

Зададим внутреннему порту маршрутизатора IP-адрес, рисунок 8.7.

По умолчанию на устройствах Cisco функция DHCP-сервера включена. Если же она ранее была выключена, включить ее можно командой

Router(config)# service dhcp

выполняемой в режиме глобального конфигурирования.

Для конфигурирования DHCP-сервера используются следующие команды:

Router(config-if)#ip dhcp subnet_0 - создание пула адресов с именем subnet_0;

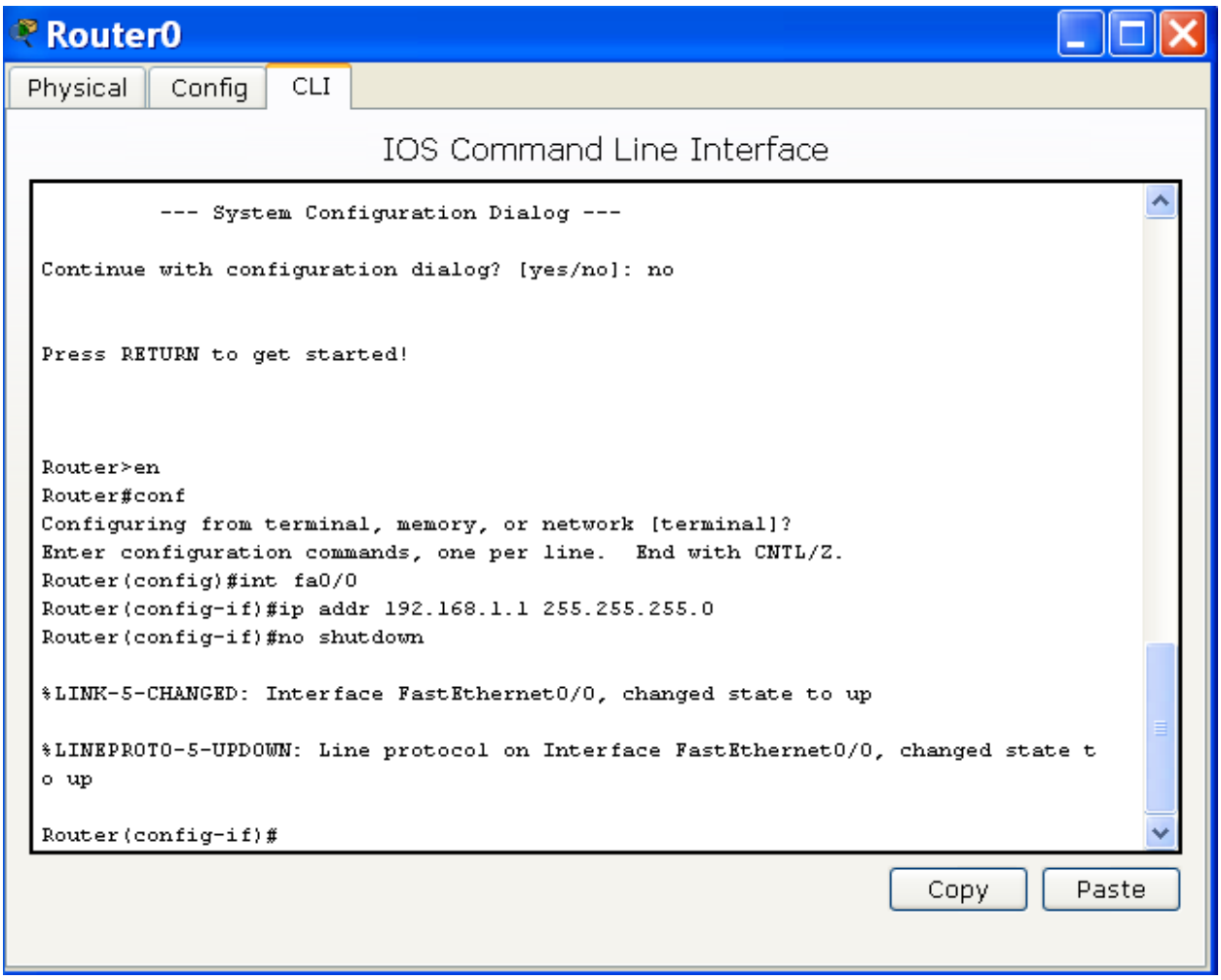

Рисунок 8.7 – Конфигурирование маршрутизатора

Router(dhcp-config)#network 192.168.1.0 255.255.255.0 – объявление адреса сети, из которой будут раздаваться IP-адреса;

Router(dhcp-config)#default-router 192.168.1.1 – назначение адреса шлюза по умолчанию;

Router(dhcp-config)#ip dhcp excluded-addr 192.168.1.1 – исключение адреса (или адресов) из пула раздаваемых адресов.

Конфигурирование маршрутизатора иллюстрируется рисунком 8.8.

```
Router>enable
Router#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface fa 0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#ip dhcp pool subnet_0
Router(dhcp-config)#network 192.168.1.0 255.255.255.0
Router(dhcp-config)#dns-server 192.168.1.30
Router(dhcp-config)#default-router 192.168.1.1
Router(dhcp-config)#exit
Router(config)#ip dhcp excluded-address 192.168.1.1
Router(config)#
```

Рисунок 8.8 – Конфигурирование DHCP на маршрутизаторе

Зайдя на вкладку IP Configurations любого компьютера, убедимся, что компьютер получил от сервера сетевые настройки, рисунок 8.9.

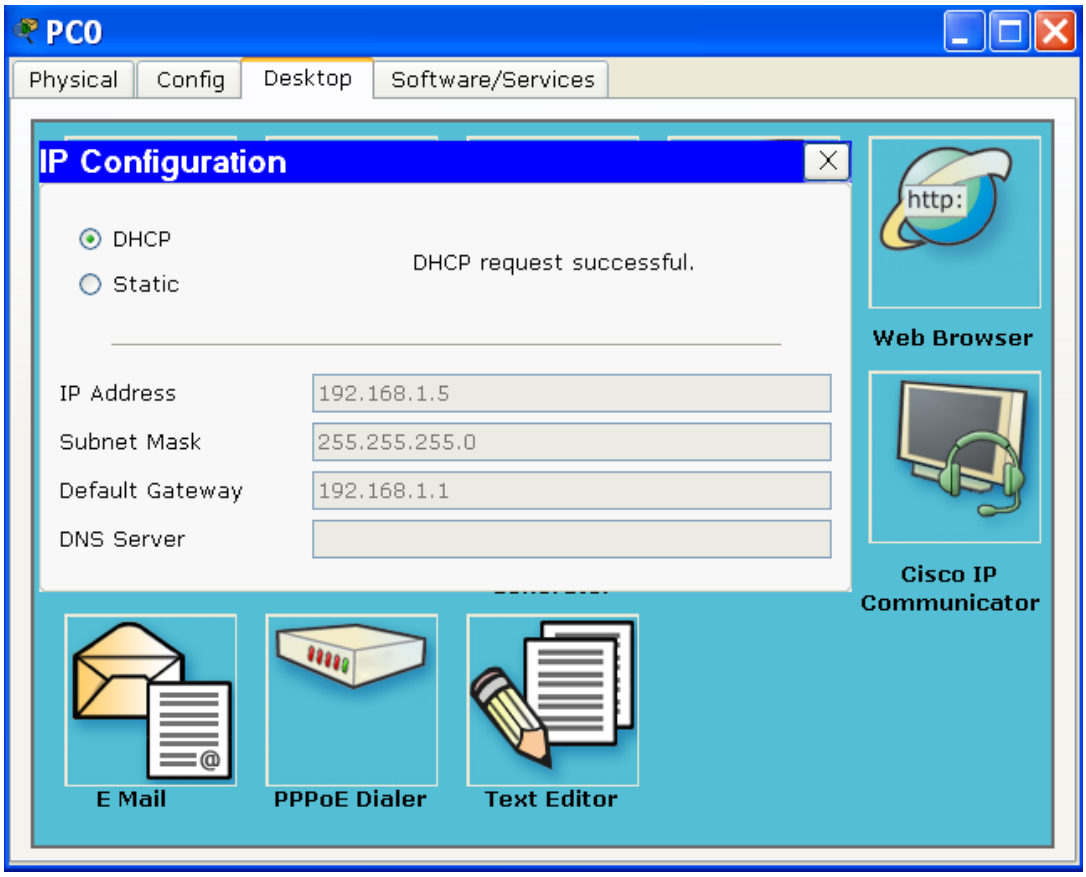

Рисунок 8.9 – Получение рабочей станцией сетевых настроек

8.4 Служба DNS

Выше отмечалось, что на сетевом уровне используются три типа адресов – локальные, сетевые и символьные доменные имена.

Для обеспечения соответствия локального адреса IP-адресу используется рассмотренный протокол ARP. Аналогично необходимо обеспечить соответствие IP-адреса символьному доменному имени. За обеспечение этого соответствия отвечает служба доменных имен – DNS (Domain Name Service).

Так же, как и ARP, DNS работает по схеме взаимодействия клиент-сервер. В качестве клиентов выступают конечные узлы, которым необходимо для передачи пакетов отобразить доменное имя на IP-адрес. Клиент обращается с запросом к DNS-серверу, который возвращает результат в виде запрашиваемого IP-адреса.

Такая схема взаимодействия клиента и сервера хорошо работает только в относительно небольших сетях. В случае же, если пользователи компьютеров сети, например, сотрудники предприятия, имеют возможность неограниченного «перемещения» по Интернету, база, которую необходимо хранить на сервере, окажется недопустимо большой.

В связи с этим служба DNS опирается на иерархию доменов [3], и каждый из доменов содержит свой DNS-сервер. В этом случае сервер хранит только часть базы доменных имен, а также ссылки на адреса DNS-серверов своих поддоменов. В этом случае для разрешения доменного имени применяется рекурсивная процедура, суть которой сводится к следующему.

Клиент обращается с запросом к «своему» DNS-серверу. В случае, если запрашиваемый адрес присутствует в базе сервера, клиенту возвращается ответ с запрашиваемым IP-адресом. В противном случае сервер обращается к DNS-серверу, расположенному в иерархии доменов на один уровень выше. Этот сервер тоже возвращает результат запроса, если запрашиваемый адрес присутствует в базе, или обращается с запросом к вышестоящему серверу, если запрашиваемого адреса в базе нет. Таким образом, от какого-то из DNS-

серверов поступает ответ, каждый из серверов, посылавших запросы, кэширует принятую информацию, и пересылает ее «ниже», пока результат не достигнет клиента. Клиент также кэширует полученные данные для ускорения последующей передачи пакетов по этому же адресу.

При конфигурировании сети в составе сетевых настроек, получаемых рабочими станциями с использованием протокола DHCP, указывается адрес «своего» DNS-сервера. При конфигурировании маршрутизатора Cisco соответствующая команда выглядит следующим образом:

Router0(dhcp-config)#dns-server <IP-адрес>

Рассмотрим пример совместной работы протоколов DHCP и DNS на примере сети, представленной на рисунке 8.6. Для большей наглядности дополним сеть внешним HTTP-сервером, на котором расположим сайт с произвольным доменным именем. Кроме того, установим внешний DNS-сервер.

Тогда схема сети принимает вид, представленный на рисунке 8.10.

Конфигурирование маршрутизатора представлено на рисунке 8.11.

Команды, представленные на рисунке 8.11, здесь разбирать не будем, так как все они были рассмотрены выше.

Далее дадим адрес внешнему серверу (192.168.2.2) и настроим его в качестве HTTP-сервера.

После этого дадим адрес DNS-серверу (192.168.3.2) и настроим его соответствующим образом – необходимо на вкладке Services выбрать пункт DNS, ввести символьное имя сайта, расположенного на HTTP-сервере, поставить ему в соответствие IP-адрес (192.168.2.2).

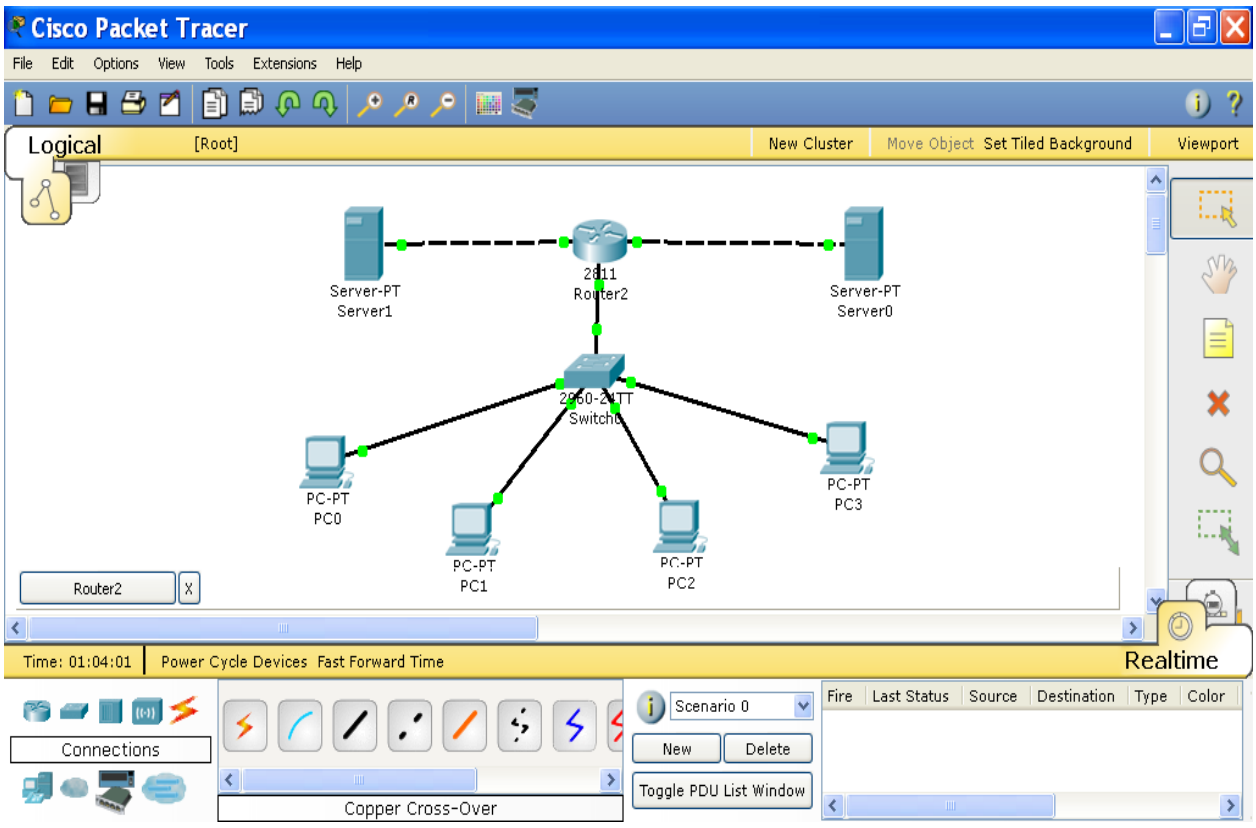

Рисунок 8.10 – Схема сети с серверами

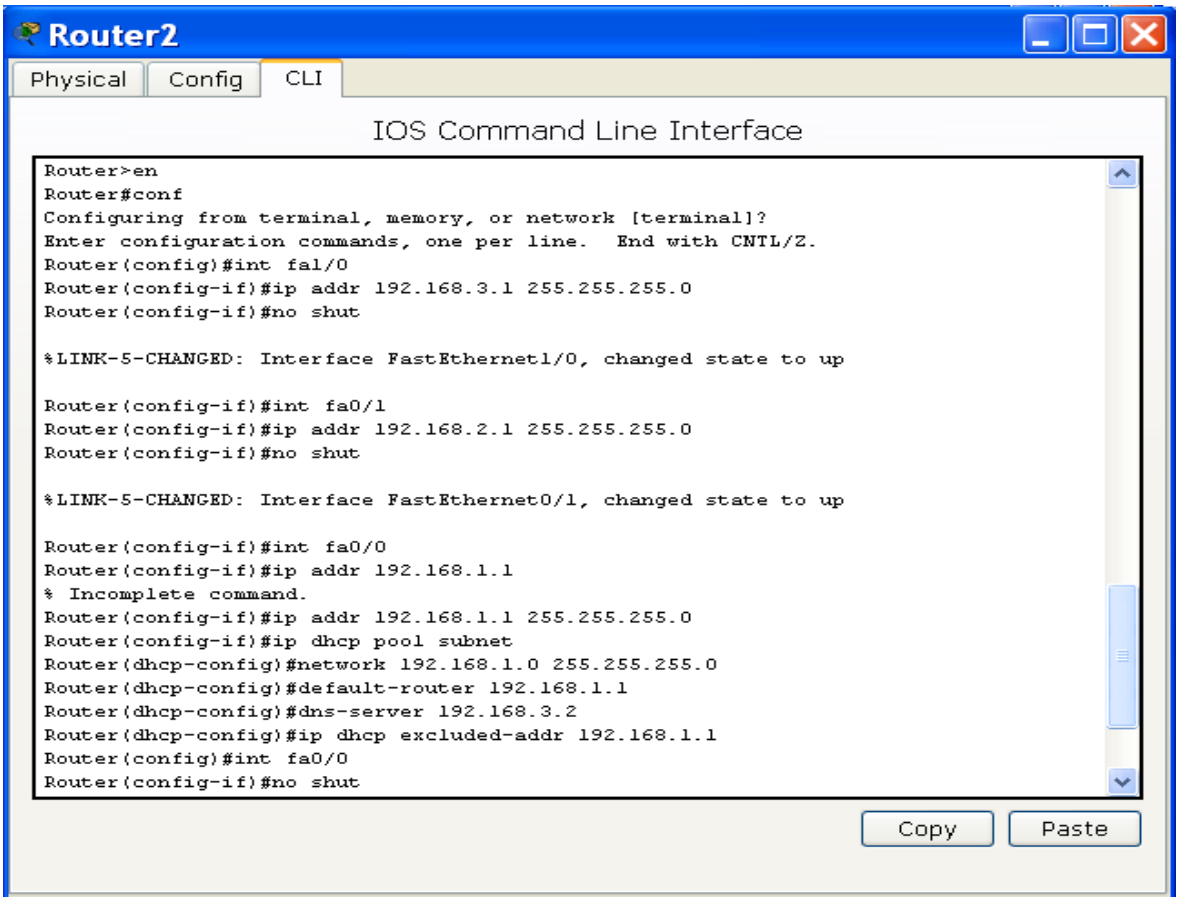

Рисунок 8.11 – Конфигурирование маршрутизатора

Если все сделано верно, то при просмотре сетевых настроек рабочих станций должна отобразиться информация, представленная на рисунке 8.12. Как видно из этого рисунка, рабочие станции получили именно те настройки, которые были сконфигурированы на маршрутизаторе при организации DHCP-сервера.

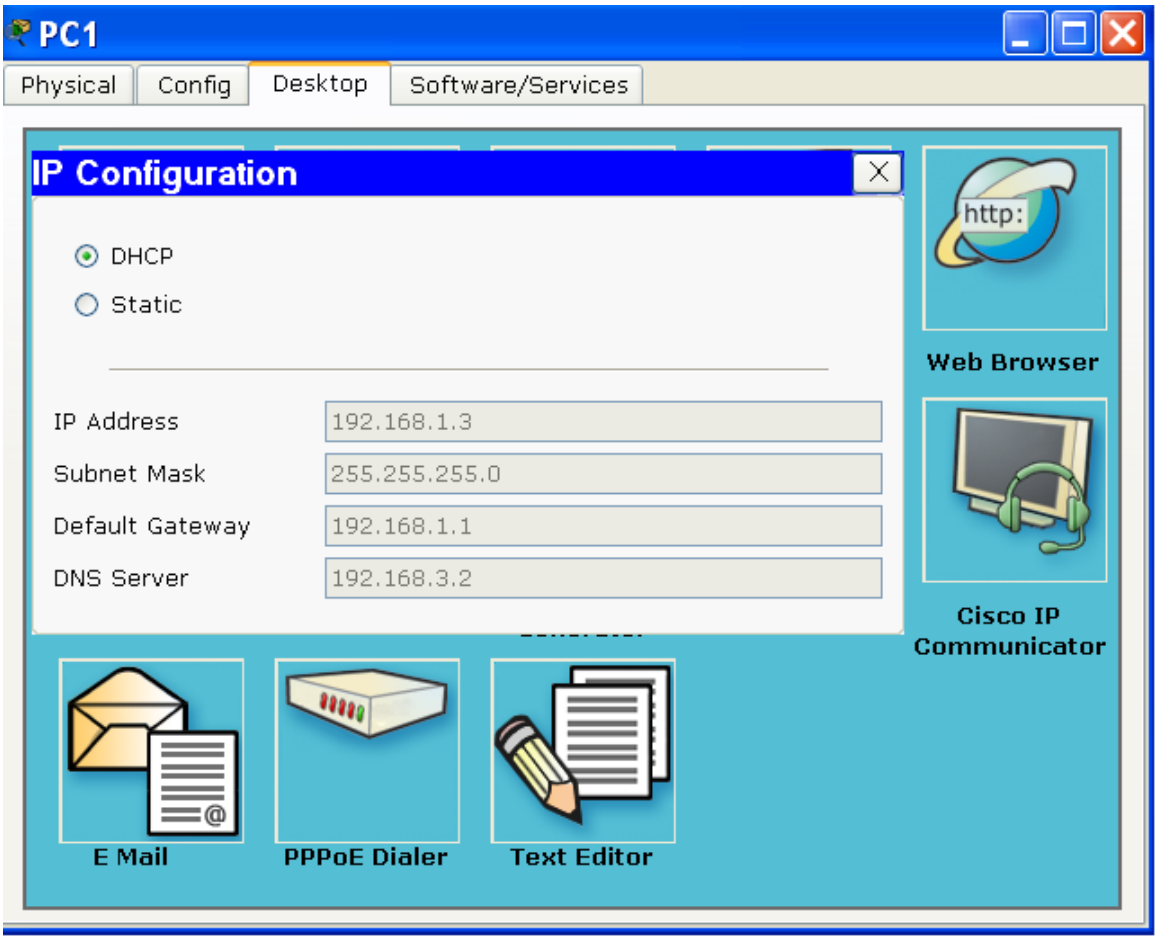

Рисунок 8.12 – Просмотр полученных сетевых настроек

При запуске на любой рабочей станции веб-браузера (вкладка Web Browser) на экране должна отобразиться размещенная на сервере веб-страница (рисунок 8.13).

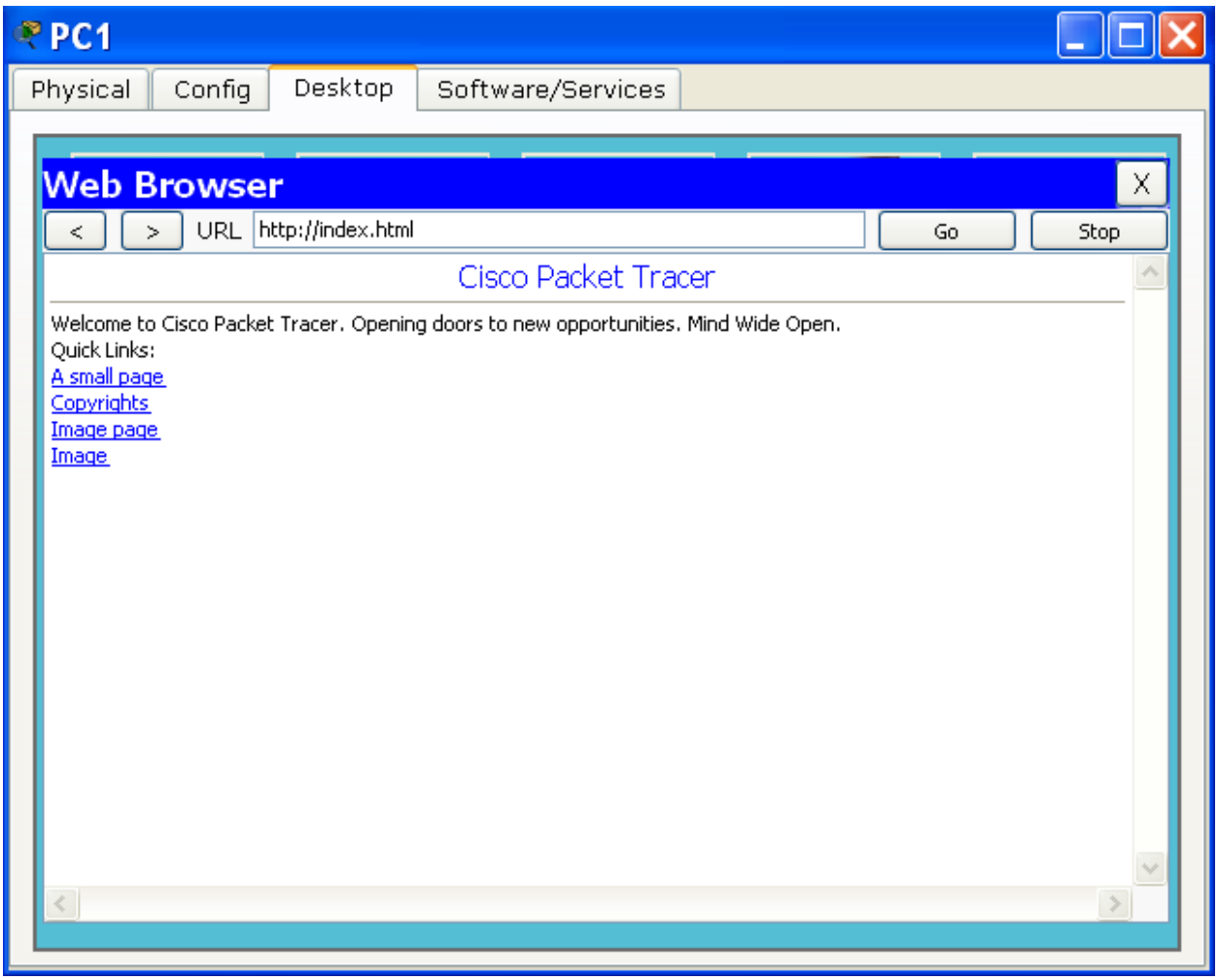

Рисунок 8.13 – Отображение веб-страницы

Таким образом, в рассмотренном примере после получения сетевых настроек и запуска веб-браузера с указанным доменным именем (index.html) рабочая станция отослала запрос на DNS-сервер (адрес 192.168.3.2). Сервер возвратил ответ, указывающий, что доменному имени http://index.html соответствует IP-адрес 192.168.2.2. Рабочая станция сохраняет эту информацию в своем DNS-кэше, и производит обращение к серверу по адресу 192.168.2.2. В ответ сервер пересылает рабочей станции веб-страницу.

8.5 Технология NAT

NAT расшифровывается как Network Address Translation – трансляция (преобразование) сетевых адресов. Рассмотрим сначала причины разработки этой технологии.

Во-первых, уже сейчас наблюдается дефицит IP-адресов четвертой версии. Кардинальным решением здесь может служить переход к шестой версии IP-протокола, но пока повсеместно используется IPv4. При использовании NAT в пределах внутренней сети могут использоваться частные адреса. К частным (немаршрутизируемым) адресам относятся адреса, входящие в специально выделенные для них диапазоны [3]:

10.0.0.1 – 10.255.255.254 для класса A (или с маской 255.0.0.0)

172.16.0.1 – 172.31.255.254 для класса B (или с маской 255.255.0.0)

192.168.0.1 – 192.168.255.254 для класса C (или с маской 255.255.255.0)

Частные адреса могут использоваться только в локальных сетях, магистральные маршрутизаторы такие адреса не обрабатывают, они работают только с общедоступными адресами.

Одни и те же частные адреса могут использоваться в различных локальных сетях, что и приводит к экономии адресного пространства. Преобразование частных адресов в общедоступные и обратно осуществляется с использованием NAT.

Во-вторых, NAT существенно повышает безопасность локальной сети, так как в этом случае извне сеть представляется единственным или несколькими общедоступными адресами. Поэтому определить структуру локальной сети, проанализировать данные, циркулирующие в ней, становится проблематично.

Основная идея технологии NAT состоит в следующем. Локальная сеть использует адресное пространство частных адресов. В маршрутизаторе или другом устройстве, связывающем локальную (внутреннюю) сеть с внешней IP-сетью, настраивается протокол NAT, осуществляющий при передаче во внешнюю сеть преобразование частного адреса в общедоступный и обратное преобразование при приеме. Так как внутренняя сеть также может содержать маршрутизаторы для разделения ее на подсети, они должны получать объявления о маршрутной информации от маршрутизаторов внешней сети. В свою очередь, внешние маршрутизаторы не должны ничего знать о

маршрутизаторах внутренней сети. Поэтому NAT-устройство должно пропускать из внешней сети во внутреннюю сообщения протоколов маршрутизации (RIP, OSPF и т.д.), но не пропускать эти сообщения в обратном направлении. Число общедоступных адресов чаще всего меньше числа частных адресов, за счет чего и достигается экономия адресного пространства. В частном, но далеко не самом редком случае, может использоваться всего один общедоступный адрес, настраиваемый на внешнем порту NAT-маршрутизатора.

Самой простой технологией является статический NAT. Суть его заключается в том, что в NAT-устройстве прописываются специальные таблицы трансляций (преобразований), жестко связывающие внутренние (частные) и внешние (общедоступные) адреса. Недостаток такой технологии очевиден – количество общедоступных адресов у NAT-устройства должно соответствовать количеству узлов внутренней сети, имеющих права доступа во внешнюю сеть. Как следствие, экономии адресного пространства не происходит.

Поэтому в настоящее время широко используется динамический NAT, суть которого рассмотрим на рисунке 8.14, заимствованным из [9].

На рисунке представлена внутренняя сеть, использующая частный адрес 192.168.1.0/24. Выход во внешнюю сеть организуется с использованием NAT-устройства, внешнему интерфейсу которого присвоен общедоступный адрес 80.254.97.169. Необходимо обеспечить всем четырем конечным узлам внутренней сети доступ к внешней сети, в частности, к web-серверу с адресом 213.180.193.11.

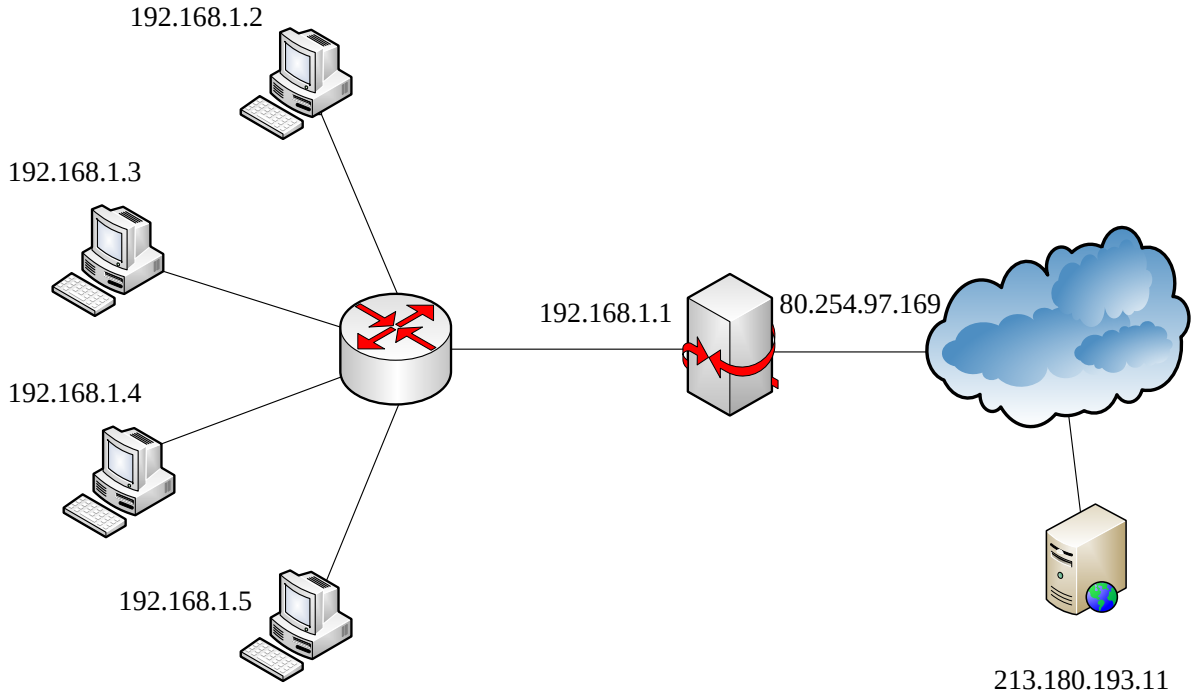

Рисунок 8.14 – Иллюстрация работы динамического NAT

Очевидно, что статический NAT для решения такой задачи непригоден, так как доступ узлов к внешней сети осуществляется с использованием единственного внешнего адреса (на практике внешних адресов также может быть несколько, но в любом случае количество внутренних узлов превышает количество внешних адресов).

При передаче пакета во внешнюю сеть NAT-устройство может подменить частный адрес отправителя на свой общедоступный адрес, как и в статическом NAT. Однако при приеме пакета-ответа из внешней сети от сервера необходимо определить, какому из внутренних конечных узлов этот пакет нужно передать. Или, другими словами, при приеме необходимо определить, на какой частный адрес нужно изменить общедоступный адрес назначения, содержащийся в ответном IP-пакете.

Таким образом, для надежного различения принимаемых пакетов NAT-устройством необходима, помимо IP-адресов, дополнительная информация. В качестве такой информации можно использовать номера портов TCP- или UDP-сегментов, переносимых IP-пакетами. Однако в нашем

примере все четыре узла могут обратиться с запросом к web-серверу с адресом 213.180.193.11, ответы которого будут иметь один и тот же номер порта 80 или 8080. Поэтому в данном случае используются так называемые назначенные номера портов. В качестве назначенных портов используются порты источника, которым в процессе передачи присваиваются значения, не стандартизированные в протоколах TCP и UDP. Назначенный порт может быть выбран произвольно, но с учетом того, что он должен быть уникален в пределах внутренней сети.

В соответствии с этим таблица NAT-устройства усложняется, в нее теперь должны входить не только IP-адреса, но и номера портов (таблица 8.1).

Поскольку описанная выше технология использует не только сетевые адреса, но и номера портов, она получила название NAPT (Network Address Port Translation) [5].

Таблица 8.1 – Примерный вид NAT-таблицы

Частный адрес	Порт	Общедоступный адрес	Назначенный порт
192.168.1.2	8080	80.254.97.169	61001
192.168.1.3	8080	80.254.97.169	61002
192.168.1.4	8080	80.254.97.169	61003
192.168.1.5	8080	80.254.97.169	61004

При передаче пакета, например, от узла 192.168.1.2 к серверу глобальной сети с адресом 213.180.193.11 в заголовок пакета в качестве адреса получателя будет указан 213.180.193.11, в качестве номера порта получателя – 8080. В качестве адреса отправителя будет указан 192.168.1.2, а в качестве номера порта отправителя – 8080. После приема этого пакета NAT-устройством будет произведена подмена адреса отправителя на 80.254.97.169, а номера порта отправителя на 61001. Эта информация динамически заносится в таблицу 8.1.

При приеме ответа от сервера глобальной сети будет выполнено обратное преобразование – адрес получателя будет заменен на 192.168.1.2. При этом в качестве номера порта получателя будет указан назначенный порт, который сервер укажет исходя из номера порта источника принятого сегмента. При этом NAT-устройство «поймет», какому из внутренних узлов передать пакет, используя номер назначенного порта.

Если NAT-устройство имеет несколько общедоступных адресов (пул адресов), то таблица 8.1 ведется динамически, то есть при передаче пакета запоминается, на какой именно адрес из пула была осуществлена подмена, и данная информация заносится таблицу. Эти действия, естественно, являются абсолютно прозрачными для конечных узлов.

8.6 Конфигурирование NAT на маршрутизаторе

Рассмотрим настройку протокола NAT для примера, представленного на рисунке 8.14, полагая, что в качестве NAT-устройства используется маршрутизатор Cisco.

Предположим, что в маршрутизаторе, используемом в качестве NAT-устройства, порт с адресом 192.168.1.1 является портом fa 0/0, а порт с адресом 80.254.97.169 – портом fa 0/1. В терминологии NAT порт fa 0/0 является внутренним портом (inside), а порт fa 0/1 – внешним портом (outside).

Пакеты, прибывающие на внутренний порт и подлежащие передаче на внешний порт, подлежат трансляции в соответствии с Source NAT (SNAT), то есть подмене подлежит IP-адрес источника (Source IP). Пакеты, прибывающие на внешний порт, подлежат трансляции в соответствии с Destination NAT (DNAT), то есть подмене подлежит IP-адрес получателя (Destination IP).

Сначала необходимо создать список доступа (подробнее списки доступа будут рассмотрены в следующем параграфе). Для этого в режиме

глобального конфигурирования необходимо выполнить следующую команду:

(config)# access-list 100 permit ip <адрес> <инвертированная маска> any

Забегая вперед, отметим, что данной командой создается список доступа с номером 100, разрешающий передавать пакеты с адресом источника, указанного в команде, на любые адреса. Списки доступа будут рассмотрены в следующем параграфе.

Пул адресов создается на маршрутизаторе в режиме глобального конфигурирования командой

(config)# ip nat pool <имя> <начальный адрес> <конечный адрес> netmask <маска>.

Если, как в нашем примере, используется единственный общедоступный адрес, начальный и конечный адреса в команде совпадают.

Затем назначаются внутренние и внешние интерфейсы командами:

(config)# interface fa 0/0;

(config-if)# ip nat inside (outside)

Включается NAT командой

ip nat inside source list 100 pool <имя>

Конфигурирование маршрутизатора Cisco с использованием указанных команд для нашего примера (рисунок 8.14) представлен на рисунке 8.15.

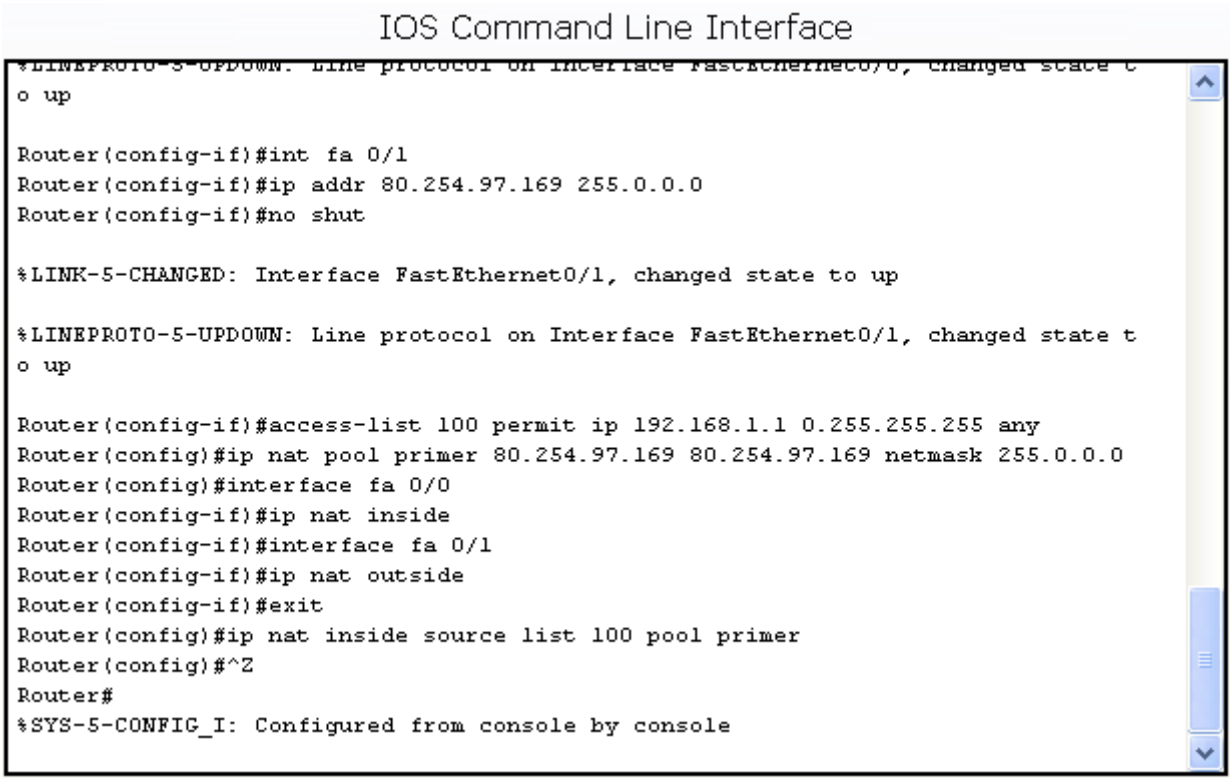

Рисунок 8.15 – Конфигурирование динамического NAT

После того, как какой-либо из внутренних узлов обменяется пакетами с внешней сетью, можно будет просмотреть трансляции адресов, произведенные NAT, с использованием команды **show ip nat translations** (рисунок 8.16).

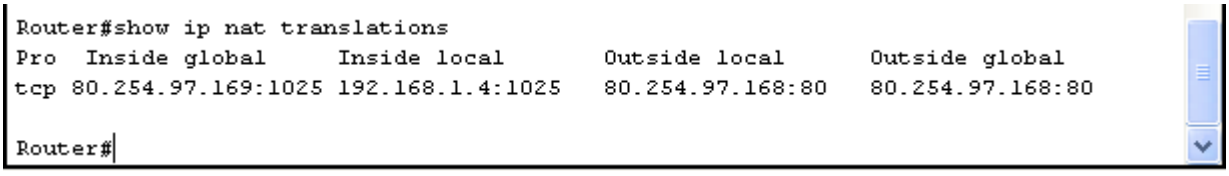

Рисунок 8.16 – Список трансляций после обращения к web-серверу

Из рисунка 8.16 следует, что была произведена одна трансляция, информация о которой представлена в четырех колонках.

Первая колонка указывает на транспортный протокол, в нашем случае это TCP.

Вторая колонка (Inside global) указывает на сокет (IP-адрес и номер порта), на который подменяется сокет отправителя.

Третья колонка (Inside local) указывает на внутренний IP-адрес отправителя с назначенным номером порта.

Четвертая колонка (Outside local) указывает на сокет узла назначения во внешней сети, который сформирован внутренним узлом-отправителем.

Пятая колонка (Outside global) указывает на IP-адрес и номер порта, используемые во внешней сети.

Таким образом, из рисунка 8.16 следует, что внутренний узел с адресом 192.168.1.4 направляет пакет web-серверу с адресом 80.254.97.168. Соответственно, IP-адрес и номер порта получателя, указанные в пакете:

80.254.97.168:80 (напомним, что для протокола HTTP используются порты 80 и 8080).

IP-адрес и порт источника в этом же пакете:

192.168.1.4:80 .

При передаче пакета во внешнюю сеть маршрутизатор подменяет IP-адрес и порт источника:

80.254.97.169:1025.

Соответственно, при приеме ответного пакета от сервера сокет 80.254.97.169:1025 будет изменен на 192.168.1.4:80, и пакет получит нужный узел внутренней сети.

В заключение параграфа отметим, что здесь показаны только базовые настройки протокола NAT, на практике используется гораздо большее число настроек. Однако, получив первоначальные навыки с использованием материала, представленного в данном параграфе, можно освоить и другие возможности, предоставляемые NAT.

8.7 Сетевые фильтры

Технология NAT, помимо экономии адресного пространства корпоративных сетей, используется также для повышения безопасности. Однако это далеко не единственный инструмент, позволяющий оградить сеть от нежелательного вмешательства извне. Практически все современные маршрутизаторы дают возможность сетевым администраторам использовать так называемые сетевые фильтры, позволяющие назначить определенные правила передачи того или иного трафика. Настройка сетевых фильтров основана на использовании списков доступа (Access Lists – ACL). Списки доступа предназначены для того, чтобы разрешать или запрещать продвижение тех или иных пакетов через маршрутизатор [8, 11].

Как указывалось выше, кадр канального уровня (Ethernet) переносит IP-пакет, который, в свою очередь, переносит сегмент. Следовательно, маршрутизатор может пропускать или фильтровать трафик исходя из значений, помещенных в заголовках канального, сетевого и (или) транспортного уровней. Кроме того, сегмент транспортного уровня с помощью номера порта переносит информацию об используемом прикладном протоколе. Соответственно, появляется возможность фильтрации или продвижения трафика, сформированного тем или иным прикладным протоколом.

Списки доступа используются для того, чтобы можно было разрешить (permit) или запретить (deny) передавать те или иные данные. Это очень похоже на создание «белых» и «черных» списков на телефоне [9]. При создании «белого» списка принимать разрешено только вызовы от источников, номера которых внесены в «белый» список, остальные вызовы отбрасываются. При использовании «черного» списка отбрасываются только вызовы от источников, внесенных в список, остальные вызовы принимаются. Основное отличие от телефонных вызовов состоит в том, что в списки permit и deny вносятся не телефонные номера, а значения заголовков различных уровней.

Запрет или разрешение на передачу того или иного вида трафика могут быть заданы для каждого из установленных на интерфейсе протоколов и для каждого из направлений передачи – исходящего и (или) входящего. По умолчанию, когда на маршрутизаторе не сформированы списки доступа, никаких ограничений на передачу данных не существует.

Список доступа ACL представляет собой набор условий, устанавливающих правила продвижения или фильтрации пакетов. Программное обеспечение маршрутизатора проверяет условия последовательно, начиная с первого. Например, если условие, разрешающее передачу пакета, находится первым, расположенные следом условия не смогут обеспечить фильтрацию, так как они проверяться маршрутизатором не будут. Однако если список создан, то в конце списка подразумевается наличие условия, запрещающего все, что не разрешено (deny any – блокировать все остальное).

В маршрутизаторах Cisco существует три типа списков доступа:

- стандартные списки доступа (Standard ACL);
- расширенные списки доступа (Extended ACL);
- именованные списки доступа (Named ACL).

При конфигурировании каждому создаваемому списку доступа присваивается числовой идентификатор, при этом различным типам ACL соответствуют различные диапазоны идентификаторов. Эти диапазоны представлены в таблице 8.2. Именованные списки доступа идентифицируются не числами, а именами.

Таблица 8.2 – Соответствие диапазонов идентификаторов типам ACL

Тип ACL	Диапазон идентификаторов
Standard ACL	1 – 99, 1300 – 1999
Extended ACL	100 – 199, 2000 – 2699

Еще часть диапазонов отводится под такие протоколы, как Appletalk, IPX, но, так как в настоящем пособии рассматривается только стек TCP/IP, рассматривать их не будем.

Стандартные списки доступа описывают правила принятия решения, исходя только из IP-адреса источника пакета. Расширенные списки доступа описывают правила принятия решения, исходя из IP-адреса источника, получателя, протокола сетевого уровня, указанного в заголовке пакета, и номера порта, указанного в заголовке сегмента.

Как указывалось выше, маршрутизатор производит проверку заголовков пакетов по порядку списка, поэтому в строках списков следует задавать условия фильтрации, начиная от частных условий и заканчивая общими. Условия списка доступа проверяются последовательно от первого условия, пока не будет найдено соответствующее совпадение. Если никакое условие не совпало с заголовком, пакет отбрасывается (deny any). При этом отправителю будет послано сообщение протокола ICMP.

Для конфигурирования списка доступа необходимо сначала создать список в режиме глобального конфигурирования, после чего, перейдя в режим конфигурирования интерфейса, применить к нему созданный список.

Рассмотрим сначала использование стандартных списков доступа.

Формат команды создания стандартного списка доступа имеет вид:

Router(config)#access-list <номер> permit / deny <адрес источника>

Списки доступа могут фильтровать как трафик, входящий в маршрутизатор (in), так и трафик, исходящий из маршрутизатора (out). Направление трафика указывается при привязке списка доступа к интерфейсу. Формат команды привязки списка к интерфейсу имеет вид:

Router(config-if)# ip access-group <номер> in / out

Рассмотрим пример создания стандартного списка доступа для сети, представленной на рисунке 8.17 (пример заимствован из [9]).

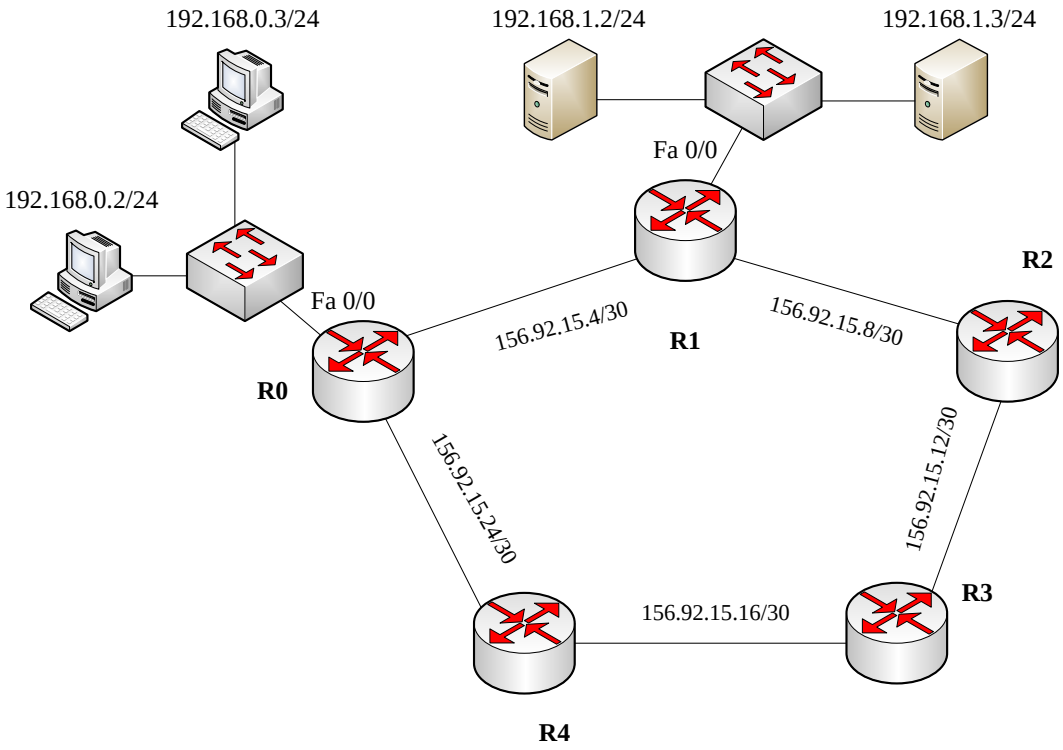

Рисунок 8.17 – Пример сети

Предположим, что к серверу, находящемуся в подсети 192.168.1.0/24 по адресу 192.168.1.2/24, доступ из подсети 192.168.0.0/24 разрешен только компьютеру 192.168.0.2/24. Это правило можно сконфигурировать с использованием стандартного списка доступа на интерфейсе Fa 0/0 маршрутизатора R1.

Для этого в режиме глобального конфигурирования на маршрутизаторе R1 необходимо выполнить следующие команды:

Router1(config)#access-list 10 permit 192.168.0.2

Router1(config)#interface fa 0/0

Router1(config-if)#ip access-group 10 out

Первая команда создает на маршрутизаторе список доступа с номером 10, который разрешает (permit) передачу пакетов с адресом источника 192.168.0.2.

Вторая команда является командой перехода к конфигурированию интерфейса fa 0/0.

Третья команда привязывает список доступа с номером 10 к интерфейсу fa 0/0 и указывает на направление передачи – исходящее (out).

Созданный таким образом список доступа будет состоять из двух строк. Первая строка в явной форме разрешает передавать на интерфейс маршрутизатора fa 0/0 пакеты с адресом источника 192.168.0.2. Вторая строка в неявном виде запрещает (deny any) передавать на этот интерфейс все остальные пакеты.

Проанализируем действия маршрутизатора R1 при поступлении на его внешний интерфейс пакета после создания списка доступа.

Если пакет поступил из подсети 159.92.15.4 и предназначен серверу 192.168.1.1, маршрутизатор, определив по таблице маршрутизации выходной интерфейс, передает этот пакет в буфер интерфейса fa 0/0.

Затем анализируется список, начиная с первой строки. Если источник имеет адрес 192.168.0.2 (совпадение в первой строке списка произошло), пакет инкапсулируется в кадр Ethernet и передается серверу. Если источник имеет любой другой адрес (совпадения в первой строке списка не произошло), происходит обращение ко второй неявной строке списка (deny any), и пакет отбрасывается.

В случае, если необходимо обеспечить доступ к серверу и второго компьютера подсети 192.168.0.0/24 с адресом 192.168.0.3, команды конфигурирования будут выглядеть следующим образом:

Router1(config)#access-list 10 permit 192.168.0.2

Router1(config)#access-list 10 permit 192.168.0.3

Router1(config)#interface fa 0/0

Router1(config-if)#ip access-group 10 out

Очевидно, что список доступа теперь содержит три строки – две явные и одну неявную.

Очевидно, что рассмотренный способ конфигурирования списков доступа удобен в том случае, если доступ к какому-либо ресурсу (серверу) необходимо обеспечить небольшому количеству источников. Если же,

например, в подсети 192.168.0.0/24 значительное количество компьютеров, такое конфигурирование становится неудобным и подверженным ошибкам, так как для каждого из них необходимо отдельно создавать строку списка.

Поэтому при создании списков доступа можно использовать специальной маски, о которой уже шла речь в параграфе 6.2. В этом случае в строке списка может содержаться указание на передачу или фильтрацию пакетов не с адресами конечных узлов, а с адресами сетей (подсетей), в которые они входят.

Напомним, что правило использования масок в этом случае можно сформулировать следующим образом – нулевые значения разрядов маски означают требование обработки соответствующих разрядов адреса, а единичные значения разрядов маски означают игнорирование соответствующих разрядов адреса. Например, если маска имеет вид 0.0.0.0, то проверять условие необходимо для всех разрядов адреса источника прибывшего пакета. Если же маска имеет вид 0.0.0.255, то проверять условие необходимо только для первых трех байтов адреса источника.

Предположим, что доступ к тому же серверу (адрес 192.168.1.1) должны получить все компьютеры подсети 192.168.0.0/24. В этом случае на маршрутизаторе R1 необходимо выполнить команды:

Router1(config)#access-list 10 permit 192.168.0.0 0.0.0.255

Router1(config)#interface fa 0/0

Router1(config-if)#ip access-group 10 out

Необходимо отметить, что, если нужно разрешить какому-либо одному узлу из другой подсети (например, 192.168.2.2/24) доступ к этому же серверу, создаваемый список необходимо дополнить командой

Router1(config)#access-list 10 permit 192.168.2.2 0.0.0.0

или, что то же самое, командой

Router1(config)#access-list 10 permit host 192.168.2.2

Используя приведенную выше аналогию с «белыми» и «черными» списками телефона, можно отметить, что рассмотренные способы

аналогичны созданию в телефоне «белых» списков – указанные в списке доступа адреса являются разрешенными, остальные – запрещенными.

В ряде случаев более удобным является использование аналогии «черного» списка – разрешено передавать данные от всех, кроме тех, кто указан в черном списке.

Предположим, что к тому же серверу необходимо обеспечить доступ всем компьютерам, кроме одного, имеющего адрес 192.168.0.15. Конфигурирование такого списка будет иметь вид:

Router1(config)#access-list 11 deny host 192.168.0.15

Router1(config)#access-list 11 permit any

Router1(config)#interface fa 0/0

Router1(config-if)#ip access-group 11 out

Напомним, что по умолчанию у создаваемых списков доступа неявно присутствует заключительная строка deny any – запретить все. В данном случае мы заменили эту строку на permit any – разрешить все. Соответственно, доступ к серверу будет разрешен всем, кроме компьютера с адресом 192.168.0.15.

Рассмотрим теперь применение расширенных списков доступа.

Общий вид команды создания списка:

access-list <номер> <permit/deny> <протокол> <адрес источника> <адрес получателя> <номер порта или название протокола>

Вместо номера порта можно использовать название прикладного протокола, который этот номер идентифицируется – FTP, HTTP и т.д. В поле «протокол» также можно использовать номер, но значительно удобнее использовать название – IP, TCP, UDP и т.д.

Все остальные правила конфигурирования расширенных списков остаются такими же, как и для стандартных списков.

Рассмотрим применение расширенного списка для конфигурирования маршрутизатора R1 (рисунок 8.17), при этом должны быть выполнены следующие условия:

- компьютеру 192.168.0.2/24 необходимо предоставить доступ к web-серверу с адресом 192.168.1.2 по протоколу HTTP;

- всем компьютерам подсети 192.168.0.0/24 необходимо предоставить доступ к FTP-серверу с адресом 192.168.1.3 по протоколу FTP.

Команды конфигурирования в этом случае будут выглядеть следующим образом:

```
Router1(config)#access-list 110 permit tcp host 192.168.0.2 host 192.168.1.2 eq www
Router1(config)#access-list 110 permit tcp 192.168.0.0 0.0.0.255 host 192.168.1.3 eq ftp
Router1(config)#interface fa 0/0
Router1(config-if)#ip access-group 110 out
```

Очевидно, что указанный способ аналогичен созданию «белого» списка в телефоне, так как третье неявное условие, находящееся в конце списка, как и при использовании стандартных списков доступа, блокирует все, что не разрешено.

Рассмотрим пример, когда удобнее использовать аналогию «черного» списка в телефоне.

На маршрутизаторе R1 должны быть выполнены следующие условия:

- компьютеру 192.168.0.2/24 необходимо запретить доступ к серверу с адресом 192.168.1.2 по протоколу WWW, но разрешить доступы к другим сервисам;

- всем компьютерам подсети 192.168.0.0/24 необходимо предоставить доступ к серверу с адресом 192.168.1.3 по протоколу FTP, но разрешить доступ к другим сервисам.

Команды конфигурирования в этом случае будут иметь вид:

```
Router1(config)#access-list 110 deny tcp host 192.168.0.2 host 192.168.1.2 eq www
Router1(config)#access-list 110 deny tcp 192.168.0.0 0.0.0.255 host 192.168.1.3 eq ftp
Router1(config)#access-list 110 permit ip any any
Router1(config)#interface fa 0/0
Router1(config-if)#ip access-group 110 out
```

Запись permit ip any any означает, что весь остальной трафик от любого источника к любому получателю должен передаваться.

Просмотреть созданные на маршрутизаторе списки доступа можно по команде **show access-list**, а списки, настроенные на конкретных интерфейсах, командами **show ip interfaces** или **show running-config**. На рисунке 8.18 показаны настроенные списки доступа для рассмотренного здесь примера.

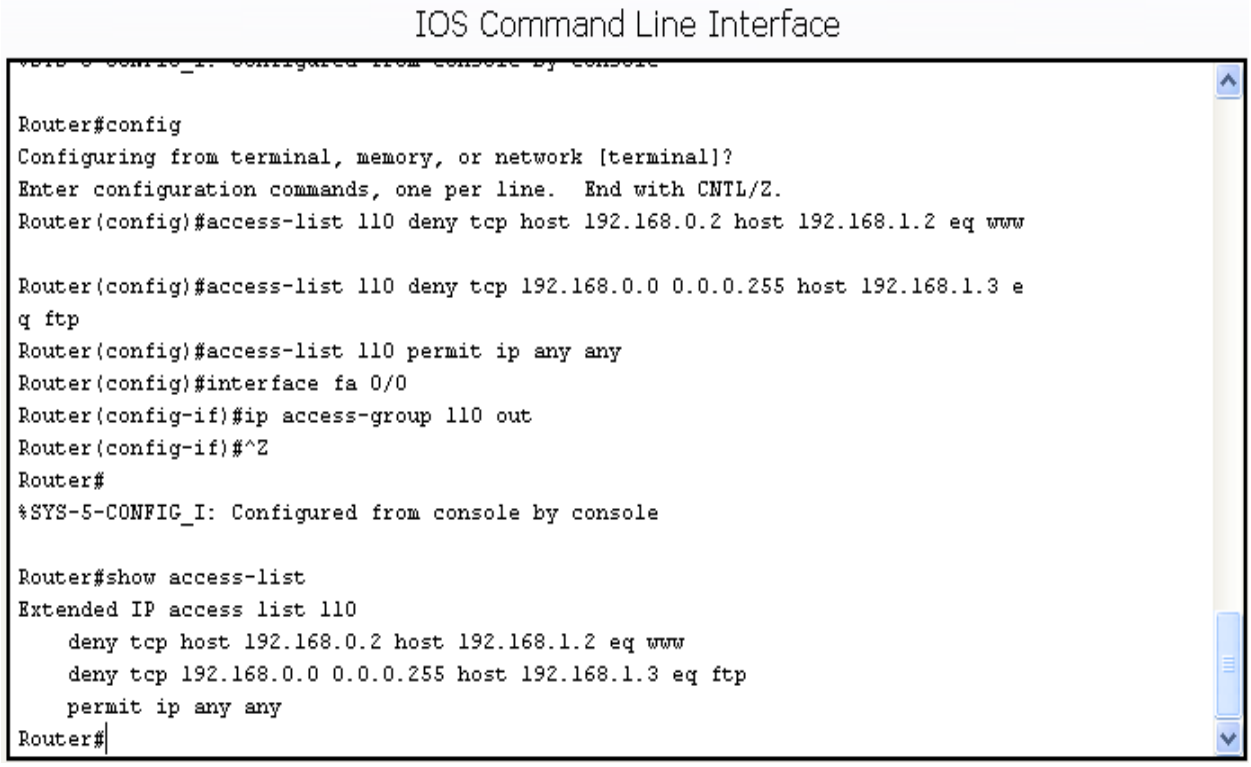

Рисунок 8.18 – Конфигурирование списка доступа и его просмотр

Необходимо помнить, что, если в созданный список доступа добавить новое условие, оно запишется в конец списка и будет проверяться последним. Поэтому, если необходимо вставить в список дополнительное условие в другое место списка, созданный список необходимо удалить и создать заново. Удаление списка доступа производится командой **no access-list <номер>**.

Именованные списки доступа отличаются от рассмотренных только тем, что вместо номеров их можно идентифицировать именами. Кроме того, именованные списки также могут быть стандартными и расширенными.

Таким образом, использование сетевых фильтров на основе списков доступа позволяет гибко настраивать политики сети.

Заключение

В учебном пособии рассмотрены основные технологии коммутации пакетов в сетях связи. Рассмотрены принципы функционирования устройств как канального, так и сетевого уровней, приведены примеры их конфигурирования. Кроме того, рассмотрены основные сетевые протоколы и службы, обеспечивающие прохождение пакетов через IP-сеть.

Знание этих базовых принципов позволит студентам освоить учебные дисциплины, направленные на изучение сетей связи следующего поколения – NGN.

Список использованных источников

1. Бакланов И.Г. NGN: Принципы построения и организации. М.: Эко-Трендз, 2008. – 400 с.
2. Гулевич Д. Сети связи следующего поколения. Курс Национального Открытого Университета «Интуит» - Электронный ресурс [http:// http://www.intuit.ru/studies/courses/1150/157/info]. Дата обращения – 20.10.2017.
3. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 5-е изд. Спб.: Питер, 2016. – 992 с.
4. Манин А.А. Системы коммутации. Принципы и технологии пакетной коммутации. Учебное пособие. Ростов-на-Дону: СКФ МТУСИ, 2016. – 108 с.
5. Соколов Н.А. Телекоммуникационные сети. Монография в 4-х главах. Часть 4 (глава 4). М.: Альварес Паблишинг, 2004. – 192 с.
6. РД.45.120-2000. Руководящий документ отрасли. Нормы технологического проектирования. Городские и сельские телефонные сети.
7. Нерсесянц А.А. Теория телетрафика. Учебное пособие. Ростов-на-Дону: СКФ МТУСИ, 2013. – 92 с.
8. Васин Н.В. Построение сетей на базе коммутаторов и маршрутизаторов. Курс Национального Открытого Университета «Интуит» – Электронный ресурс [http://www.intuit.ru/studies/courses/636/492/info]. Дата обращения – 28.10.17.
9. Соболь Б.В., Манин А.А., Герасименко М.С. Сети и телекоммуникации. Учебное пособие. Ростов-на-Дону: Феникс, 2015. – 191 с.

10. Электронный ресурс [http://www.ieee.org/index.html]. Дата обращения – 3.11.17.
11. Электронный ресурс [http://rfc.com.ru/]. Дата обращения – 3.11.17.
12. Люсин О.Б. Сетевое программирование для ОС Unix и Windows: Учебное пособие. Рига: ИТС, 2006. – 316 с.
13. Электронный ресурс [http://rfc2.ru/792.rfc]. Дата обращения 3.11.17.
14. Дибров М.В. Маршрутизаторы: Учебное пособие. Иркутск: Иркутский государственный университет, 2008. – 389 с.
15. Мамаев М. Телекоммуникационные технологии (Сети TCP/IP): Учебное пособие – Электронный ресурс [abc.vvcu.ru]. Дата обращения – 5.11.17.