МГУПИ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение
высшего профессионального образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПРИБОРОСТРОЕНИЯ И ИНФОРМАТИКИ»

Кафедра «Персональные компьютеры и сети»

Брейман А.Д.

Сети ЭВМ и телекоммуникации. Глобальные сети

Учебное пособие

Москва, 2006

УДК 681.326(075)

Сети ЭВМ и телекоммуникации. Глобальные сети. Учебное пособие. / Брейман А.Д. — М.:МГУПИ, 2006. — 116с.

Настоящее пособие предназначено для подготовки студентов, изучающих принципы построения компьютерных сетей и технологии, использующиеся в глобальных сетях.

Для специальности 2301 (2201) «Вычислительные машины, комплексы, системы и сети» настоящее пособие может быть использовано при изучении курсов «Сети и средства телекоммуникаций» и «Администрирование сетевых систем».

Рекомендовано Ученым Советом МГУПИ в качестве учебного пособия для специальности 2301 (2201) «Вычислительные машины, комплексы, системы и сети».

Автор: Брейман А.Д.

Рецензенты: проф., к.т.н. Рощин А.В.
доц., к.т.н. Журавлев В.А.

Работа рассмотрена и одобрена на заседании кафедры ИТ-4 “Персональные компьютеры и сети” 13 апреля 2006 г.

Зав. кафедрой ИТ-4,
профессор, д.т.н. Михайлов Б.М.

Содержание

1 Организация составных сетей

1.1 Корпоративные сети

Корпоративной сетью (англ. corporate network) называется сеть смешанной топологии, объединяющая локальные сети (и, возможно, отдельные компьютеры) подразделений и филиалов корпорации (предприятия, организации, учреждения и т.д.). Корпоративная сеть является собственностью корпорации.

1.2 Функции сетевого уровня

Сетевой (межсетевой) уровень отвечает за передачу пакетов между узлами в составных сетях, включая выбор маршрута передачи и согласование протоколов канального уровня.

Составные сети могут строиться на стандартах канального уровня, с использованием повторителей, мостов и коммутаторов. Такой подход часто бывает оправдан, однако имеет некоторые существенные недостатки и ограничения:

- в сети должны отсутствовать замкнутые маршруты (или, что то же самое, разные пути между двумя узлами); использование протокола STP позволяет коммутаторам работать в таких сетях, но только отключив часть линий связи;
- такие сети подвержены широковещательным штормам;
- сложно управлять трафиком на основе передаваемых данных;
- плоская система адресации (только MAC-адреса, привязанные к сетевым адаптерам);
- ограниченные возможности трансляции разных протоколов канального уровня.

1.3 Составные сети

Сеть, которая рассматривается как совокупность нескольких сетей, называется **составной сетью** или **интерсетью** (англ. internetwork, internet), а ее части — **подсетями** (англ. subnet) или просто сетями.

Составная сеть образуется путем соединения подсетей **маршрутизаторами** (англ. router). Разные подсети могут быть локальными и глобальными, строиться на разных технологиях — Ethernet, Token Ring, X25, Frame Relay, ATM и т.д. Для того чтобы не зависеть от разных способов адресации, используемых в разных технологиях (локальных адресов), вводятся сетевые адреса, уникально идентифицирующие все узлы составной сети. Как правило, сетевой адрес состоит из двух частей: номера подсети и номера узла в подсети. Существуют способы адресации, использующие в качестве номера узла локальные (MAC) адреса (например, IPX/SPX). Такой вариант имеет ограниченную сферу применения, поскольку зависит от локальных технологий. Другие технологии назначают узлам номера независимо от их локальных адресов (например, TCP/IP).

1.4 Принципы маршрутизации

Маршрутизатор, как и, например, мост, имеет несколько портов и должен для каждого поступающего пакета решить — отфильтровать его или передать на какой-то другой порт.

Как и мосты, маршрутизаторы решают эту задачу с помощью специальной таблицы — таблицы маршрутизации. По этой таблице маршрутизатор определяет, на какой порт нужно передавать пакет, чтобы он достиг нужной подсети (не обязательно сразу). Если сеть содержит петли, в таблицах маршрутизации может быть несколько записей на одну подсеть, описывающих разные возможные маршруты.

Каждый порт маршрутизатора рассматривается, как отдельный узел сети. Другие узлы должны знать его адрес и направлять пакеты для передачи в другие подсети на этот адрес, а не просто выдавать их в канал (как при прозрачных мостах).

Рассмотрим принципы маршрутизации на примере сети, изображенной на рисунке 1.1. Здесь S1..S10 — подсети, M1..M13 — маршрутизаторы. Порты маршрутизаторов будем обозначать сочетанием имени маршрутизатора и сети, например, M5/S2 — порт маршрутизатора M5, подключенный к сети S2.

Таблица маршрутизации для маршрутизатора M7 приведена в таблице 1.1

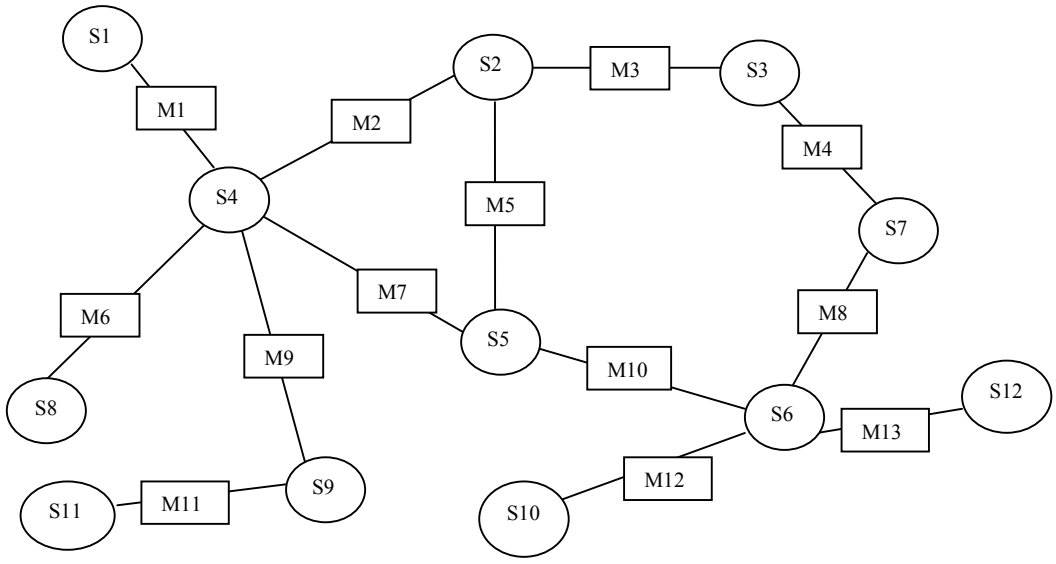

Рисунок 1.1 — Составная сеть с маршрутизаторами

Таблица 1.1 — Таблица маршрутизации для маршрутизатора M7

Номер подсети назначения	**Адрес порта следующего маршрутизатора**	**Адрес выходного порта**	**Расстояние до сети назначения**
S1	M1/S4	M7/S4	1
S2	M5/S5	M7/S5	2
S3	M5/S5	M7/S5	3
S4	——	M7/S4	0
S5	——	M7/S5	0
S6	M6/S4	M7/S4	1
S9	M9/S4	M7/S4	1
S11	M9/S4	M7/S4	2
Default	M10/S5	M7/S5	1

Алгоритм маршрутизации — правило назначения выходной линии связи (порта) на основе данных, содержащихся в заголовке пакета, данных, описывающих состояние маршрутизатора и сети в целом.

Эффективность алгоритмов характеризуется:

- временем доставки пакетов,
- нагрузкой на сеть,
- затратами ресурсов маршрутизаторов (времени и памяти).

Для повышения эффективности желательно, чтобы каждый маршрутизатор имел информацию, как о топологии сети, так и состоянии узлов и связей между ними.

В зависимости от того, какой компонент сети — узел или маршрутизатор — принимает решение о пути пакета, различают маршрутизацию от источника и одношаговую. При маршрутизации от источника (англ. source routing) узел записывает путь (последовательность адресов промежуточных маршрутизаторов) в каждый отправляемый пакет, и маршрутизаторам остается только выполнять указания узла, если это возможно. При одношаговой маршрутизации решение о том, куда дальше передавать пакет принимается на каждом шаге очередным маршрутизатором.

Алгоритмы одношаговой маршрутизации бывают простые, фиксированные и адаптивные. Решение, принимаемое при простой маршрутизации не зависит ни от топологии, ни от состояния сети. Основные варианты простой маршрутизации:

- **случайная** — передача пакета на любой порт, кроме исходного;
- **по кратчайшей очереди** — передача пакета на порт с самой короткой очередью;
- **лавинная** — передача пакета на все порты, кроме исходного;
- **по предыдущему решению** — передача пакета на тот порт, куда отправлялись предыдущие пакеты от этого источника (если пакет — первый, то случайно или по кратчайшей очереди).

Фиксированная маршрутизация основана на статически заданных неизменяемых таблицах маршрутизации (например, они могут создаваться при загрузке операционной системы) и может быть однонаправленной (может быть только один маршрут для сети назначения), либо многонаправленной (допускается наличие нескольких маршрутов для одной сети).

Адаптивная маршрутизация опирается на знание топологии составной сети и может учитывать изменения состояния сети. Различают следующие виды адаптивной маршрутизации:

- **локальная** — только на основе информации о состоянии своих выходных каналов и очередях пакетов;
- **распределенная** — на основе информации, получаемой от других узлов (регулярный обмен узлов таблицами маршрутизации);
    - на основе **векторов расстояний** (или дистанционно-векторная, англ. Distance Vector Algorithms) — рассылается вектор из метрик соседних сетей;
    - на основе **состояния связей** (англ. Link State Algorithms) — каждый узел строит полный граф сети (передаются ребра графа маршрутизатор-маршрутизатор и маршрутизатор-сеть);
- **централизованная** — с выделенным центром маршрутизации, собирающем информацию о состоянии узлов и каналов и рассылающем ее всем узлам;
- **гибридная** — централизованная+локальная (если путь в таблице один, то пакет отправляется по нему, иначе — на порт с самой короткой очередью).

1.5 Вопросы для самопроверки

- Должны ли подсети, соединяемые в составную сеть, иметь одинаковые канальные уровни (например, Ethernet)?
- Можно ли построить составную сеть диаметром 5 км только на коммутаторах, если все ее подсети построены на Ethernet?

- Какая строка в таблице маршрутизации маршрутизатора M4 описывает путь к сети S12 (по рисунку 1.1)?
- По каким характеристикам маршрутизация от источника превосходит одношаговую маршрутизацию?
- При каких условиях предпочтительно применять фиксированную маршрутизацию?
- При каких условиях предпочтительно применять лавинную маршрутизацию?
- Какие преимущества и недостатки имеются у централизованной маршрутизации?

2 Протоколы TCP/IP

Все протоколы стека TCP/IP, сетевые службы, принципы их реализации и другие сопутствующие вопросы описываются в документах RFC (Request For Comment — предложение для обсуждения). Исходно это были действительно документы, предназначенные для обсуждения, но впоследствии сформировавшиеся стандарты также оформлялись в виде RFC.

Рассмотрение стека протоколов TCP/IP будем сопровождать указаниями номеров RFC, описывающих тот или иной протокол. Документы RFC можно получить, например, с Web-узла РосНИИРОС: http://www.ripn.net/nic/

2.1 Адресация в Internet

Узлы сетей, входящих в Internet, используют, по крайней мере, три системы адресации: **локальную** (адреса канального уровня, т.е., например, MAC-адреса в локальной Ethernet-сети), **сетевую** (IP-адреса) и **символьную** (доменные имена, DNS-имена).

Локальные адреса используются для доставки пакетов в пределах подсети, сетевые адреса — для маршрутизации пакетов между подсетями, а символьные имена — для более простого и запоминающегося именования узлов.

Сетевой уровень стека TCP/IP передает пакеты между сетями, опираясь на IP-адреса (RFC 990 и RFC 997). IP-адрес (четвертой версии, являющейся основной в настоящее время) состоит из 32 бит (4 байт). Как правило, IP-адрес записывают как четыре десятичных числа (значения отдельных байтов), разделенные точками, например: 123.45.67.89. Адрес состоит из двух частей: номера подсети и номера узла, причем номер узла не зависит от его MAC-адреса (или другого локального адреса). Распределение номеров подсетей для Internet осуществляется централизованно (долгое время только InterNIC, потом ICANN, в скором времени — группа независимых организаций), а для

внутренних подсетей, не связанных напрямую с Internet, может назначаться администратором сети.

Все IP-адреса разделены на 5 классов (от A до E), задающих разные соотношения между количеством подсетей и количеством узлов в них. На рисунке 2.1 показана структура адресов разных классов (буквой N обозначены биты адреса, составляющие номер сети, буквой H — биты адреса, составляющие номер узла в сети, буквой M — биты адреса, составляющие номер группы рассылки).

Класс	1 байт								2 байт								3 байт								4 байт							
A	0	N	N	N	N	N	N	N	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
B	1	0	N	N	N	N	N	N	N	N	N	N	N	N	N	N	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
C	1	1	0	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	H	H	H	H	H	H	H	H
D	1	1	1	0	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M
E	1	1	1	1	0							P	E	З	E	P	B															

Рисунок 2.1 — Классы IP-адресов

Некоторые IP-адреса интерпретируются специальным образом:

- адрес, все биты которого равны нулю, обозначает адрес того узла, который выдал этот пакет;
- адрес, в поле номера сети которого стоят только нули, считается относящимся к той же сети, что и узел, выдавший этот пакет;
- адрес, все биты которого равны единице, означает, что данный пакет должны получить все узлы подсети, к которой относится узел, выдавший этот пакет (ограниченное широковещательное сообщение, англ. limited broadcast);
- адрес, в котором все биты поля номера узла равны единице, а поле номера сети задает определенную сеть (не все нули и не все единицы), то такой пакет должен рассылаться всем узлам указанной подсети (широковещательное сообщение, англ. broadcast).

Таким образом, ни номер подсети, ни номер узла не может состоять из одних нулей или одних единиц. Это ограничивает количество узлов в подсети

соотношением: $N_{узлов} = 2^n–2$, где n — количество бит в поле номера узла, а количество подсетей — соотношением: $N_{подсетей} = 2^m–2$, где m — количество бит в поле номера подсети.

Например, каждая из 16382 ($2^{14}–2$) подсетей класса B (14 бит под номер подсети, 16 бит под номер узла) максимально может включать 65534 ($2^{16}–2$) узла с номерами от x.x.0.1 до x.x.255.254.

Кроме того, выделяется группа адресов, первый байт которых равен 127. Эти адреса используются для передачи данных между процессами на одном компьютере или для тестирования. Данные, отправленные по такому адресу, рассматриваются, как только что принятые из сети, в результате чего образуется как бы “петля” (англ. loopback). Обычно используется адрес 127.0.0.1, но для этих целей можно использовать любой адрес вида 127.x.x.x.

Три блока IP-адресов зарезервированы для внутреннего использования в корпоративных сетях, отделенных от Интернета маршрутизаторами с функцией трансляции сетевых адресов NAT (англ. Network Address Translation), описанной в RFC 1631. Это одна подсеть класса A (адреса с 10.0.0.0 по 10.255.255.255), 16 подсетей класса B (адреса с 172.16.0.0 по 172.31.255.255) и 256 подсетей класса C (адреса с 192.168.0.0 по 192.168.255.255). Такие адреса называются «частными» (англ. private). Маршрутизаторы Интернета не должны маршрутизировать пакеты с такими адресами получателя.

2.1.1 Маска подсети

При классовой адресации определить, сколько бит адреса отведено под номер подсети, а сколько — под номер узла в подсети, можно по самому адресу (его старшим битам). Чтобы выделить эти номера из IP-адреса, можно, например, использовать 32-битовую маску, в которой старшие биты, соответствующие номеру подсети, выставлены в 1, а остальные — в 0. Тогда номер подсети выделяется операцией побитового И над адресом и маской, а номер узла — той же операцией над адресом и инвертированной маской. Такая

битовая маска называется в TCP/IP **маской подсети** (англ. subnet mask); их принято записывать так же, как IP-адреса: четыре десятичных значения байтов, разделенных точками. Каждому классу соответствует фиксированная маска: классу A — 255.0.0.0, классу B — 255.255.0.0, классу C — 255.255.255.0. Сначала маски использовались только внутри программных модулей, реализующих сетевые протоколы, но со временем стали при IP-адресе указывать его маску подсети практически всегда.

Если организация получила в пользование некоторую подсеть, то она может распоряжаться всеми возможными значениями поля номера узла по своему желанию. Можно, например, продолжить идею разделения на подсети внутрь этого поля: разбить его на две части — номер подподсети и номер узла в подподсети. Такое разбиение может оказаться удобным, если у организации несколько локальных сетей и было бы желательно, чтобы можно было по IP-адресу видеть, к какой локальной сети (т.е. подподсети) относится данный компьютер. При обычной классовой адресации нет возможности указать, где проходит граница между номером подподсети и номером узла в ней. Использование масок подсети позволяет реализовать такое разбиение очень просто: если при адресе всегда передается маска подсети, то ничто не мешает считать номером узла только биты адреса, соответствующие нулевым битам маски, а остальные биты, которые в соответствии с классом адреса тоже относятся к номеру узла, считать номером подподсети. Например, адресу 194.153.99.210 с маской 255.255.255.224 соответствует подсеть класса C 195.153.99, подподсеть 6, узел 18. Поскольку маска подсети всегда имеет одинаковую структуру: старшие биты — 1, младшие — 0, вместо четырех байт маски достаточно хранить и передавать всего один байт (реально хватило бы четырех бит) — количество старших единичных бит: например, маске 255.255.255.224 соответствует суффикс подсети /19.

При нумерации подподсетей и узлов них необходимо соблюдать те же правила, что и для обычных подсетей: все нули и все единицы в номере подподсети и в номере узла используются только для служебных целей. Таким

образом, при разбиении на подподсети часть адресного пространства подсети расходуется впустую.

Использование классовой адресации привело к очень неравномерному распределению адресного пространства: например, половина все возможных адресов — адреса класса A — зарезервирована за 127 подсетями, которые заведомо не используют все эти адреса. В то же время сети класса C стали дефицитом уже в середине 1990-х годов. Решением этой проблемы стала бесклассовая адресация. При бесклассовой адресации старшие биты адреса перестают нести особый смысл, а границу между номером подсети и номером узла определяет маска (или суффикс) подсети. Старые классовые адреса при этом сохраняются (с соответствующими масками).

2.2 Протокол IP

Протокол IP (англ. Internet Protocol, протокол межсетевого взаимодействия) описан в RFC 791. Основная функция протокола IP — передача пакетов между узлами, принадлежащими к разным подсетям, через промежуточные подсети. Каждый пакет (дейтаграмма, англ. datagram) обрабатывается независимо от других. Доставка дейтаграмм не гарантируется. Возможны потери дейтаграмм, доставка с ошибками, дублирование и нарушение порядка следования. Вторая функция протокола IP — выполнение фрагментации пакетов при передаче их между сетями с разным максимально допустимым размером поля данных кадра (англ. Maximum Transfer Unit, MTU).

Существенное свойство протокола IP состоит в нетрадиционном порядке передачи битов: байт передается, начиная со старшего бита. Кроме того, нумерация битов в байте также начинается со старшего: самый старший бит имеет номер 0, самый младший — номер 7.

Дейтаграмма (IP-пакет) состоит из заголовка и поля данных. Формат заголовка приведен на рисунке 2.2.

1 байт								2 байт								3 байт								4 байт							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Версия				IHL				Тип Сервиса								Общая длина															
Идентификация																Флаги			Смещение фрагмента											
Время жизни								Протокол								Контрольная сумма заголовка															
IP-адрес отправителя																															
IP-адрес получателя																															
Опции (переменная длина)																								Дополнение до 4 байт							

Рисунок 2.2 — Формат заголовка IP-пакета

Поле Номер версии (англ. Version) [4 бита] — указывает используемый формат заголовка. В настоящее время основная используемая версия имеет номер 4.

Поле Длина заголовка (англ. Internet Header Length, IHL) [4 бита] — длина заголовка в четырехбайтовых словах, минимальное допустимое значение — 5 (соответствует длине заголовка в 20 байт). Максимальная длина заголовка — 60 байт.

Поле Тип сервиса (англ. Type of Service) [8 бит] — указывает на желаемые параметры качества обслуживания. Формат байта Типа сервиса приведен на рисунке 2.3.

Биты	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Подполя	Приоритет			D	T	R	C	0

Рисунок 2.3 — Структура поля «Приоритет» заголовка IP-пакета

Поле Приоритета (англ. Precedence) для обычных пакетов равно 0, остальные значения (от 1 до 7) используются для служебных целей, чем больше значение, тем выше приоритет.

Поля D (англ. Delay, задержка), T (англ. Throughput, пропускная способность) и R (англ. Reliability, надежность) используются для указания наиболее важного для передающего узла параметра качества. Выбор происходит между малой задержкой, большой пропускной способностью и высокой надежностью. Соответствующий бит (биты) устанавливается в 1,

остальные — в 0. Как правило, улучшение одного из параметров связано с ухудшением остальных.

Поле Общая длина (англ. Total Length) [16 бит] — длина дейтаграммы (заголовка и данных) в байтах. Хотя размер поля позволяет создавать дейтаграммы длиной до 65535 байт, стандарт требует, чтобы любой узел мог принимать, как минимум, 576-байтовые дейтаграммы (как целиком, так и частями), а отправлять дейтаграммы большей длины только, будучи уверенным, что получатель может их принять. Значение 576 выбрано, чтобы можно было передавать в одном IP-пакете 512 байт данных («блок данных разумного размера», как написано в стандарте; отметим, что 512 байт — это типичный размер сектора жесткого или гибкого диска), оставляя 64 байта под заголовки протоколов: IP-заголовок занимает от 20 до 60 байт.

Поле Идентификация (англ. Identification) [16 бит] — значение, одинаковое для всех дейтаграмм, содержащих фрагменты одного пакета («большого пакета»).

Поле Флаги (англ. Flags) [3 бита] — флаги, управляющие фрагментированием:

0 бит — зарезервирован, должен быть равен 0;

1 бит (DF, англ. Don’t Fragment) — «0» = можно фрагментировать,
«1» = нельзя фрагментировать;

2 бит (MF, англ. More Fragments) — «0» = последний фрагмент,
«1» = еще будут фрагменты.

Поле Смещение фрагмента (англ. Fragment Offset) [13 бит] — указывает на место в «большом пакете», с которого начинаются данные текущей дейтаграммы. Измеряется в 64-битовых (8-байтовых) словах. Например, Смещение фрагмента, равное двум, означает, что данные текущей дейтаграммы должны находиться в «большом пакете», начиная с 16-го байта. Первый фрагмент имеет нулевое смещение.

Поле Время жизни (англ. Time to Live, TTL) [8 бит] — максимальное время, которое дейтаграмма может находиться в сети. Каждый маршрутизатор

должен уменьшать это значение на единицу, и отбрасывать дейтаграммы со значением TTL = 0, передавая при этом отправителю соответствующее ICMP-сообщение. Наличие этого поля обеспечивает уничтожение «зациклившихся» или «заблудившихся» дейтаграмм. Поле TTL также позволяет ограничить дальность распространения дейтаграммы (это удобно, например, при одновременной передаче множеству абонентов) и является основой для работы утилиты traceroute.

Протокол (англ. Protocol) [8 бит] — указывает, данные какого протокола верхнего уровня передаются в дейтаграмме. Возможные значения этого поля стандартизованы (RFC «Assigned Numbers»), приведем некоторые из них: 1 — ICMP, 4 — IP, 6 — TCP, 17 — UDP, 89 — OSPF.

Поле Контрольная сумма заголовка (англ. Header Checksum) [16 бит] — контрольная сумма всех полей заголовка, вычисляемая как дополнение суммы всех 16-битовых слов заголовка (с нулевыми битами в поле контрольной суммы). Поскольку некоторые поля заголовка (например, время жизни) изменяются при передаче дейтаграммы через сеть, контрольная сумма пересчитывается каждым маршрутизатором. Если получена дейтаграмма с неверной контрольной суммой, такая дейтаграмма отбрасывается.

Поле Адрес отправителя (англ. Source IP Address) [32 бита] — IP-адрес отправителя дейтаграммы.

Поле Адрес получателя (англ. Destination IP Address) [32 бита] — IP-адрес получателя дейтаграммы.

Поле Опции (англ. Options) [переменная длина] — необязательное поле, может содержать данные о безопасности, маршрут дейтаграммы (при маршрутизации от источника) и т.д. В одной дейтаграмме может быть несколько опций, каждая из которых состоит из кода опции (1 байт), длины опции (1 байт) и байтов данных опции. Если для опции не нужны дополнительные данные, она состоит из одного байта — кода опции.

Опции в настоящее время практически не используются.

Поле Дополнение (англ. Padding) — нулевые байты в таком количестве, чтобы размер заголовка был кратен 4 байтам.

2.2.1 Фрагментация IP-пакетов

На пути пакета от отправителя к получателю могут встречаться локальные и глобальные сети разных типов с разными допустимыми размерами полей данных кадров канального уровня (MTU). Так, например, сети Ethernet могут передавать кадры размером до 1500 байт, сети FDDI — до 4500 байт, в других сетях действуют свои ограничения. Протокол IP умеет передавать дейтаграммы, длина которых больше MTU промежуточной сети, за счет фрагментирования — разбиения «большого пакета» на некоторое количество частей (фрагментов), размер каждой из которых удовлетворяет промежуточную сеть. После того, как все фрагменты будут переданы через промежуточную сеть, они будут собраны на узле-получателе модулем протокола IP обратно в «большой пакет». Отметим, что сборку пакета из фрагментов осуществляет только получатель, а не какой-либо из промежуточных маршрутизаторов. Маршрутизаторы могут только фрагментировать пакеты, но не собирать их. Это связано с тем, что разные фрагменты одного пакета не обязательно будут проходить через одни и те же маршрутизаторы.

Для того чтобы не перепутать фрагменты разных пакетов, используется поле «Идентификация», значение которого должно быть одинаковым для всех фрагментов одного пакета и не повторяться для разных пакетов, пока у обоих пакетов не истекло время жизни.

При делении данных пакета, размер всех фрагментов, кроме последнего, должен быть кратен 8 байтам. Это позволяет отвести меньше места в заголовке под поле «Смещение фрагмента».

Второй бит поля «Флаги» (MF), если равен единице, указывает на то, что данный фрагмент — не последний в пакете.

Если пакет отправляется без фрагментации, флаг MF устанавливается в 0, а поле Смещение фрагмента — заполняется нулевыми битами.

Если первый бит поля «Флаги» (DF) равен единице, то фрагментация пакета запрещена. Если этот пакет должен быть передан через сеть с недостаточным MTU, то маршрутизатор вынужден будет его отбросить (и сообщить об этом отправителю посредством протокола ICMP). Этот флаг используется в случаях, когда отправителю известно, что у получателя нет достаточно ресурсов по восстановлению пакетов из фрагментов.

2.3 Протокол ARP

Протокол ARP (Address Resolution Protocol, Протокол Разрешения Адресов) описан в RFC 826.

При передаче пакетов внутри локальных сетей протоколы канального уровня пользуются локальными адресами узлов, отправитель же может знать только IP-адрес получателя. Для того чтобы определить, какой локальный адрес (например, MAC-адрес в сети Ethernet) соответствует данному IP-адресу, применяется протокол ARP. Этот протокол разрабатывался специально для Ethernet-сетей, но может работать в любых сетях, поддерживающих широковещательную передачу.

Все узлы, поддерживающие протокол ARP, ведут ARP-таблицу, состоящую из записей <IP-адрес;MAC-адрес>.

Когда узлу нужно определить локальный адрес другого узла, его ARP-модуль сначала ищет его в ARP-таблице, и, если нужный адрес не найден, то передает широковещательное сообщение: «Знает ли кто-нибудь локальный адрес для IP 123.45.67.89? Я 123.45.67.90, мой MAC-адрес 10:20:30:40:50:60.». Узел, которого разыскивают, отвечает (не широковещательно, а прямой передачей): «Да, 123.45.67.89 — это я. Мой MAC-адрес 10:20:30:40:50:61». При этом он сохраняет пару <IP-адрес;MAC-адрес> искавшего его узла в своей

ARP-таблице. Наконец, первый узел, получив ответ, заносит его в свою ARP-таблицу.

Как правило, записи в ARP-таблице имеют ограниченное время жизни (стандарт описывает возможные схемы ограничения времени жизни и тайм-аутов, но не требует их применения).

Формат сообщения ARP позволяет использовать этот протокол в сетях с разным размером адресов (до 256 бит).

Сообщения ARP не содержат IP-заголовка и непосредственно размещаются в поле данных кадра канального уровня.

2.4 Протокол ICMP

Протокол ICMP (Internet Control Message Protocol, Протокол Управляющих Сообщений Интернет) описан в RFC 792.

Он используется для сообщений об ошибках или нештатных ситуациях, передаваемых узлу-отправителю дейтаграммы узлом-получателем или промежуточным маршрутизатором.

Хотя сообщения ICMP вкладываются в поле данных IP-дейтаграммы, то есть ICMP как бы является протоколом более высокого уровня, чем IP, модуль обработки ICMP-сообщений входит в модуль, реализующий протокол IP.

Сообщения ICMP всегда начинаются с заголовка, состоящего из трех полей (см. рисунок 2.4), за которым следуют данные об ошибке.

<table>
<tr><th colspan="8">1 байт</th><th colspan="8">2 байт</th><th colspan="8">3 байт</th><th colspan="8">4 байт</th></tr>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="8">Тип</td><td colspan="8">Код</td><td colspan="16">Контрольная сумма</td></tr>
<tr><td colspan="24">Данные (переменная длина; могут отсутствовать)</td></tr>
</table>

Рисунок 2.4 — Формат сообщения ICMP

Поле Тип (англ. Type) [8 бит] — тип сообщения (см. таблицу 13.1).

Поле Код (англ. Code) [8 бит] — причина проблем с доставкой дейтаграммы (см. таблицы 2.1, 2.2, 2.3 и 2.4). Для остальных типов в поле Код записывается нулевое значение.

Поле Контрольная сумма (англ. Checksum) [16 бит] — контрольная сумма ICMP-сообщения (начиная с поля Тип), вычисляемая, как в протоколе IP.

Таблица 2.1 — Типы сообщений ICMP

Тип	**Описание**
0	ответ на эхо (Echo reply)
3	получатель недостижим (Destination unreachable)
4	подавление источника (Source quench)
5	изменение маршрута (Redirect)
8	запрос эха (Echo)
11	время жизни дейтаграммы истекло (Time exceeded)
12	ошибка в параметре (Parameter problem)
13	запрос временной метки (Timestamp)
14	временная метка (Timestamp reply)
15	запрос информации (Information request)
16	ответ на запрос информации (Information reply)
17	запрос маски адреса (Mask request)
18	маска адреса (Mask reply)

Таблица 2.2 — Коды сообщений ICMP для типа 3 (получатель недостижим)

Код	**Описание**
1	сеть недостижима
2	узел недостижим
3	протокол недостижим
4	требуется фрагментация, но установлен флаг DF
5	сбой в маршрутизации от источника
6	неизвестна сеть назначения
7	неизвестно устройство назначения
8	отправитель изолирован
9	закрыт доступ к сети назначения
10	закрыт доступ к устройству назначения
11	сеть недостижима из-за требований к классу обслуживания
12	устройство недостижимо из-за требований к классу обслуживания

Таблица 2.3 — Коды сообщений ICMP для типа 5 (изменение маршрута):

Код	**Описание**
0	переадресовать дейтаграммы для сети
1	переадресовать дейтаграммы для узла
2	переадресовать дейтаграммы для типа обслуживания и сети
3	переадресовать дейтаграммы для типа обслуживания и узла

Таблица 2.4 — Коды сообщений ICMP для типа 11 (время жизни дейтаграммы истекло):

Код	Описание
0	время жизни истекло при передаче
1	время жизни истекло при ожидании фрагмента для сборки

2.5 Базовые утилиты для тестирования сетей TCP/IP

Утилита Ping позволяет проверить существование указанного узла и измерить время передачи до него одного пакета (можно задавать разные размеры пакета для исследования промежуточных сетей). Эта утилита выполняет передачу ICMP-сообщения типа 8 (Echo request), на которое получатель должен ответить ICMP-сообщением типа 0 (Echo reply).

Утилита Traceroute показывает последовательность узлов, через которые проходит пакет на пути к получателю. Реализовано это следующим образом: последовательно отправляются пакеты с возрастающим значением в поле TTL: 1,2,3 и т.д. Тот маршрутизатор, который уменьшит TTL до нуля, обязан будет отправить ICMP-сообщение типа 11 (Time exceeded). В результате будут получены такие ICMP-сообщения по очереди от всех маршрутизаторов на пути пакета к получателю. В различных ОС эта утилита реализована по-разному: в ОС семейства Windows отправляются ICMP-сообщения (подобно утилите ping), а в Linux/FreeBSD — UDP-сообщения.

2.6 Протоколы транспортного уровня

2.6.1 Функции транспортного уровня

Отправителем и получателем данных, передаваемых через сеть, с точки зрения транспортного уровня, является приложение (процесс). Как любая программа, процессы создаются и уничтожаются, на каждом узле может выполняться несколько процессов, а каждый процесс может иметь несколько

точек подключения к сети. Такие логические точки (программно организуемые, как правило, в виде очередей сообщений) называются **портами** (англ. port). Номер порта однозначно идентифицирует процесс. Когда узел получает дейтаграмму транспортного уровня, он направляет ее прикладному процессу, используя номер порта, заданный при установлении связи.

Порты нумеруются положительными целыми 16-битовыми числами. Разные протоколы транспортного уровня нумерую свои порты независимо, то есть, например, порт 20 протокола TCP и порт 20 протокола UDP совершенно не связаны друг с другом.

Некоторые номера портов заданы стандартами. Эти номера выделяются организацией IANA (англ. Internet Assigned Numbers Authority). В настоящее время под стандартные порты отведен диапазон от 0 до 1023 (ранее — до 255).Остальные порты могут свободно использоваться прикладными процессами. Порты в диапазоне от 1024 до 5000 называются временными (англ. ephemeral). Назначение этих портов не стандартизовано, но IANA поддерживает информацию об их использовании.

Пара «порт — IP-адрес» называется (в терминологии TCP/IP) **гнездом** или **сокетом** (англ. socket) и однозначно указывает программный процесс, выполняющийся на одном из узлов в сети.

2.6.2 Протокол UDP

Протокол UDP (англ. User Datagram Protocol, Протокол пользовательских дейтаграмм) описан в RFC 768. Он предоставляет прикладным процессам простейшие услуги транспортного уровня. Две основные функции UDP — распределение дейтаграмм между процессами (на основании номеров портов) и контроль передачи пользовательских данных (не только заголовка, как в протоколе IP). Как и IP, UDP не гарантирует доставку и не поддерживает установку соединений.

Сообщение протокола UDP называется **пользовательской дейтаграммой** (англ. User datagram) и состоит из заголовка и пользовательских данных. Структура заголовка приведена на рисунке 2.5. Сразу за заголовком идут пользовательские данные.

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Порт отправителя																Порт получателя															
Длина сообщения																Контрольная сумма															

Рисунок 2.5 — Формат заголовка дейтаграммы UDP

Если значение поля «Порт отправителя» не важно для получателя, в него можно записать нулевое значение.

В поле «Длина сообщения» записывается размер пользовательских данных в байтах.

Нулевое значение в поле «Контрольная сумма» означает, что контрольная сумма не вычислялась. Для расчета контрольной суммы к началу дейтаграммы приписывается псевдозаголовок, состоящий из пяти полей (см. рисунок 2.6).

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
IP-адрес отправителя																															
IP-адрес получателя																															
0	0	0	0	0	0	0	0	Протокол								Длина дейтаграммы															

Рисунок 2.6 — Формат псевдозаголовка UDP для расчета контрольной суммы

Если длина дейтаграммы нечетна, то к ее концу добавляют один байт с нулевым значением. Перед началом расчета в поле «Контрольная сумма» записывается нулевое значение. Затем вычисляется контрольная сумма (по тому же алгоритму, как и в протоколе IP), результат записывается в поле «Контрольная сумма», а псевдозаголовок отбрасывается.

2.6.3 Протокол TCP

Протокол TCP (англ. Transmission Control Protocol, Протокол управления передачей) описан в RFC 793. Он обеспечивает надежную передачу потока данных, используя сервис передачи дейтаграмм протокола IP. Пакеты, передаваемые протоколом TCP, называются сегментами. Каждый TCP-сегмент размещается в одном IP-пакете (а в каждом IP-пакете может находиться только один TCP-сегмент). Надежность передачи обеспечивается при помощи нумерации байтов потока и подтверждений приема. Все байты исходного потока данных нумеруются (этот номер называется номером в последовательности (англ. sequence number)), и с каждым сегментом передается номер в последовательности его первого байта.

Поскольку два узла могут передавать два встречных потока данные по одному TCP-соединению, для передачи подтверждений одного потока используются сегменты встречного потока. В каждом сегменте передается номер в последовательности байта, который собирается принять данный узел.

После того, как модуль TCP передаст сегмент модулю IP, он записывает его копию в очередь на повторную передачу и запускает таймер для этого сегмента. Когда поступит подтверждение приема сегмента (то есть будет принят сегмент, в котором будет заявлено, что та сторона готова принять байт с номером, большим всех номеров байтов сегмента, ждущего повторной передачи), сегмент удаляется из очереди. Если подтверждение не поступает до срабатывания таймера, сегмент отправляется повторно.

Сегмент состоит из заголовка и поля данных. Формат заголовка сегмента TCP приведен на рисунке 2.7.

<table>
<tr><th colspan="8">1 байт</th><th colspan="8">2 байт</th><th colspan="8">3 байт</th><th colspan="8">4 байт</th></tr>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></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>
</table>

Checksum	Urgent pointer
Options (переменная длина)	Padding

Рисунок 2.7 — Формат заголовка TCP-сегмента

Поле Порт отправителя (англ. Source port) и Порт получателя (англ. Destination port) [16 бит] — номера портов на узлах.

Поле Номер в последовательности (англ. Sequence Number) [32 бита] — номер в потоке первого байта данных этого сегмента. Если установлен управляющий бит SYN, то в этом поле содержится начальный номер в последовательности (англ. Initial Sequence Number, ISN) и первый байт данных сегмента имеет номер в потоке ISN+1.

Поле Номер подтверждения (англ. Acknowledgement number) [32 бита] — номер байта в потоке, ожидаемого отправителем данного сегмента. При этом должен быть установлен управляющий бит ACK.

Поле Смещение данных (англ. Data offset) [4 бита] — количество 32-битовых слов в заголовке TCP-сегмента. Минимальное значение поля — 5 (20-байтовый заголовок).

Поле Резерв (англ. Reserved) [6 бит] — должны быть заполнены нулями.

Поле Биты управления (англ. Control bits) [6 бит] — от старшего к младшему:

- URG (англ. Urgent Pointer field significant) — принимать во внимание поле «Указатель срочности»;
- ACK (англ. Acknowledgement field significant) — принимать во внимание поле «Номер подтверждения»;
- PSH (англ. Push function) — сегмент был принудительно отправлен не дожидаясь заполнения данными;
- RST (англ. Reset the connection) — прервать связь;
- SYN (англ. Synchronize sequence numbers) — синхронизировать номера байтов в потоке;

- FIN (англ. No more data from sender) — отправитель больше не будет передавать данные.

Поле Окно (англ. Window) [16 бит] — размер окна — количество байтов данных, начиная с указанного в поле «Номер подтверждения», которые отправитель данного сегмента готов принять.

Поле Контрольная сумма (англ. Checksum) [16 бит] — контрольная сумма всего сегмента (заголовка и данных), вычисляется по алгоритму протокола IP. Как и в UDP, перед вычислением контрольной суммы к сегменту приписывается псевдозаголовок (той же структуры, что и в UDP).

Поле Указатель срочности (англ. Urgent Pointer) [16 бит] — содержит номер первого байта, имеющего обычный статус срочности. При этом должен быть установлен управляющий бит URG.

Поле Опции (англ. Options) [переменный размер] — дополнительная служебная информация. Подобно опциям заголовка IP-дейтаграммы, имеют переменную длину и могут вообще отсутствовать.

Поле Выравнивание (англ. Padding) — поле, используемое для доведения размера заголовка до целого числа 32-битовых слов.

Установление соединения

Прежде, чем процессы смогут обмениваться данными по TCP, необходимо установить TCP-соединение (англ. session); при этом используется процедура **трехэтапного приветствия** (англ. three-way handshake):

- Клиент генерирует случайное число, которое будет использовано в качестве ISN (т.е. клиент будет нумеровать отправляемые байты начиная с этого числа), и отправляет сегмент с установленным управляющим битом SYN и ISN в поле «Номер в последовательности».
- Сервер, получив сегмент с управляющим битом SYN, сохраняет пришедший в нем ISN клиента, генерирует свой ISN как случайное число, начиная с которого он будет нумеровать отправляемые байты, и

отправляет сегмент с установленными управляющими битами SYN и ACK, своим ISN в поле «Номер в последовательности» и полученным от клиента ISN+1 в поле «Номер подтверждения».

- Клиент, получив сегмент с управляющими битами SYN и ACK, сохраняет пришедший в нем ISN сервера и отправляет сегмент с установленным управляющим битами ACK, своим ISN в поле «Номер в последовательности» и полученным от сервера ISN+1 в поле «Номер подтверждения».

После того, как каждая из сторон получила ISN другой стороны и подтвердила его, по соединению можно передавать данные.

Отметим, что на SYN-сегмент расходуется одно значение номера в последовательности (равное ISN), хотя в нем не передаются байты пользовательских данных.

Разрыв соединения

Поскольку TCP-соединение по сути представляет собой два противоположно направленных канала передачи данных, для корректного разрыва соединения необходимо и достаточно закрыть оба этих канала.

Если одна из сторон больше не собирается передавать данные по соединению, она должна передать сегмент с установленным управляющим битом FIN. Получив такой сегмент, вторая сторона должна подтвердить его получение сегментом с установленным управляющим битом ACK. После этого один из пары каналов (тот, по которому передавался FIN-сегмент) считается закрытым. Второй канал закрывается аналогично (FIN-сегмент в одну сторону, ACK-сегмент в ответ), но по инициативе другой стороны. Соединение может быть наполовину закрытым сколь угодно долго.

Таким образом, для корректного разрыва соединения нужно передать четыре сегмента. Отметим, что, как и при установлении соединения, на каждый FIN-сегмент расходуется одно значение номера в последовательности, хотя в нем не передаются байты пользовательских данных.

Если один из компьютеров, установивших соединение, отключается от сети, не выполнив корректного разрыва соединения (например, при сбое ОС или при нарушении работы канала связи), то вторая сторона соединения разорвет соединение не сразу, а только после истечения достаточно длительного промежутка времени. Это дает возможность первому компьютеру продолжить обмен данными по этому соединению (например, после восстановления работоспособности канала связи).

Размер сегмента

При формировании TCP-сегментов желательно, чтобы они имели такой размер, чтобы несущие их IP-пакеты не нужно было фрагментировать. Это значит, что максимальный размер сегмента должен зависеть от того, по сетям каких технологий пролегает путь между двумя узлами, установившими TCP-соединение: если на этом пути встречаются сети с маленьким размером кадра (например, сети X.25 с размером кадра 512 байт), максимальный размер данных поля сегмента должен быть таким, чтобы размер IP-пакета (т.е. размер данных + размер TCP-заголовка (обычно 20 байт) + размер IP-заголовка (обычно 20 байт)), в котором он размещен, не превосходил минимальный размер кадра промежуточных сетей.

В любом случае, максимальный размер IP-пакета, несущего TCP-сегмент, не должен превышать максимального размера поля данных кадра сети, к которой непосредственно подключен узел. Например, для сетей Ethernet с максимальным размером поля данных кадра, равным 1500 байт (для кадров Ethernet II/DIX и 802.3), максимальный размер сегмента (англ. MSS, Maximum Segment Size) не должен превышать 1460 байт.

Если есть возможность увеличения MSS (например, известно, что обмен данными по TCP происходит только в локальной сети Ethernet), то ею не стоит пренебрегать: чем больше MSS, тем больше данных переносится в одном кадре, тем меньше накладные расходы и тем эффективнее передача данных.

По умолчанию (если в настройках TCP/IP не указано иное значение) размер сегмента равен 536 байт (поскольку максимальный размер IP-пакета по умолчанию равен 576 байт).

Окна приема

Протокол TCP имеет средства управления потоком данных: если принимающая данные сторона не успевает их обрабатывать, отправитель замедляет или приостанавливает передачу.

В процессе установки соединения каждая из сторон выделяет память под входной и выходной буферы и передает (в поле «Окно» того же SYN-сегмента, в котором передается ISN) второй стороне количество байт, которые она готова принять (не больше, чем размер своего входного буфера); это число в TCP называется **окном приема** (англ. receive window). Получив значение окна приема, вторая сторона сохраняет его в блоке управления передачей, связанном с соединением, и в процессе передачи данных следит за тем, чтобы объем отправленных, но неподтвержденных данных никогда не превышал размер окна приема.

В процессе передачи данных стороны могут изменять свои окна приема (размер окна приема передается в каждом сегменте в поле «Окно»): например, если приемник перегружен, он может передать сегмент с полем «Окно», равным нулю (чтобы отправитель приостановил передачу), а когда перегрузка приемника закончится, передать сегмент с нормальным значением в поле «Окно».

В начале передачи данных, пока приемник и передатчик не уверены в качестве соединяющих их каналов связи, обычно обе стороны устанавливают небольшое окно приема (размером в один MSS). По мере того, как приходят подтверждения, размер окна увеличивается экспоненциально (после первого подтверждения — два MSS, после второго — четыре MSS, после третьего — восемь MSS и т.д.) до некоторого порога, после которого растет линейно (с каждым новым подтверждением окно увеличивается на один MSS). Если хотя

бы один сегмент не подтвержден (или пришел сегмент с запросом повторной передачи), размер окна уменьшается вдвое с каждым неподтвержденным или перезапрошенным сегментом, пока сегменты не начнут снова подтверждаться. Дальше окно приема опять начинает расти.

Блок управления передачей и состояния соединения

Каждый раз при установлении соединения модуль TCP создает структуру данных — Блок управления передачей (англ. Transmission Control Block, TCB), хранящую постоянную информацию о соединении (IP-адреса, номера портов, указатели на входной и выходной буферы, очередь повторной отправки и т.д.) и текущие значения переменных, описывающих текущее состояние соединения. К этим переменным относятся:

- SND.UNA — не подтвержденная посылка;
- SND.NXT — следующий сегмент на отправку;
- SND.WND — окно передачи;
- SND.UP — указатель срочности для отправляемых данных;
- SND.WL1 — номер в последовательности, использованный для последней коррекции окна;
- SND.WL2 — номер подтверждения, использованный для последней коррекции окна;
- ISS — начальный номер в последовательности для отправки;
- RCV.NXT — следующий сегмент, который можно принять;
- RCV.WND — окно приема;
- RCV.UP — указатель срочности для принимаемых данных;
- IRS — начальный номер в последовательности для приема.

Кроме того, значения некоторых полей заголовка текущего сегмента тоже переносятся в переменные TCB:

- SEG.SEQ — номер в последовательности;
- SEG.ACK — номер подтверждения;

- SEG.LEN — длина сегмента;
- SEG.WND — размер окна;
- SEG.UP— указатель срочности;
- SEG.PRC — приоритет.

TCP-соединение в каждый момент времени находится в одном из следующих состояний:

- LISTEN (Прослушивание) — ожидание запроса на соединение;
- SYN-SENT (Синхронизация отправлена) — ожидание парного запроса на соединение (после того, как был отправлен запрос на соединение);
- SYN-RECEIVED (Синхронизация получена) — ожидание подтверждения после обмена запросами на соединение;
- ESTABLISHED (Установлено) — соединение установлено и может передавать пользовательские данные; основное рабочее состояние;
- FIN-WAIT-1 (Ожидание завершения 1) — ожидание запроса на завершение соединения или подтверждения своего запроса на завершение соединения;
- FIN-WAIT-2 (Ожидание завершения 2) — ожидание запроса на завершение соединения;
- CLOSE-WAIT (Ожидание закрытия) — ожидание запроса на закрытие соединения от своего прикладного процесса;
- CLOSING (Закрытие) — ожидание подтверждения запроса на закрытие соединения;
- LAST-ACK (Последнее подтверждение) — ожидание подтверждения запроса на закрытие соединения (в котором содержалось подтверждение запроса на закрытие соединения);
- TIME-WAIT (Ожидание) — пауза для уверенности, что дальняя сторона соединения получила подтверждение своего запроса на закрытие соединения
- CLOSED (Закрыто) — соединение закрыто.

2.7 Вопросы для самопроверки

- Необходимы ли символьные адреса (обеспечиваемые, например, серверами DNS) для обмена данными между узлами TCP/IP-сети?
- Для чего IP адрес состоит из двух полей — номера подсети и номера узла в подсети?
- Для чего используется IP адрес 255.255.255.255?
- Зачем нужны маски подсети?
- Как протокол IP обеспечивает надежную доставку дейтаграмм?
- Данными из какого поля заголовка IP-пакета пользуются маршрутизаторы для определения зациклившихся пакетов?
- Для чего служит поле «Протокол» в заголовке IP-пакета?
- Как получатель определяет, что два IP-пакета являются фрагментами одного «большого пакета»?
- Достаточно ли для передачи данных между двумя узлами локальной сети им знать IP-адреса друг друга?
- Что позволяет отправителю рассчитывать на то, что его ARP-запрос увидят все соседние узлы?
- Какое сообщение ICMP отправит маршрутизатор, если не сможет определить локальный адрес узла-получателя пакета, чей IP-адрес принадлежит к его локальной сети?
- Для чего транспортные протоколы используют порты?
- В каких случаях предпочтительно использовать протокол UDP?
- Как протокол TCP обеспечивает надежную доставку данных?
- Почему протокол TCP генерирует случайный исходный номер в последовательности для каждого соединения, а не просто нумерует байты, начиная с 0 или 1?
- Что такое «трехэтапное приветствие» в протоколе TCP?
- Для чего используются окна приема в протоколе TCP?

3 Маршрутизация в сетях TCP/IP

Протокол маршрутизации — это набор правил, описывающий способ передачи маршрутизаторами друг другу информации о доступных им сетях. На основе этой информации каждый маршрутизатор может выбирать наилучший путь передачи для каждого пакета.

Важно различать маршрутизируемые (англ. routed) протоколы и протоколы маршрутизации (англ. routing). Маршрутизируемый протокол — это тот протокол, ради которого выполняется маршрутизации, например, IP в сетях TCP/IP. Протокол маршрутизации (например, RIP, OSPF или IGRP) является служебным протоколом, обеспечивающим наличие у каждого маршрутизатора достаточной информации для выполнения маршрутизации IP-пакетов.

Основная задача маршрутизации — выбор для каждого пакета пути от исходного узла к узлу назначения. В объединенных TCP/IP сетях она сводится к более простой задаче нахождения шлюзов (маршрутизаторов), соединяющих подсети на этом пути. В каждой отдельной подсети пакеты доставляются средствами локальной технологии (как правило, канального уровня).

Информация, используемая маршрутизатором при выборе путей передачи пакетов, сводится в таблицу маршрутизации (англ. routing table). Как минимум, таблица маршрутизации должна содержать следующие столбцы:

- подсеть назначения;
- выходной интерфейс;
- расстояние до подсети назначения (метрика);
- IP-адрес следующего маршрутизатора.

Для того чтобы маршрутизатор мог выбрать из нескольких возможных путей наилучший, каждому пути сопоставляется числовое значение — метрика. Как правило, чем метрика меньше, тем путь лучше.

Ранние протоколы маршрутизации в качестве метрики использовали длину пути (т.е. количество промежуточных маршрутизаторов; используется также термин ‘хоп’, от англ. hop — hands-off points (точки передачи)). Однако

этот способ не позволяет учесть, например, разную пропускную способность каналов, что приводит к неэффективной работе. При использовании такой метрики, например, прямой канал через асинхронный модем (56 кбит/с, метрика равна 1) окажется предпочтительнее пути по двум каналам с пропускной способностью по 100 Мбит/с, но через лишний маршрутизатор (100 Мбит/с, метрика равна 2).

Современные протоколы маршрутизации используют сложные метрики, учитывающие следующие факторы:

- пропускная способность канала;
- время задержки;
- загруженность канала;
- надежность канала;
- количество промежуточных маршрутизаторов;
- стоимость пользования каналом.

Для просмотра таблицы маршрутизации в ОС семейства Windows используется команда ‘route PRINT’, в ОС семейства Unix — ‘netstat —r’, в ОС сетевого оборудования фирмы Cisco (IOS) — ‘show ip route’.

Топология сетей, особенно больших, может изменяться: подключаются и отключаются каналы, меняются настройки маршрутизаторов, происходят сбои в оборудовании. Маршрутизаторам необходимо поддерживать общее, согласованное с действительностью представление о сети. Если достигнуто именно такое положение дел, говорят, что сеть «сходится» (англ. converge). Время, которое требуется для того, чтобы сеть «сошлась» после изменения ее топологии, называется временем схождения. Время схождения зависит от протокола маршрутизации. Чем меньше время схождения, тем лучше.

В зависимости от предназначения протоколы маршрутизации делятся на протоколы внутренней маршрутизации (англ. Interior Gateway Protocol, IGP) и протоколы внешней маршрутизации (англ. Exterior Gateway Protocol, EGP). Внутренняя маршрутизация осуществляется в пределах автономной системы (англ. Autonomous System, AS) — совокупности сетей и подсетей с единым

управлением (например, принадлежащих одной организации). Протоколы внутренней маршрутизации должны выбирать наилучшие пути для пакетов по сетям, входящим в автономную систему, исходя только из технических характеристик сетей и каналов связи. Внешняя маршрутизации, напротив, гораздо сильнее зависит от установленных политик маршрутизации, чем от технических характеристик сетей. Так, например, может быт установлен запрет на передачу пакетов через соседнюю автономную систему, взимающую слишком высокую плату за трафик, невзирая на качество канала связи с ней.

Протоколы маршрутизации в зависимости от способа расчета маршрутов делятся на три типа: дистанционно-векторные протоколы (англ. distance vector routing protocol), протоколы на основе состояний связей (англ. link-state routing protocol) и гибридные протоколы.

Маршрутизаторы, выполняющие дистанционно-векторный протокол, периодически рассылают всем соседним маршрутизаторам обновления своей таблицы маршрутизации. Получив от соседа обновление, маршрутизатор соответственно корректирует свою таблицу маршрутизации и рассылает всю таблицу или только изменившуюся ее часть всем своим соседям.

Процесс рассылки обновлений (англ. routing update) выполняется достаточно часто: так протокол RIPv1 рассылает полную таблицу маршрутизации каждые 30 секунд, а протокол IGRP — только изменившуюся часть таблицы каждые 90 секунд.

Протоколы на основе векторов расстояний обычно используют простую метрику для определения расстояний между сетями: количество маршрутизаторов на всем протяжении пути. Для выбора маршрута используется алгоритм Беллмана-Форда.

Одна из основных проблем, связанная с этой группой протоколов — возможность создания бесконечных циклов при отключении отдельных подсетей. При этом два маршрутизатора могут быть уверены, что отключенная сеть доступна через соседа, и бесконечно передавать пакет друг другу. Для предотвращения таких ситуаций или уменьшения их негативного влияния

протоколы на основе векторов расстояний часто реализуют дополнительные функции, среди которых:

- счет до бесконечности (англ. count to infinity): при добавлении к таблице маршрутизации новых записей, полученных от соседа, все метрики в этих записях увеличиваются на 1; если метрика превысила 15 (бесконечность в данном случае принимается равной 16), то соответствующая сеть помечается как недоступная;
- расщепленный горизонт (англ. split horizon): маршрутизатор никогда не передает информацию о сетях, полученную от некоторого соседа, обратно этому соседу;
- расщепленный горизонт с ответом-заглушкой (англ. split horizon with poison reverse): до передачи обновлений по методу расщепленного горизонта, маршрутизатор присваивает записям об отключенных сетях метрику 16;
- пережидание (англ. hold-down): после того, как маршрутизатор пометил сеть, как недоступную, он в течение нескольких (по умолчанию — трех) периодов рассылки обновлений будет игнорировать любую информации о достижимости этой сети (кроме как от того маршрутизатора, который сообщил о недоступности сети);
- мгновенное обновление (англ. triggered updates): маршрутизатор будет рассылать обновления сразу после того, как изменилась метрика у какого-либо маршрута (а не будет дожидаться момента регулярной рассылки);
- балансировка нагрузки (англ. load balancing): если к сети назначения ведут несколько путей с одинаковыми метриками, маршрутизатор будет распределять поток пакетов по всем этим путям равномерно.

Целью работы протоколов на основе состояний связей является построение каждым из маршрутизаторов полного графа связности сети (как правило, с указанием характеристик связей — пропускной способности, загрузки каналов и т.д.) При этом маршрутизаторы передают друг другу не

полную свою таблицу маршрутизации, а только информацию о ребрах графа сети — связях «маршрутизатор — маршрутизатор» и «маршрутизатор — подсеть». На основании этой информации каждый маршрутизатор строит полный граф сети, затем для каждой достижимой сети ищет на этом графе путь с минимальной суммой метрик (как правило, по алгоритму Дейкстры) и заносит его в таблицу маршрутизации. Собственно маршрутизация пакетов осуществляется на основании таблицы маршрутизации (а не графа сети). Регулярные передачи информации между маршрутизаторами в таких протоколах выполняются относительно редко, а информация об изменениях в состояниях связей распространяется максимально быстро.

Основные достоинства таких протоколов — быстрая сходимость, возможность учета разных характеристик каналов связи, небольшая нагрузка на сеть при передаче служебной информации (и объем данных, и частота их передачи меньше, чем в протоколах на основе векторов расстояний).

Основные недостатки протоколов на основе состояний связей — большая вычислительная сложность, которая влечет необходимость применения более быстродействующих процессоров и большой размер графов сети, что влечет необходимость использования больших объемов оперативной памяти. Кроме того, в начале работы сети все маршрутизаторы обмениваются большими объемами топологической информации, что приводит к скачкообразному росту нагрузки на сеть. К этой группе протоколов относится OSPF.

Третий тип протоколов маршрутизации, гибридные протоколы, выбирают маршруты на основе векторов расстояний, но выполняют обновление таблиц сразу, как только изменяется топология. В результате гибридные протоколы сходятся практически так же быстро, как протоколы на основе состояний связей, но требуют гораздо меньше ресурсов (прежде всего памяти и процессорного времени). К гибридным протоколам относятся IS-IS и EIGRP.

3.1 Протокол маршрутизации RIP

Первая версия протокола RIP (англ. Routing Information Protocol version 1, RIPv1) использует технологию векторов расстояний и рассылает полную таблицу маршрутизации каждые 30 секунд. Максимальный диаметр сети — 15 переходов. Полное описание RIPv1 содержится в RFC 1058.

Поддержка RIP в маршрутизаторах может быть либо пассивной (только прослушивание RIP-сообщений и модификация своей таблицы маршрутизации), либо активной (и прослушивание, и рассылка RIP-сообщений). Узлы могут поддерживать RIP только в пассивном режиме.

При внесении в таблицу маршрутизации записи, полученной от другого маршрутизатора, с ней связывается таймер. Если за 180 секунд (шестикратный период рассылки) не будет получено повторное сообщение, содержащее этот же маршрут, запись станет недействительной.

Протокол RIP использует UDP-дейтаграммы, передаваемые через 520-й порт. Каждое сообщение протокола RIP состоит из заголовка фиксированной длины (4 байта, см. рисунок 3.1) и нескольких (от 1 до 25) записей (длиной по 20 байт, см. рисунок 3.2), соответствующих записям таблицы маршрутизации. Первая версия протокола RIP использует только часть полей, имеющихся в заголовке и записях; все остальные биты должны быть установлены в “0”. Если размер таблицы маршрутизации превышает 25 записей, то она передается в нескольких сообщениях.

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Команда								Версия								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Рисунок 3.1 — Формат заголовка сообщения протокола RIPv1

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Идентификатор адресной схемы																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IP-адрес																															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Метрика																															

Рисунок 3.2 — Формат записи сообщения протокола RIPv1

Поле Команда определяет тип операции, которую выполнит маршрутизатор, приняв это сообщение. Основные используемые значения: «1» — запрос, «2»— ответ.

Поле Версия хранит номер версии протокола RIP.

Поле Идентификатор адресной схемы указывает тип адреса (IP-адресам соответствует идентификатор «2»).

Поле IP-адрес хранит IP-адрес сети или узла назначения (кроме того, в этом поле может содержаться значение «0», соответствующее маршруту по умолчанию). Восемь следующих байт в случае IP-адресов должным быть заполнены нулями, но при других адресных схемах (с более длинными сетевыми адресами) могут хранить последующие байты адреса.

Поле Метрика хранит количество переходов от данного маршрутизатора до указанной сети или узла. При вычислении метрики в RIP маршрутизатор всегда учитывает самого себя, т.е. непосредственно подключенная сеть находится на расстоянии одного перехода, сеть, непосредственно подключенная к соседнему маршрутизатору — двух переходов и т.д. Администратор может вручную устанавливать большие метрики для сетей, подключенных через низкоскоростные каналы. Значение метрики, равное 16, указывает на отсутствие соединения с данной сетью (прием “poison reverse”).

Вторая версия протокола RIP (описанная в RFC 1388) поддерживает маски подсети, бесклассовую маршрутизацию CIDR и групповую передачу. Форматы заголовка и записи сообщения протокола RIP версии 2 показаны на рисунках 3.3 и 3.4. соответственно.

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Команда								Версия								Домен маршрутизации															

Рисунок 3.3 — Формат заголовка сообщения протокола RIPv2

1 байт								2 байт								3 байт								4 байт							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Идентификатор адресной схемы																Метка маршрута															
IP-адрес																															
Маска подсети																															
Следующий хоп																															
Метрика																															

Рисунок 3.4 — Формат записи сообщения протокола RIPv2

Поле Домен маршрутизации может использоваться при внешней маршрутизации. По умолчанию оно должно содержать нули.

Наиболее существенное отличие RIPv2 от RIPv1 — в наличии поля Маска подсети, позволяющего выполнять маршрутизацию не только для сетей классов A,B и C, но и для любых подсетей.

Поле Метка маршрута может использоваться, если RIP выполняет функции протокола внешней маршрутизации; в этом случае в это поле должен быть записан номер автономной системы.

Поле Следующий хоп содержит адрес ближайшего маршрутизатора, через который пролегает путь к данной сети (от маршрутизатора, сформировавшего данное RIP-сообщение).

В отличие от RIPv1, не поддерживающего аутентификацию отправителя сообщения, RIPv2 допускает замену первой записи после заголовка на запись аутентификации (см. рисунок 3.5), и в безопасном режиме игнорирует сообщения от неаутентифицированных источников.

1 байт								2 байт								3 байт								4 байт							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Тип аутентификации															
Данные для аутентификации																															
Данные для аутентификации (продолжение)																															
Данные для аутентификации (продолжение)																															
Данные для аутентификации (продолжение)																															

Рисунок 3.5 — Формат записи аутентификации сообщения протокола RIPv2

Протокол RIP хорошо подходит для использования в небольших сетях, в качестве протокола внутренней маршрутизации в автономных системах с небольшим количеством маршрутизаторов

3.2 Протокол маршрутизации OSPF

Протокол OSPF (англ. Open Shortest Path First) использует технологию состояний связей и описан в RFC 1247 и RFC 2328.

Сообщения OSPF передаются в IP-дейтаграммах со значением «89» в поле заголовка «Протокол». Информация о состоянии каналов широковещательно рассылается каждым маршрутизатором всем своим соседям каждые 30 минут (даже в том случае, если никаких изменений с момента прошлой рассылки не было). Если топология сети изменилась, сообщения об этом рассылаются немедленно.

Протокол OSPF позволяет администратору либо задавать метрики маршрутов произвольно (исходя из собственных, часто неформализуемых соображений), либо выбирать способ вычисления сложной метрики. Метрика по умолчанию — время передачи по каналу одного бита, деленное на 10 наносекунд (10^{-8} с), например, для Ethernet 10 Мбит/с эта метрика равна 10 ($(1/10^7)/10^{-8} = 10$), для Ethernet 100 Мбит/с — 1, а для модемного канала связи с пропускной способностью 28800 бит/с — 3472.

Каждый маршрутизатор строит описание графа сети (вершины — маршрутизаторы и сети, ребра — однонаправленные связи между ними) в виде базы данных состояний связей: для каждой связи хранится ее метрика. Графы, построенные всеми маршрутизаторами, должны совпадать. По базе данных состояний связей алгоритм Дейкстры вычисляет кратчайшие пути от заданной вершины (соответствующей данному маршрутизатору) до всех остальных вершин. Результат его работы заносится в таблицу маршрутизации.

Маршрутизаторы, соединенные отдельным каналом, или подключенные к одной сети, называются соседними. Соседние маршрутизаторы, обменивающиеся информацией о топологии сети, называются смежными.

На рисунке 3.6 показан заголовок сообщения протокола OSPF.

1 байт								**2 байт**								**3 байт**								**4 байт**							
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Версия								Тип								Длина сообщения															
IP-адрес маршрутизатора																															
Идентификатор области																															
Контрольная сумма																Тип аутентификации															
Данные для аутентификации																															
Данные для аутентификации (продолжение)																															

Рисунок 3.6 — Формат заголовка сообщения протокола OSPF

Поле Версия — номер версии протокола OSPF (текущая версия — 2).

Поле Тип — содержит тип сообщения из следующего перечня:

1 — проверка достижимости (англ. HELLO),

2 — описание базы данных (англ. Database description),

3 — запрос состояния связей (англ. Link state request),

4 — обновление состояния связей (англ. Link state update),

5 — подтверждение состояния (англ. Link state acknowledgement).

Поле Длина сообщения — длина сообщения (с заголовком) в байтах.

Поле IP-адрес маршрутизатора — адрес маршрутизатора, отправившего сообщение (должен быть выбран адрес одного из интерфейсов).

Поле Идентификатор области — номер области, к которой относится сообщение; обычно используется IP-адрес подсети.

Поле Контрольная сумма — вычисляется для всего сообщения (с заголовком, но без полей аутентификационных данных) по алгоритму протокола IP.

Поле Тип аутентификации: «0» — нет аутентификации,

«1» — аутентификация по паролю.

Поле Аутентификационные данные: если аутентификация используется, то в этом поле хранится, например, пароль.

3.3 Протоколы внутренней маршрутизации IGRP и EIGRP

Протокол IGRP (англ. Interior Gateway Routing Protocol, внутренний протокол маршрутизации шлюзов) разработан фирмой Cisco Systems в начале 1980-х и использует технологию векторов расстояний. Протокол IGRP используется только в маршрутизаторах фирмы Cisco. Его принципиальные отличия от RIPv1 сводятся к следующему:

- метрика вычисляется исходя из пропускной способности, задержки, нагрузки и надежности связей;
- максимальный диаметр сети — 255 хопов (по умолчанию — 100);
- период рассылки обновлений — 90 секунд;
- при использовании нескольких путей к одной сети нагрузка распределяется не равномерно, а пропорционально их метрикам;
- более экономная структура служебных пакетов.

В протоколе IGRP используются многие из перечисленных выше дополнительных функций протоколов на основе векторов расстояний: расщепленный горизонт (англ. split horizon), расщепленный горизонт с ответом-заглушкой (англ. split horizon with poison reverse), пережидание (англ. hold-down), мгновенное обновление (англ. triggered updates или flash updates).

Гибридный протокол EIGRP (англ. Enhanced Interior Gateway Routing Protocol, усовершенствованный внутренний протокол маршрутизации шлюзов) разработан фирмой Cisco Systems на основе IGRP в ответ на появление протокола OSPF. Как и IGRP, протокол EIGRP используется только в маршрутизаторах фирмы Cisco. Перечислим основные усовершенствования EIGRP по сравнению с IGRP:

- специальный алгоритм распространения информации об изменениях топологии сети — алгоритм DUAL (англ. Diffuse Update Algorithm, алгоритм распространения обновлений);
- поддержка бесклассовой адресации;
- поддержка других протоколов сетевого уровня (кроме IP);
- передача частичных обновлений таблицы маршрутизации.

Алгоритм DUAL основан на определении для каждой подсети назначения двух (а не единственного, как в других протоколах) маршрутизаторов: преемника (англ. successor) и возможного преемника (англ. feasible successor). Для каждого возможного маршрута до подсети назначения вычисляются две метрики: объявленное расстояние (англ. advertised distance) и возможное расстояние (англ. feasible distance). Объявленное расстояние вычисляется как сумма метрик всех связей, составляющих маршрут, кроме самой первой. Возможное расстояние — это просто сумма метрик всех связей, составляющих маршрут.

Маршрутизатором — преемником для подсети назначается тот из соседних маршрутизаторов, через который проходит маршрут с минимальным значением объявленного расстояния. Маршрутизатором — возможным преемником для подсети назначается тот из соседних маршрутизаторов, через который проходит маршрут, объявленного расстояние для которого меньше, чем возможное расстояние для маршрута через преемника. Возможный преемник используется для доставки пакетов к подсети назначения в том случае, если доставка пакетов через преемника невозможна.

3.4 Протоколы внешней маршрутизации EGP и BGP

Напомним, что автономная система (AS) — это совокупность сетей, с единым организационным управлением, соединенных группой маршрутизаторов, имеющих единые правила маршрутизации. Всеми остальными маршрутизаторами сети данная автономная система

воспринимается как единое целое. Каждой автономной системе Реестром Интернета назначается уникальный идентификационный номер. Каждая автономная система может самостоятельно и независимо от других автономных систем выбирать протокол (или протоколы) внутренней маршрутизации.

Различают следующие типы автономных систем: одноканальные (англ. single-homed), многоканальные без транзита (англ. multi-homed nontranzit) и многоканальные с транзитом (англ. multi-homed tranzit).

Одноканальная автономная система имеет единственное соединение с Интернетом, следовательно, весь ее внешний трафик должен по умолчанию отправляться по этому соединению, а вся работа по оповещению других автономных систем о маршрутах к данной автономной системе должна выполняться маршрутизаторами провайдера, предоставляющего это соединение.

Автономная система называется многоканальной, если она имеет несколько соединений с Интернетом. Автономная система без транзита не разрешает передавать через себя чужой трафик, для чего сообщает другим автономным системам только свои собственные маршруты (т.е. маршруты до своих пограничных маршрутизаторов). Автономная система с транзитом разрешает транзитное прохождение чужого трафика и сообщает не только собственные маршруты, но и маршруты, полученные ими от других автономных систем.

Протоколы внешней маршрутизации используются для определения маршрутов передачи данных между автономными системами. Метрики маршрутов в таких протоколах рассчитываются не относительно своих интерфейсов маршрутизатора, а относительно некоторой сети (называемой исходной сетью, англ. Source Network), достижимой из всех сетей данной автономной системы.

Протокол EGP (англ. Exterior Gateway Protocol, внешний протокол шлюзов) был первым протоколом внешней маршрутизации в Интернете и описан в RFC 904. В настоящее время стандартом де-факто является протокол

BGP (англ. Border Gateway Protocol, граничный протокол шлюзов) версии 4 описанный в RFC 1771. Он более интеллектуален, чем EGP, в частности, использует метрики для выбора каналов и умеет предотвращать образование петель.

С точки зрения протокола BGP, весь Интернет представляет собой граф, узлами которого являются автономные системы с уникальными идентификаторами, а ребрами — соединения между автономными системами. от Последовательность ребер, проходя по которым можно попасть из одной автономной системы в другую, называется маршрутом. Каждый маршрутизатор BGP хранит таблицу маршрутов от своей автономной системы до всех достижимых автономных систем.

Маршрутизаторы, поддерживающие BGP, называются спикерами (англ. speaker) BGP. Два спикера BGP, устанавливающие соединение друг с другом, называются соседями (англ. neightbors) или одноранговыми (англ. peers). Для соединения спикеры BGP используют порт 179 протокола TCP.

В начале сеанса соседи обмениваются всей имеющейся у них маршрутной информацией, а в дальнейшем передают друг другу только информацию о новых и удаленных маршрутах — инкрементные обновления (англ. incremental updates) — в пакетах UPDATE. Если никаких изменений в структуре маршрутов нет, соседи периодически обмениваются только маленькими пакетами KEEPALIVE (размером 19 байт), подтверждающими наличие соединения.

Заголовок сообщения в BGP состоит из трех полей, приведенных на рисунке 3.7.

<table>
<tr><th colspan="8">1 байт</th><th colspan="8">2 байт</th><th colspan="8">3 байт</th><th colspan="8">4 байт</th></tr>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="32">Маркер (16 байт)</td></tr>
<tr><td colspan="32"></td></tr>
<tr><td colspan="32"></td></tr>
<tr><td colspan="32"></td></tr>
<tr><td colspan="24">Длина</td><td colspan="8">Тип</td></tr>
</table>

Рисунок 3.7 — Формат заголовка сообщения протокола BGP

Поле маркера используется для аутентификации (с использованием алгоритма MD5) входящих сообщений. В поле длины хранится длина сообщения в байтах, включая заголовок. Наименьшая длина сообщения — 19 байт, наибольшая — 4096 байт. Поле типа определяет один из четырех типов сообщения:

- OPEN (открытие соединения),
- UPDATE (обновление маршрутной информации),
- NOTIFICATION (сообщение об ошибке),
- KEEPALIVE (подтверждение наличия соединения).

Информация о действующих маршрутах передается в сообщениях UPDATE, каждое из которых состоит из трех частей: недостижимые маршруты (т.е. те маршруты, которые были в таблице маршрутизации, но стали недействительными), атрибуты маршрута и информация о доступности сети NLRI (англ. Network Layer Reachability Information),

3.5 Вопросы для самопроверки

- Какие протоколы маршрутизации требуют больше оперативной памяти на маршрутизаторе — дистанционно-векторные или состояния связей?
- Какие недостатки первой версии протокола RIP привели к разработке второй его версии?
- Чем отличаются метрики в протоколах RIP и OSPF?
- Для чего протокол OSPF строит полный граф сети?
- Какие проблемы позволяет предотвратить метод расщепленного горизонта (split horizon)?
- Для чего используется мгновенное обновление (triggered update)?
- Что такое «исходная сеть» (source network) в протоколе BGP?

4 Протоколы и службы на основе TCP/IP

4.1 Служба DNS

Служба именования доменов (DNS, Domain Name System) описана в RFC и предназначена для установления глобального соответствия между символическими имена узлов и их IP-адресами. На ранних этапах развития TCP/IP сетей нужды в распределенной службе имен не было: существовал один текстовый файл, в котором были перечислены все имена узлов и их IP-адреса. Однако с ростом количества узлов такое решение стало неприемлемым.

DNS использует иерархическую схему выделения имен, позволяя децентрализовать управление отдельными участками пространства имен.

DNS-сервер домена должен поддерживать таблицу соответствий IP-адресов и символических имен для всех узлов, входящих в домен.

Каждый узел домена должен знать только IP-адрес своего сервера домена и направлять ему все запросы на преобразование символических имен узлов в IP-адреса (такие запросы называются рекурсивными (англ. recursive); сервер, получивший рекурсивный запрос, должен самостоятельно его обслужить и вернуть либо IP-адрес, соответствующий запрошенному имени, либо сообщение об ошибке). Если сервер домена не может самостоятельно ответить на запрос (например, запрошен IP-адрес узла, принадлежащего к другому домену), он может обратиться к корневому DNS-серверу, узнать он него IP-адрес сервера требуемого домена, а затем обратиться уже к этому серверу домена. Такие запросы от одного сервера к другому называются итеративными (англ. iterative); сервер, получивший итеративный запрос, может либо обслужить его полностью, либо выдать IP-адрес другого сервера, обладающего более полной информацией. Естественно, DNS-сервер может запоминать результаты подобных поисков и при повторном обращении за той же информацией не повторять весть путь, а сразу выдавать кэшированные данные.

DNS-сервер хранит, в первую очередь, пары «символическое имя—IP-адрес» для всех узлов своего домена. Кроме того, в базе данных DNS-сервера

имеется информация об адресах всех DNS-серверов данного домена (для повышения надежности поддерживается, как правило, не менее двух DNS-серверов для каждого домена, причем все они должны располагаться в разных IP-подсетях), списки псевдонимов (один и тот же узел может иметь несколько символических имен), а также список почтовых серверов данного домена.

Для передачи запросов на разрешение адресов (и ответов на них) протокол DNS использует UDP-дейтаграммы, передаваемые через порт 53. Для повышения безопасности может использоваться протокол TCP (также порт 53).

4.2 Протоколы сетевого управления

Для управления сетевым оборудованием создано два протокола: SNMP (англ. Simple Network Management Protocol, RFC 1157, 1215, 1187, 1089; разработан в 1988 году) и CMOT (англ. Common Management Information Services and Protocol Over TCP/IP, RFC 1095, в последнее время применение этого протокола ограничено). Протокол SNMP для передачи сообщений использует UDP и предназначен для использования сетевыми управляющими станциями. Он позволяет управляющим станциям собирать информацию о положении в TCP/IP-сети. Протокол определяет формат данных, а их обработка и интерпретация остаются на усмотрение управляющих станций или менеджера сети. SNMP-сообщения не имеют фиксированного формата и фиксированных полей. При своей работе SNMP использует управляющую базу данных (англ. Management Information Base, MIB, RFC 1213, 1212).

Алгоритмы управления в Интернет обычно описывают в нотации ASN.1 (abstract syntax notation). Все объекты в Интернет разделены на 10 групп и описаны в MIB: система, интерфейсы, обмены, трансляция адресов, IP, ICMP, TCP, UDP, EGP, SNMP. В группу «система» входит название и версия оборудования, операционной системы, сетевого программного обеспечения и т.п. В группу «интерфейсы” «ходит число поддерживаемых интерфейсов, тип интерфейса, работающего под IP (Ethernet, LAPB и т.п.), размер дейтаграмм,

скорость обмена, адрес интерфейса. Группа «IP» включает в себя время жизни дейтаграмм, информация о фрагментации, маски подсетей и т.д. В группу «TCP» входят параметры алгоритма повторной пересылки — максимальное число повторных пересылок и т.д. В таблице 4.1 приведен перечень команд SNMP.

Таблица 4.1 — Команды SNMP

Команда SNMP	**Код команды**	**Назначение**
GET request	0	Получить значение указанной переменной
GET next request	1	Получить значение следующей переменной
SET request	2	Изменить значение переменной
GET response	3	Ответ на команды с кодами 0-2, содержит информацию о состоянии
TRAP	4	Уведомление сетевого объекта о наступлении события или изменения состояния
GetBulkRequest	5	Запрос пересылки большого объема данных
InformRequest	6	Указание на информацию в
SNMPv3 TRAP	7	Уведомление о наступлении события (новые возможности протокола версии 3)
Report	8	Отчет

SNMP-клиент ожидает поступления дейтаграмм на UDP-порт 161. Если SNMP-менеджер заказывает уведомление, то клиент отправляет UDP-дейтаграмму с уведомлением на порт 162 менеджера.

4.3 Вопросы для самопроверки

- Какие запросы отправляет DNS-сервер (с пустым кэшем), которому пришел запрос на поиск узла www.mgapi.edu?
- Может ли узел, не являющийся DNS-сервером, выдавать итеративные запросы?
- Какая информация предоставляется узлами по протоколу SNMP?

5 Технологии X.25, FRAME RELAY, PDH, SDH

5.1 Технология X.25

Глобальные сети X.25 основаны на коммутации пакетов, и долгое время являлись единственными сетями с коммутацией пакетов, гарантирующими готовность. Структурно сеть X.25 состоит из **аппаратуры передачи данных** (АПД, англ. Data Terminal Equipment, DTE) — терминалы, компьютеры и т.д., **аппаратуры окончания канала данных** (АОКД, англ. Data Circuit-Terminating Equipment, DCE) —телекоммуникационного оборудования, модемов и т.п. и **центров коммутации пакетов** (ЦКП, англ. Packet Switching Exchange, PSE). Кроме того, применяются **сборщики-разборщики пакетов** (СРП, англ. Packet Assembler/Disassembler, PAD), буферизующие потоки байтов от асинхронных “тупых” терминалов и преобразующие их в потоки пакетов (и производящие обратные преобразования).

Рекомендации ITU-T X.25 «Интерфейс между оконечным оборудованием данных и аппаратурой передачи данных для терминалов, работающих в пакетном режиме в сетях передачи данных общего пользования» охватывают три нижних уровня модели ВОС.

На физическом уровне определены последовательные синхронные интерфейсы X.21 и X.21bis (допустимо использование интерфейсов V.24, V.35, RS-232C, RS-449, G.703). Физический уровень не контролирует правильность передачи.

Интерфейс X.21

Цепь сигнальной земли G (англ. ground) устанавливает общий уровень, относительно которого измеряются уровни сигналов. Цепь передачи T (англ. transmit) используется для передачи данных от АПД к АОКД, т.е. от узла в сеть. Цепь приема R (англ. receive) используется для приема данных АПД от АОКД. Цепь управления C (англ. control) используется АПД для того, чтобы сообщить АОКД состояние интерфейса: на протяжении всего времени передачи данных

по цепи T АПД поддерживает цепь C в активном состоянии. Аналогичную функцию несет цепь индикации I (англ. indication): АОКД переводит ее в активное состояние, чтобы сообщить АПД о передаче данных по цепи R.

По цепи битовой синхронизации S (англ. signal element timing) АОКД постоянно передает синхросигнал. Цепь байтовой синхронизации B (англ. byte timing) также управляется АОКД: цепь B активна во время передачи всех битов байта, кроме последнего, а при передаче последнего бита — пассивна.

На канальном уровне сеть гарантирует доставку, обеспечивает целостность данных и управление потоком. Канальный уровень реализуется подмножеством протокола HDLC — протоколом LAP-B (Link Access Protocol — Balanced). Протокол LAP-B функционирует подобно протоколу LLC2 (IEEE 802.2), то есть требует передачи квитанций и выполняет повторную передачу потерянных пакетов.

На сетевом уровне определен протокол X.25/3 (англ. Packet Layer Protocol, PLP), описывающий обмен пакетами между АПД и СПД. Он выполняет функции маршрутизации пакетов, установления и разрыва виртуального канала, управления потоком пакетов.

После установления соединения АПД с коммутатором на канальном уровне, АПД передает пакет запроса виртуального соединения с другим АКП (англ. Call Request), который маршрутизируется коммутаторами, прокладывая виртуальный канал.

Протокол X.25 допускает длину поля данных пользователя в пакете до 4096 байт, но предпочтительной является длина по умолчанию 128 байт.

Адресация узлов в сети X.25, не связанной с другими сетями, может быть произвольной (до 16 байт на адрес). Для сетей передачи общего пользования Рекомендация ITU-T X.121 определяет международную систему нумерации адресов. Первая цифра указывает на формат остальной части адреса: «0» — полный международный сетевой адрес (три следующих десятичных цифры указывают на страну, например, 250 или 251 — Россия, затем одна цифра под

номер сети в стране, затем до десяти цифр номер узла в сети), «9» — полный международный телефонный номер (следующие цифры содержат телефонный код страны, например, 7 — Россия, затем код города и номер телефона в городе, всего до 14 цифр).

Производительность коммутаторов X.25 обычно составляет несколько тысяч пакетов в секунду, что существенно ограничивает пропускную способность сети. Это связано с тем, что протоколы X.25 предназначались для использования на низкоскоростных линиях связи с высоким уровнем помех, поэтому каждый коммутатор должен подтвердить каждый принятый пакет, выполнить разбор пакета для определения дальнейшего пути и повторную его упаковку в кадр LAP-B. При этом задержка коммутации составляет сотни миллисекунд.

5.2 Технология Frame Relay

Сети с **ретрансляцией кадров** (англ. Frame Relay) представляют собой сети с коммутацией пакетов, ориентированные на цифровые линии связи со скоростью до 45 Мбит/с (изначально — до 2 Мбит/с). Frame Relay работает в дейтаграммном режиме с малыми задержками, не гарантирует доставку, целостность данных и управление потоком. При этом сети Frame Relay могут гарантировать среднюю скорость передачи данных по виртуальному каналу при допустимых пульсациях трафика (подобные гарантии качества обслуживания сегодня предоставляет только технология ATM).

Технология Frame Relay происходит из сетей ISDN, где в начале 1980-х она была стандартизована как одна из служб пакетного режима. Frame Relay предназначена для динамического разделения пропускной способности физического канала между отдельными процессами передачи данных (фактически, сети Frame Relay используются не столько для соединения отдельных узлов, сколько для соединения отдельных локальных сетей). При использовании Frame Relay предполагается, что канал передачи данных

достаточно надежен, что позволяет перенести контроль ошибок и управление потоком на вышележащие уровни. В результате кадры Frame Relay несут минимальное количество служебной информации и максимально быстро обрабатываются сетевым оборудованием.

Frame Relay обеспечивает установление постоянных (англ. Permanent Virtual Circuit, PVC) и коммутируемых (англ. Switched Virtual Circuit, SVC) виртуальных каналов. В отличие от X.25, Frame Relay передает кадры только по протоколам физического и канального уровня, не затрагивая сетевой уровень.

Протокол канального уровня LAP-F является упрощенной версией протокола LAP-D, работающего в сетях ISDN по D-каналам. В основном режиме LAP-F кадры передаются без преобразований и контроля, как при коммутации в локальных сетях. Хотя для кадров используется синхронный формат HDLC с длиной поля данных до 4 Кбайт и двухбайтовым полем CRC, значение CRC коммутаторами не проверяется. Пакеты могут аннулироваться в случае перегрузки сети.

Идентификация виртуального канала, к которому относится пакет, происходит по 10-битовому полю заголовка кадра DLCI (англ. Data Link Connection Identifier). При необходимости размер DLCI можно довести до 22 бит.

Основу сети Frame Relay образуют специализированные коммутаторы — FRAD (англ. Frame Relay Access Device, устройство доступа к сети с ретрансляцией кадров).

5.2.1 Структура кадра Frame Relay

Структура кадра Frame Relay приведена на рисунке 5.1.

1 байт	**2 байт**	**3 байт**	**4 байт**	**…**	**…**	**N-3 байт**	**N-2 байт**	**N-1 байт**	**N байт**
Флаг	Заголовок		Данные пользователя				Контрольная сумма		Флаг

Рисунок 5.1 — Структура кадра Frame Relay

Поля «Флаг» обозначают начало и конец кадра. Двоичное значение этого поля — ‘01111110’.

Поле «Данные пользователя» может иметь размер до 4056 байт и предназначено для данных, передаваемых протоколами верхних уровней.

Поле «Контрольная сумма» содержит 16-ти разрядную контрольную сумму для полей «Заголовок» и «Данные пользователя».

Поле «Заголовок» несет информацию, необходимую для управления передачей данных и имеет формат, приведенный на рисунке 5.2.

7 бит	**6 бит**	**5 бит**	**4 бит**	**3 бит**	**2 бит**	**1 бит**	**0 бит**
DLCI						C/R	EA0
DLCI				FECN	BECN	DE	EA1

Рисунок 5.2 — Структура поля Заголовок кадра Frame Relay

Поля EA0 И EA1 (англ. Effective Address, исполнительный адрес) управляют размером заголовка. Если бит EA сброшен (EA0), то в следующем байте содержатся дополнительные биты DLCI. Если бит EA установлен (EA1), то данный байт — последний в заголовке. Минимальный заголовок приведен на рис. и состоит из двух байт, в первом из которых бит EA сброшен, а во втором — установлен. Возможны также трех- и четырехбайтовые заголовки, в которых все байты, кроме последнего, имеют признак EA0, а последний — EA1.

Поле DLCI (англ. Data Link Connection Identifier, идентификатор виртуального соединения) используют коммутаторы (FRAD) для указания друг другу какие данные передаются в этом кадре. При двухбайтовом заголовке поле имеет длину 10 бит, при трехбайтовом — 16 бит, а при четырехбайтовом — 22 бита. Стандарт резервирует интервалы значений DLCI 0..15 и 992..1023 для служебных целей, внутрисетевых соединений и управления канальным уровнем. Пользователями для нумерации PVC и SVC могут использоваться 976 идентификаторов DLCI с номерами от 16 до 991.

Поля FECN (англ. Forward Explicit Congestion Notification, явное уведомление о заторе в прямом направлении) и BECN (англ. Backward Explicit Congestion Notification, явное уведомление о заторе в обратном направлении) используются коммутаторами при возникновении перегрузок в сети. Если коммутатор получает больше кадров, чем он может обработать, то он устанавливает в кадрах, отправляемых источнику избыточных данных, бит BECN, а в кадрах, отправляемых получателю избыточных данных — бит FECN. Поступление кадра с установленным битом BECN означает, что часть выдаваемых кадров может быть отброшена коммутаторами и нужно замедлить выходной поток. Поступление кадра с установленным битов FECN означает, что в данном потоке возможны (хотя и не обязательно произойдут) выпадения кадров.

Поле DE (англ. Discard Eligibility, приемлемость удаления) устанавливается отправителем кадра и означает, что данный кадр при возникновении перегрузок можно удалить.

Поле C/R (англ. Command/Reply, команда/ответ), когда равно 1, в кадрах, содержащих команды, требует, чтобы на команду был дан ответ, а в кадрах, содержащих ответы, указывает на последний кадр ответа.

5.3 Плезиохронная цифровая иерархия

Плезиохронная цифровая иерархия (англ. Plesiochronous Digital Hierarchy, PDH) разработана корпорацией AT&T (Bell Labs) в 1960-х годах для передачи множества потоков оцифрованной голосовой информации по каналам связи. Основная цель разработки заключалась в повышении скорости многоканальной связи крупных телефонных коммутаторов друг с другом.

Цифровое представление звуковых сигналов

Канал, выделяемый телефонными сетями для одного соединения, имеет полосу пропускания 4 кГц, достаточную для приемлемой передачи

человеческой речи. Соответственно, характеристики базового цифрового канала выбирались так, чтобы по одному такому каналу можно было передавать данные одного телефонного соединения. В соответствии с теоремой Котельникова-Найквиста, для того, чтобы было можно восстановить исходный сигнал, частота дискретизации должна быть не меньше 2*4 кГц = 8 кГц. Для приемлемого представления человеческой речи достаточно 12-ти бит на отсчет, что соответствует 4096-ти различимым уровням сигнала, а логарифмическое преобразование позволяет снизить разрядность отсчетов до 8 бит, сохраняя субъективное качество сигнала. Логарифм является положительной функцией только при аргументе, превышающем единицу, соответственно, для диапазона аргументов от 0 до 1 необходимо использовать какую-то другую функцию. В Европе и США используют разные преобразования — A-зависимость и μ-зависимость соответственно. В Европе для «малых» аргументов используют линейную функцию $y \sim ax$, а для «больших» — непосредственно логарифмическую: $y = (1+\ln A\, x)/(1+\ln A)$, где $A = 87{,}6$. В США сдвигают график функции на единицу в сторону оси ординат: $y \sim \log(1+\mu x)$.

Поскольку количество разных аргументов невелико (4096), на практике не вычисляют для каждого отсчета соответствующий логарифм, а хранят заранее подготовленную таблицу соответствий аргументов и значений функции.

Мультиплексор T1 имеет 24 аналоговых канала и один цифровой. Он постоянно перебирает аналоговые каналы (обращаясь к каждому из них 8000 раз в секунду, т.е. с частотой 8 кГц), оцифровывает поступающие аналоговые данные (телефонные разговоры) по 12 бит на отсчет, выполняет логарифмическое преобразование и выдает полученный байт данных в цифровой канал. Этот единственный байт составляет кадр DS-0. Время, отводимое на передачу одного байта, принадлежащего одному аналоговому каналу, называется таймслотом (англ. timeslot временной интервал). Отдельный таймслот отводится для синхронизации. В T1 для синхронизации используется один бит (F-бит, англ. Framing bit) поочередно ноль и единица. Таким образом,

по одному цифровому каналу T1 передаются 24 базовых голосовых канала, а группа, состоящая из 24 байт и F-бита, называется кадром DS-1. За одну секунду передается 8000 кадров DS-1. Суммарная скорость канала T1 составляет (24*8+1)*8=1544 Кбит/с.

Если по какому-либо из аналоговых каналов не поступают данные, его таймслот остается закрепленным за ним, соответственно, часть пропускной способности цифрового канала расходуется впустую.

Демультиплексор T1 выполняет обратную задачу — в него поступает поток кадров DS-1, из которых он извлекает по одному байту для каждого из 24-х аналоговых каналов, выполняет цифро-аналоговое преобразование и выдает его результат в канал.

Управляющая информация в T1 передается младшим разрядом байтов данных (поскольку байт представляет собой значение замера голоса, было сочтено, что искажение младшего разряда не должно быть замечено слушателем). В ранних версиях младший бит каждого байта был служебным, фактически передавались 7-битовые байты, а скорость передачи пользовательских данных составляла 56 Кбит/с. Затем для служебных целей использовался только каждый шестой кадр: в пяти кадрах в каждом байте передаются восемь бит пользовательских данных, а в шестом — только семь.

Четыре канала T1 объединяются в канал T2 (следующий уровень иерархии PDH), семь каналов T2 — в T3, шесть каналов T3 — в T4. Аппаратура T1, T2, T3 и T4 может взаимодействовать, образуя сеть с иерархией каналов.

Кадр DS-2 состоит из четырех кадров DS-1, разделенных F-битами, а сами кадры DS-2 разделяются 12 служебными синхробитами.

Позже эта технология (с некоторыми отличиями от оригинального варианта) была стандартизована ITU-T (в то время CCITT). В Америке, Канаде и Японии используется исходная американская версия, а в Европе — стандарт ITU-T. Базовый канал в обеих версиях имеет скорость 64 Кбит/с. Основное отличие европейских каналов — в кратности вхождения низкоскоростных каналов в канал следующего уровня, и, соответственно, их скорости. Канал E1

(аналог T1) состоит из 30 базовых каналов, канал E2 — из 4 каналов E1, канал E3 — из 4 каналов E2, а канал E4 — из 4 каналов E3.

Стандарт ITU-T (G.700-G.706) отказался от использования отдельных разрядов байтов пользовательских данных для передачи служебной информации. Кадр DS-1, передаваемый по каналу E1, состоит из 30 байт пользовательских данных (по одному из каждого базового канала) и 2 служебных байт. Суммарная скорость составляет 32*8*8=2048 Кбит/с.

В технологии PDH (стандарт ITU-T G.704) все уровни скоростей (и форматы кадров для этих уровней) называются DS-n, где n — номер уровня (DS — от англ. Digital Signal, цифровой сигнал).

В таблице 5.1 приведены сводные данные об уровнях скоростей американской и европейской плезиохронных иерархий.

Таблица 5.1 — Уровни скорости PDH

	Америка			**Европа**		
Уровень скорости (канал)	**Кол-во каналов пред. уровня**	**Кол-во базовых каналов**	**Скорость Кбит/с**	**Кол-во каналов пред. уровня**	**Кол-во базовых каналов**	**Скорость Кбит/с**
DS-0	1	1	64	1	1	64
DS-1 (T1/E1)	24	24	1544	30	30	2048
DS-2 (T2/E2)	4	96	6312	4	120	8488
DS-3 (T3/E3)	7	672	44736	4	480	34368
DS-4 (T4/E4)	6	4032	274176	4	1920	139264

На физическом уровне технология PDH (зафиксированная в стандарте ITU-T G.703) допускает использование витой пары, коаксиального и волоконно-оптического кабеля. В каналах T1/E1 используется преимущественно витая пара (две пары с волновым сопротивлением 120 Ом, разъем RJ-48), данные передаются с использованием кодов AMI/B8ZS (T1) и HDB3 (E1). Для каналов T2/E2 обычно используется коаксиальный кабель с волновым сопротивлением 75 Ом, а для T3/E3 — коаксиальный или волоконно-оптический кабель.

Основная проблема при использовании PDH — сложность выделения (демультиплексирования) пользовательских каналов. Это связано с использованием служебных бит синхронизации между кадрами. Если нужно выделить один базовый канал из кадров канала T3, нужно произвести полное демультиплексирование в кадры T2, кадр T2 — в кадры T1, а из кадра T1 выделить данные одного базового канала. Для уменьшения количества операций мультиплексирования используются специальные приемы, усложняющие работу сети и требующие специальной настройки.

Другой недостаток PDH — слабые средства управления сетью, недостаточное количество информации о состоянии канала, отсутствие процедур поддержки отказоустойчивости.

Наконец, предел скорости технологии PDH — 274 Мбит/с (T4) и 139 Мбит/с (E4), в то время, как современные кабели позволяют передавать данные со скоростями на порядок выше.

5.4 Синхронная цифровая иерархия

Синхронная цифровая иерархия (англ. Synchronous Digital Hierarchy, SDH) разработана компанией Bellcore в середине 1980-х годов под названием «Синхронные оптические сети» (англ. Synchronous Optical NETs, SONET), а в 1988 году была стандартизована CCITT и ANSI (G.707-G.709). Европейский (CCITT) и американский (ANSI) стандарты совместимы на скоростях, начиная с 155,52 Мбит/с (кадры STM-1 и STS-3 соответственно). В стандарте ANSI, кроме, того предусмотрена скорость 51,38 Мбит/с (кадр STS-1).

Основное достоинство SDH по сравнению с PDH — прозрачность мультиплексирования и демультиплексирования. Кадры SDH всех уровней имеют такую структуру, что позволяют легко, не разбирая на составляющие весь высокоскоростной поток, выделять и вставлять данные, относящиеся к требуемому низкоскоростному каналу. Для этого используется побайтовое чередование при мультиплексировании.

Сети SDH/SONET рассматриваются как состоящие из пяти уровней (см. рисунок 5.3).

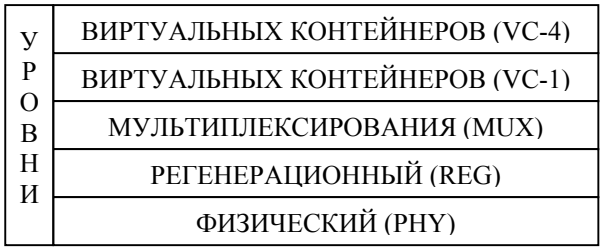

Рисунок 5.3 — Многоуровневая модель SDH/SONET

Нижний уровень — физический (в стандарте он называется фотонным), он соответствует среде передачи. Как правило, в качестве среды передачи используется оптоволокно, но допустимы также радио- и спутниковые каналы. Второй уровень — регенерационный, он соответствует **линии** (англ. line) — участку сети между регенераторами (восстановителями сигнала). Для передачи служебных сигналов на этом уровне в кадре предусмотрена группа служебных байтов — **заголовок секции регенерации** (англ. Regeneration Section OverHead, RSOH), занимающая первые девять байт в первых трех строках кадра STM-1. Третий уровень — мультиплексирования, он соответствует **секции** (англ. section) — участку сети между мультиплексорами. Для передачи служебных сигналов на этом уровне в кадре предусмотрена группа служебных байтов — **заголовок секции мультиплексирования** (англ. Multiplex Section OverHead, MSOH), занимающая первые девять байт с пятой по девятую строку кадра STM-1. Два следующих уровня — виртуальных контейнеров VC-4 и VC-1 — соответствуют **тракту** (англ. path) и выполняют передачу данных между двумя терминалами. Виртуальные контейнеры VC-4 используются для передачи данных ATM или высокоскоростных каналов PDH (E4), а VC-1 — для передачи низкоскоростных каналов (T1/E1). Контейнер VC-1 имеет две разновидности: VC-11 для T1 (1544 Кбит/с) и VC-12 для E1 (2048 Кбит/с).

Мультиплексирующиеся в синхронный поток кадров низкоскоростные (плезиохронные, ATM, IP) потоки, а также каналы, по которым они передаются, в терминологии SHD/SONET называются “приточными” (англ.).

В синхронных сетях используется четыре типа оборудования: регенераторы, мультиплексоры терминалов, добавляющие/извлекающие мультиплексоры, цифровые кросс-панели.

Регенератор (англ. regenerator) предназначен для восстановления проходящего через него ослабленного и расплывшегося цифрового сигнала. Мультиплексор терминалов (англ. terminal multiplexer) используется для объединения входящих плезиохронных и синхронных потоков в более высокоскоростной поток кадров STM-N. Добавляющий/извлекающий мультиплексор (англ. add/drop multiplexer, ADM) пропускает через себя высокоскоростной поток кадров STM-N и имеет возможность добавлять в него либо извлекать из него плезиохронные и низкоскоростные синхронные потоки. Из таких мультиплексоров можно строить надежные кольцевые сети с автоматическим переключением в случае отказа одной из линий. Цифровой кросс-коннектор (англ. digital cross-connect, DXC) — это наиболее многофункциональное оборудование SDH. Она позволяет упаковывать/распаковывать плезиохронные потоки в виртуальные контейнеры и выполнять коммутацию контейнеров разных уровней. Все перечисленные типы оборудования SDH, показанные на рисунке 5.4, поддерживают удаленное управление и мониторинг.

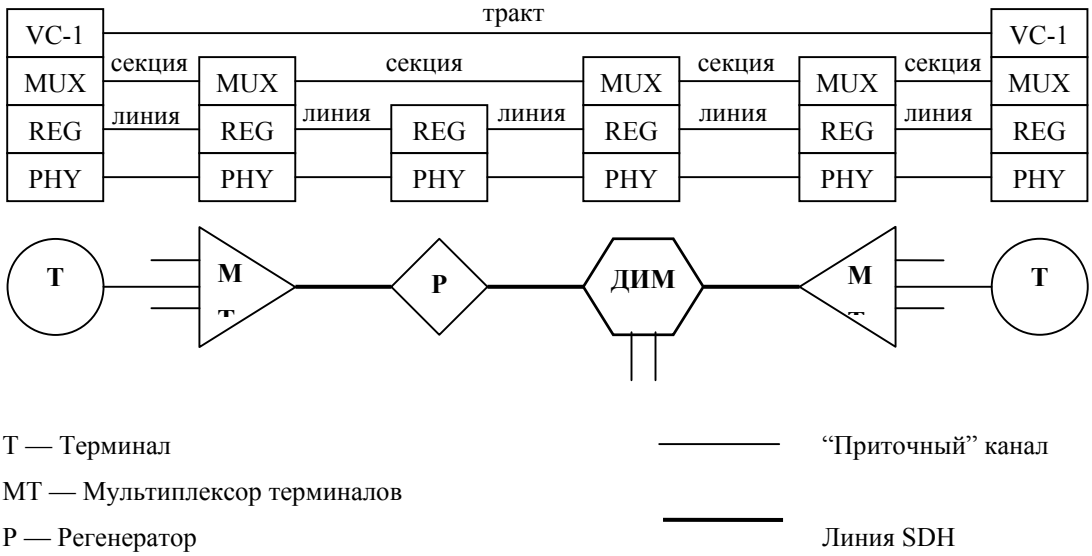

Рисунок 5.4 — Типы оборудования SDH

В качестве физической среды в SDH/SONET обычно используется одномодовое оптоволокно и лазерные приемопередатчики на длинах волн 1310 нм и 1550 нм. Пользователи сети работают с электрическими сигналами, а по сети физически передаются оптические. Для повышения стабильности лазерных передатчиков желательно, чтобы в потоке передаваемых битов было одинаковое количество нулей и единиц, поэтому в процессе преобразования электрических сигналов в оптические выполняется скремблирование.

Кадр уровня **STM-1** (англ. Synchronous Transport Module, модуль синхронной передачи) имеет длину 2430 байт, время его передачи составляет 125 мс, что соответствует скорости 155,52 Мбит/с. Кадр состоит из 9 строк по 270 байт, первые 9 байт в каждом ряду содержат служебные данные (см. рисунок 5.5 и таблицу 5.2), остальные байты каждой строки (с 10 по 270) переносят полезные данные. Каждый байт полезных данных соответствует потоку 64 Кбит/с. Кадр передается построчно, начиная с верхнего левого угла и заканчивая правым нижним углом. Передача каждого байта начинается со старшего бита. Скорость передачи полезных данных составляет 150,34 Мбит/с,

соответственно, в кадрах STM-1 может передаваться любой из плезиохронных потоков до E4 (140 Мбит/с) включительно.

байты	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10……270**
1	A1	A1	A1	A2	A2	A2	J0			
2	B1			E1			E1			
3	D1			D2			D3			
4	AU pointer									
5	B2	B2	B2	K1			K2			Данные
6	D4			D5			D6			
7	D7			D8			D9			
8	D10			D11			D12			
9	S1					M1	E2			

Рисунок 5.5 — Структура кадра STM-1

Таблица 5.2 — Служебные байты кадра STM-1 и их функции

Служебный байт	**Функция**
A1, A2	Начало (выравнивание) кадра
B1, B2	Байты контроля данных по четности
D1,…,D12	Управление сетью Q_{ECC}
E1, E2	Голосовое соединение
F1	Обслуживание
J0	Идентификатор трассировки
K1, K2	Управление автоматическим защитным переключением
S1	Индикатор качества синхронизации
M1	Подтверждение ошибки передачи

К каждому виртуальному контейнеру прикрепляется **заголовок тракта** (англ. Path OverHead, POH), предназначенный для мониторинга качества передачи и указания типа контейнера. Формат заголовка тракта различен для разных контейнеров (см. рисунки 5.6, 5.7).

Байт	Функция
J1	Индикация пути
B3	Мониторинг качества
C2	Формат контейнера
G1	Подтверждение ошибки передачи
F2	Обслуживание
H4	Индикация суперкадра
F3	Обслуживание
K3	Автоматическое защитное переключение
N1	Совместный мониторинг соединения

Рисунок 5.6 — VC-3/4 POH

Байт	Функция
V5	Индикация и мониторинг ошибок
J2	Индикация пути
N2	Совместный мониторинг соединения
K4	Автоматическое защитное переключение

Рисунок 5.7 — VC-11/12 POH

Синхронные сети предназначены для передачи как потоков плезиохронной иерархии и современных ATM-сетей, так и непосредственно потоков IP-дейтаграмм.

Уровни иерархии SDH/SONET перечислены в таблице 5.3.

Таблица 5.3 — Уровни SDH/SONET

Уровень SONET	**Уровень SDH**	**Скорость передачи, Мбит/с**
STS-1		51,38
STS-3	STM-1	155,52
STS-9	STM-3	466,56
STS-12	STM-4	622,08
STS-18	STM-6	933,12
STS-24	STM-8	1244,16
STS-36	STM-12	1866,24
STS-48	STM-16	2488,32
STS-96	STM-32	
STS-192	STM-64	

5.5 Вопросы для самопроверки

- С чем связана низкая пропускная способность сетей X.25?
- За счет чего сети Frame Relay могут гарантировать среднюю скорость передачи данных?
- Какие поля заголовка кадра Frame Relay позволяют уведомлять о перегрузках в сети? Как они обрабатываются?
- Почему базовый канал PDH (DS-0) имеет пропускную способность 64 Кбит/с?
- Почему в каналах PDH используется кодирование каждого отсчета 8 битами, в то время, как известно, что для приемлемого представления человеческой речи требуется 12 бит на отчет?
- С чем связана сложность демультиплексирования плезиохронных потоков?
- Какие функции выполняет добавляющий/извлекающий мультиплексор SDH?
- Какие потоки данных могут переноситься в виртуальных контейнерах SDH?

6 Технологии ISDN И ATM

6.1 Технология ISDN

Термин **Цифровая сеть интегрального обслуживания** (ЦСИО, англ. Integrated Services Digital Network, ISDN) описывает организацию цифровых каналов передачи данных на основе существующей телефонной сети. Естественно, магистральные каналы и каналы связи между АТС уже давно, как правило, цифровые, так что речь идет о так называемой «последней миле» — линии связи абонента с АТС. Подобная организация позволяет, сохранив возможность традиционного (голосового) использования телефонной линии, предоставлять абоненту разнообразные услуги передачи данных. Одной из целей разработки технологии ISDN было предоставление абоненту стандартного интерфейса, с помощью которого можно запрашивать у сети разные услуги. В основе ISDN лежит глобальная коммутация виртуальных каналов, основное устройство, образующее ISDN-сеть — это коммутатор ISDN.

ISDN предоставляет абонентам услуги выделенных каналов, коммутируемых каналов, коммутации пакетов и кадров.

6.1.1 Интерфейсы ISDN

Абонент получает услуги ISDN на **терминальном оборудовании** (англ. Terminal Equipment, TE): компьютере, телефонном аппарате, факсимильном аппарате, мини-АТС, маршрутизаторе и т.п.

Непосредственно к каналу связи с коммутатором подключается устройство, называемое **сетевым окончанием** (англ. Network Termination, NT). Сетевые окончания бывают двух видов: NT-1 и NT-2. Основная функция устройства NT-1 — преобразование двухпроводного U-интерфейса (обычной телефонной пары, соединенной с коммутатором) в четырехпроводный T-интерфейс, к которому может подключаться терминальное оборудование. Непосредственно к T-интерфейсу подключают оборудование, предназначенное

для индивидуального (монопольного) пользования каналом связи (например, если канал будет использован только для передачи данных, компьютерный ISDN-адаптер может подключаться напрямую к T-интерфейсу). В большинстве случаев общим каналом будут пользоваться несколько терминалов. Для их мультиплексирования предназначены устройства NT-2, подключаемые к T-интерфейсу, и предоставляющие некоторое количество (обычно до 8) S-интерфейсов, к которым можно подключать ISDN-терминалы.

Терминальное оборудование, которое может подключаться к S-интерфейсу, называется TE-1. Прочие устройства (аналоговые телефоны, факсы, модемы) называются TE-2 и могут подключаться к S-интерфейсу через дополнительное устройство — **терминальный адаптер** (англ. Terminal Adapter, TA).

Пользовательский интерфейс ISDN базируется на каналах трех типов: B-, D- и H-каналах. B-каналы (64 Кбит/с, ИКМ) обеспечивают дуплексную передачу пользовательских данных. D-каналы (16 или 64 Кбит/с) передает адресную информацию для работы коммутаторов, а также выполняет некоторые другие сервисные функции. H-каналы объединяют несколько B-каналов для высокоскоростной передачи данных: H0=384 Кбит/с (6 B-каналов), H10=1472 Кбит/с (23 B-канала), H11=1536 Кбит/с (24 B-канала), H12=1920 Кбит/с (30 B-каналов).

Абоненту ISDN может быть предоставлен один из двух типов интерфейса: базовый (англ. Basic Rate Interface, BRI) и первичный (англ. Primary Rate Interface, PRI).

Базовый интерфейс BRI состоит из двух B-каналов и одного D-канала (16 Кбит/с). Предназначается для индивидуальных пользователей. Суммарная скорость передачи пользовательских данных по BRI составляет 144 Кбит/с в каждом направлении.

Данные передаются кадрами, несущими по 48 бит данных: по 16 бит для каждого B-канала и 4 бита для D-канала (см. рисунок 6.1). Кадр содержит также биты синхронизации. Передача одного кадра занимает 250 мс.

Начало кадра	Канал B1	Канал D	Канал B2
	16 бит	4 бит	16 бит

Рисунок 6.1 — Кадр базового интерфейса BRI ISDN

Первичный интерфейс PRI предназначен для пользователей с потребностями в высокой пропускной способности. В США PRI состоит из 23 B-каналов и одного D-канала (64 Кбит/с), что в сумме дает скорость передачи 1536 Кбит/с. В Европе и России PRI состоит из 30 B-каналов и одного D-канала (64 Кбит/с), что в сумме дает скорость передачи 1984 Кбит/с.

PRI может быть основан и на H-каналах, но общая пропускная способность не должна превышать 2048 Кбит/с для Европы или 1544 Кбит/с для США (это связано со скоростями наиболее распространенных цифровых каналов в разных регионах).

B-каналы образуют сеть с коммутацией цифровых каналов, для них определен только протокол физического уровня (I.430/431). Коммутация составного B-канала происходит при прохождении пакетов по D-каналу.

D-каналы образуют сеть с коммутацией пакетов. Для этой сети определены протоколы трех уровней: физического (I.430/431), канального (LAP-D, Q.921), сетевого (Q.931). Кроме передачи запросов на коммутацию B-каналов, D-канал может переносить пакеты X.25 (например, для связи двух X.25-сетей через ISDN-сеть).

6.2 Технология ATM

6.2.1 Основные принципы технологии ATM

Технология **Асинхронного режима доставки** (АРД, англ. Asynchronous Transfer Mode, ATM) разрабатывалась как единый универсальный транспорт для передачи разнородного трафика по одним и тем же линиям связи. Основная идея ATM заключена в преодолении главного недостатка мультиплексирования с разделением времени (TDM) — невозможности перераспределения

пропускной способности объединенного канала между активными подканалами во время простоя остальных подканалов. ATM совмещает подходы коммутации пакетов (передача данных в виде индивидуально адресуемых пакетов) и коммутации каналов (использование пакетов небольшого фиксированного размера, уменьшающих задержки в сети). Информация передается в **ячейках** (англ. cell) фиксированного размера в 53 байта, из которых 5 байт отведено под заголовок, а 48 байт — под пользовательские данные. Основное оборудование в сетях ATM — коммутаторы, соединенные цифровыми линиями связи. Малый размер ячеек позволяет передавать чувствительный к задержкам трафик (например, голос), а фиксированный формат ячеек позволяет обрабатывать их аппаратно с высокой скоростью.

В сети ATM все физические соединения выполняются по принципу «точка-точка». Определено два основных интерфейса, которые должны поддерживать коммутаторы ATM: **интерфейс пользователь-сеть** (англ. User-to-Network Interface, UNI), используемый для соединения абонентов с коммутаторами, и **интерфейс сеть-сеть** (англ. Network-to-Network Interface, NNI), предназначенный для соединения коммутаторов. Ячейки, передаваемые через UNI и NNI имеют практически одинаковый (отличающийся в одном поле) заголовок.

Заголовок ячейки ATM состоит из пяти байт, его формат приведен на рисунке 6.2. Непосредственно за заголовком следует 48 байт пользовательских данных.

	7 бит	**6 бит**	**5 бит**	**4 бит**	**3 бит**	**2 бит**	**1 бит**	**0 бит**
1 байт	GFC/VPI				VPI			
2 байт	VPI				VCI			
3 байт	VCI							
4 байт	VCI				PT			CLP
5 байт	HEC							

Рисунок 7.2 — Заголовок ячейки ATM

Поле CFG (англ. Generic Flow Control) — **общее управление потоком**, существует только в UNI и, как правило, не используется. В NNI биты, занимаемые этим полем, передаются полю VPI.

Поле VPI (англ. Virtual Path Identifier), **идентификатор виртуального пути**. Виртуальный путь может объединять виртуальные каналы, проложенные по одному маршруту через сеть, или каналы, имеющие общую часть маршрута.

Поле VCI (англ. Virtual Channel Identifier), **идентификатор виртуального канала**, назначаемый соединению при его установлении. Все ячейки, передаваемые через это соединение, имеют одинаковый VCI. Поля VCI и VPI позволяют определить следующую точку назначения ячейки — следующий коммутатор на маршруте ее передачи. Каждый коммутатор назначает этим полям новые значения, так что содержимое полей VPI и VCI имеют смысл только для одной конкретной линии связи, а не для всей сети.

Поле PT (англ. Payload Type), **тип информации**, позволяет различать пользовательские и служебные ячейки. Первый бит этого поля в пользовательских ячейках равен 0, а в служебных — 1. В пользовательских ячейках может быть выставлен в 1 второй бит поля PT, что сигнализирует о перегрузке сети (этот бит называется **явным указателем перегрузки в прямом направлении** (англ. Explicit Forward Congestion Identifier, EFCI). Третий бит этого поля может использоваться в кадрах более высокого уровня.

Поле CLP (англ. Congestion Loss Priority), **приоритет потерь при перегрузках**, выставляется в 1 для указания низкоприоритетных ячеек, которые можно аннулировать при перегрузке сети.

Поле HEC (англ. Header Error Control), **контрольная последовательность для заголовка**, содержит контрольную сумму, вычисленную с помощью корректирующих кодов Хемминга. Помимо выявления многократных ошибок, позволяет исправлять все однократные и некоторые двойные ошибки. Это поле играет важную роль при передаче ячеек по сетям SONET/SDH. Поле данных кадра STS-n (STM-n) не содержит указаний на границы ячеек ATM. Поэтому коммутатор ATM вычисляет

контрольную сумму для предполагаемых заголовков ячеек, пока она не совпадет со значение поля HEC.

Коммутаторы ATM могут работать в двух режимах, различающихся использованием значений полей VCI и VPI: режиме коммутации виртуального пути и режиме коммутации виртуального канала. Первый режим игнорирует поле VCI и выполняет передачу ячеек только на основе поля VPI. Так работают магистральные коммутаторы, коммутирующие группы виртуальных каналов как единое целое — виртуальный путь. Коммутаторы локальных сетей обычно работают во втором режиме — режиме коммутации виртуальных каналов, игнорируя поле VPI и анализируя только поле VCI.

ATM предоставляет три типа сервиса:

- **Постоянные виртуальные цепи** (англ. Permanent Virtual Circuits, PVC) функционируют аналогично выделенным линиям, обеспечивая прямую связь между узлами. В этом случае соединение устанавливается вручную (при создании цепи) и не требует выполнения дополнительных процедур перед передачей данных.
- **Коммутируемые виртуальные цепи** (англ. Switched Virtual Circuits, SVC) функционируют аналогично коммутируемым телефонным линиям — соединение устанавливается только на время передачи данных, для чего необходимо затратить время и трафик на выполнение специальных процедур установления соединения.
- **Сервис без установления соединения** (англ. connectionless service) представляет собой традиционный для сетей с коммутацией пакетов дейтаграммный сервис.

Для различных видов информации, передаваемой через сети ATM, определены пять классов сервиса (или классов трафика):

- Класс A используется для передачи данных с постоянной битовой скоростью (англ. Constant Bit Rate, CBR) и изохронностью, позволяет передавать голос, видеоизображение (в том числе телевизионное).

- Класс B используется для передачи данных с переменной битовой скоростью (англ. Variable Bit Rate, VBR) и изохронностью, позволяет передавать компрессированный голос и видеоизображение, например, для видеоконференций.
- Класс C используется для передачи данных с переменной битовой скоростью без требования изохронности с установлением соединения, позволяет передавать трафик компьютерных сетей, работающих по протоколам с установлением соединения — TCP, X.25 и т.п.
- Класс D используется для передачи данных с переменной битовой скоростью без требования изохронности, без установления соединения, позволяет передавать трафик компьютерных сетей, работающих по протоколам без установления соединения — IP (UDP), Ethernet и т.п.
- Класс X не имеет стандартного описания и определяется параметрами трафика и качества обслуживания, оговоренными в контракте на предоставление услуг передачи данных.

6.2.2 Стек протоколов ATM

Стек протоколов соответствует нижним уровням модели ВОС и включает три уровня:

- Физический уровень (англ. Physical layer) определяет способы передачи в зависимости от среды. Физический уровень делится на два подуровня:
    - Подуровень, зависимый от среды передачи (англ. Physical Medium-Dependent, PMD), определяет физическую среду, включая типы кабелей и разъемов.
    - Подуровень конвергенции передачи (англ. Transmission-Convergence) определяет границы ячеек в потоке бит, генерирует

и проверяет поле контрольной суммы HEC, упаковывает ячейки в кадры, согласует скорость передачи ячеек.

- Уровень ATM (англ. ATM Layer) отвечает за передачу ячеек через сеть.
- Уровень адаптации ATM (англ. ATM Adaptation Layer, AAL) обеспечивает изоляцию верхних протокольных уровней от деталей формирования и передачи ATM-ячеек.

Архитектурная модель ATM обычно содержит, кроме того, еще и высшие уровни, расположенные над AAL.

На все уровни распространяется три плана (плоскости, plane):

- Управление (англ. Control) — генерация и обслуживание запросов сигнализации;
- Пользовательский (англ. User) — обслуживание передачи данных;
- Менеджмент (англ. Management) — управление функциями, специфическими для разных уровней (например, обнаружение отказов), и управление остальными планами.

6.2.3 Уровень адаптации AAL

Уровень адаптации содержит **подуровень сборки и сегментации** (англ. Segmentation And Reassembly, SAR) и **подуровень конвергенции** (англ. Convergence Sublayer, CS).

Нижний подуровень SAR не зависит от класса передаваемого трафика и предназначен для взаимного преобразования сообщений, принимаемых от верхних уровней в ячейки и обратно.

Подуровень CS зависит от класса передаваемого трафика, причем первоначально каждому классу соответствовал свой уровень адаптации — от AAL1 до AAL4, но с развитием стандарта от уровня AAL2 отказались, уровни AAL3 и AAL4 объединили в AAL3/4, а на основе AAL4 был разработан его

упрощенный вариант — AAL5. Таким образом, в настоящее время используется три уровня адаптации: AAL1, AAL3/4 и AAL5.

Протокол AAL1 предназначен для передачи данных с постоянной скоростью (класс A). AAL1 требует синхронизации между источником и приемником данных, которую должна предоставлять линия связи (например, SONET/SDH). Заголовок AAL1 состоит из одного или двух байт — номера ячейки SN (англ. Sequence Number) и, возможно, контрольного значения SNP (англ. Service Number Protection), служащего для контроля ошибок в поле SN. Протокол AAL1 допускает потерю ячеек, но, за счет синхронизации и нумерации ячеек, минимизирует ухудшение качества.

Протокол AAL3/4 обслуживает классы C и D, для которых допустима переменная скорость передачи и характерен пульсирующий трафик. Протокол AAL3/4 старается не допустить потери ячеек, для чего ячейки могут задерживаться и буферизоваться коммутаторами. При формировании ячеек из данных верхних уровней, протокол AAL3/4, подобно AAL1, нумерует ячейки, и дополнительно снабжает каждую ячейку контрольной суммой (CRC-10). При искажениях или потере ячеек не происходит их восстановления или повторного запроса — AAL только сигнализирует верхнему уровню о произошедшей ошибке.

Протокол AAL5 работает подобно AAL4, за исключением того, что контрольной суммой снабжается не каждая ячейка, а все сообщение (общая контрольная сумма передается в последней ячейке). Для указания последней ячейки сообщения используется третий бит поля PT заголовка ячейки: у последней ячейки он устанавливается в 1. Это уменьшает избыточность, но делает невозможным мультиплексирование ячеек разных сообщений: ячейки одного сообщения должны следовать одна за другой, не чередуясь с другими ячейками.

6.3 Вопросы для самопроверки

- Как можно подключить несколько терминалов к одному Т-интерфейсу ISDN?
- Что понимается под «интегральным обслуживанием» в термине «Цифровая сеть интегрального обслуживания»?
- Почему в АТМ выбрана такая маленькая единица передачи данных – всего 53 байта?
- Чем отличаются постоянные и коммутируемые виртуальные цепи?
- Позволяет ли протокол уровня АТМ повторно передавать или восстанавливать ошибочные ячейки?
- Позволяют ли протоколы уровня адаптации AAL повторно передавать или восстанавливать ошибочные данные?
- В каких случаях имеет смысл применять протокол AAL1?
- Чем отличаются протоколы AAL5 и AAL3/4? Как их отличия влияют на их области применения?

7 Сетевые операционные системы

Сетевая операционная система — операционная система, обеспечивающая обработку, хранение и передачу данных в информационной сети. Сетевая операционная система определяет взаимосвязанную группу протоколов верхних уровней, обеспечивающих основные функции сети: адресацию объектов, функционирование служб, обеспечение безопасности данных, управление сетью.

В большинство современных СОС сетевые функции встроены, хотя существуют и специализированные комплекты программ — сетевые оболочки — добавляющие сетевые функции к локальной ОС. Также традиционно различают одноранговые (англ. peer-to-peer), пользовательские и серверные СОС, хотя фактически все используемые СОС обладают функциями как клиента, так и сервера. Так, например, пользовательская СОС Microsoft Windows XP Professional может играть роль сервера, предоставляя доступ к своим файлам, а серверная СОС Microsoft Windows Server 2003 может играть роль клиента, обращаясь к этим файлам. Далее будем различать две роли СОС: пользовательскую и серверную, понимая, что они могут совмещаться в единой программной системе.

К основным функциям СОС относят управление локальными ресурсами и организацию сетевой работы. Дополнительные аспекты функционирования СОС включают защиту от несанкционированного доступа и обеспечение отказоустойчивости. К управляемым СОС локальным ресурсам относятся дисковые файлы, принтеры, модемы и прочие ресурсы. К организации сетевой работы относят адресацию, буферизацию, маршрутизацию, управление потоками данных и др.

Пользовательская СОС должна обеспечивать для локально выполняющихся программ прозрачный доступ к удаленным файлам, размещенным на других узлах. Соответствующий компонент СОС называется редиректором (англ. redirector — перенаправитель). Прозрачность состоит в

том, что для локальной программы обращение к удаленным файлам выглядит так же, как к локальным. Например, файл D:\file.dat может быть размещен на локальном диске D:, а файл G:\file.dat — на диске другого узла. Основные функции редиректора: перехват обращений к удаленным файлам, формирование сетевых пакетов с запросами на чтение/запись данных и передача их на соответствующий узел, прием сетевых пакетов с содержимым удаленных файлов и передача их запросившей локальной программе.

Серверная СОС должна обеспечивать доступ к локальным файлам для программ, выполняющихся на других узлах сети. Соответствующий компонент СОС называется сервером. Основные функции сервера: получение от редиректоров пользовательских СОС других узлов пакетов с запросами на чтение/запись данных, обращение к соответствующим локальным файлам, формирование ответных пакетов с их содержимым и передача их по сети.

Средства защиты от несанкционированного доступа в СОС позволяют определять права доступа к объектам — например, для файлов могут быть определены права чтения, изменения, создания, запуска на выполнение и др., — и назначать конкретные комбинации прав доступа конкретным пользователям или их группам. Помимо прав доступа к объектам могут определяться системные привилегии, например, право изменения времени на сервер, резервного копирования, останова сервера и т.д. Для предоставления пользователю доступа к объектам на основе установленных прав (авторизации) необходима предварительная аутентификация пользователя, т.е. проверка, что он является тем, за кого себя выдает. Среди основных методов сетевой аутентификации: предъявление пароля, выполнение процедуры шифрования с использованием пары публичного/приватного ключей, предъявление смарт-карты, и др.

Отказоустойчивость характеризуется сохранением работоспособности системы при воздействии дестабилизирующих факторов. Отказоустойчивость обеспечивается применением автономных источников питания, зеркалированием и дублированием дисков и серверов.

В настоящее время наибольшее распространение получили две основные группы СОС: Unix (Sun Solaris, IBM AIX, HP-UX, *BSD, Linux и др.) и Microsoft Windows (Windows 98/ME/2000 Professional/XP получили наибольшее распространение в качестве ОС настольных систем, а Windows NT Server 4.0, Windows 2000 Server, Windows Server 2003 — в качестве серверных ОС).

8 Технологии распределенных вычислений

Программы, работающие в сети и совместно решающие ту или иную задачу, часто бывает удобно считать частями одного приложения. Такое приложение называют распределенным. Распределенными могут быть как прикладные, так и системные программы. Распределенные программы классифицируют по следующим критериям:

- способ разделения приложения на части, выполняющиеся на разных компьютерах;
- способ взаимодействия между частями приложения;
- способ организации специализированных сетевых серверов, выполняющих общие для приложений функции.

Приложения можно делить на части самыми разнообразными способами. Тем не менее, существуют и типовые модели, например, шестиуровневая модель структуры распределенного приложения, состоящая из следующих уровней:

- средства представления данных (пользовательский интерфейс);
- логика представления данных;
- прикладная логика;
- логика данных;
- внутренние операции базы данных;
- файловые операции.

8.1 Удаленный вызов процедур

Удаленный вызов процедур (англ. remote procedure call, RPC) — это технология взаимодействия прикладных программ, выполняющихся на разных узлах, разработанная корпорацией Sun Microsystems (см. RFC 1831). RPC дает возможность программам вызывать процедуры, расположенные на других

узлах, (или в других адресных пространствах) точно так же, как и локальные процедуры.

В случае локального вызова процедуры, вызывающая сторона размещает передаваемые параметры в некотором известном месте (например, в регистрах или стеке) а затем передает управление этой процедуре. После того, как процедура выполнена, вызывающая сторона вновь получает управление, извлекает возвращаемые параметры из регистров или стека и продолжает свою работу.

Удаленный вызов процедуры для вызывающей стороны выглядит очень похоже. Поток управления проходит через два процесса: вызывающий (клиентский) и обслуживающий (серверный). Клиентский процесс отправляет сообщение-запрос серверному и приостанавливает свою работу до получения сообщения-ответа. Сообщение-запрос содержит передаваемые параметры, а сообщение-ответ — возвращаемые параметры. После того, как сообщение-ответ получено, клиентский процесс извлекает из него данные и продолжает свою работу. Серверный процесс в основном ожидает поступления сообщений-запросов. Когда запрос поступил, из него извлекаются значения параметров, выполняется требуемая процедура, формируется и отправляется сообщение-ответ, и серверный процесс опять переходит в состояние ожидания.

В этой модели клиентский и серверный процессы выполняются синхронно — в каждый момент времени выполняется только один из них. Тем не менее, расширения протокола RPC (например, пакет ONC+ фирмы Sun Microsystems) допускают и асинхронный режим, когда клиентский процесс может выполнять некоторую полезную работу во время ожидания ответа.

Наряду со значительным подобием, имеется ряд серьезных отличий между удаленным и локальным вызовом процедур. Во-первых, обработка ошибок: необходимо обрабатывать отказы сети и удаленного сервера. Во-вторых, глобальные переменные и побочные эффекты: поскольку сервер не имеет доступа к адресному пространству клиента, нельзя передавать в процедуру данные через глобальные переменные, а также возвращать данные

путем побочных эффектов. В-третьих, производительность: удаленные вызовы процедур выполняются на несколько порядков дольше, чем локальные. В-четвертых, контроль доступа: поскольку удаленные вызовы могут производиться через небезопасные сети, серверу необходимо проверять аутентичность клиента, и наоборот.

Протокол RPC определяет только структуру сообщений и может быть реализован поверх любого транспортного протокола. Надежность RPC зависит от надежности транспортного протокола: при работе через TCP выполняется управление потоком, квитирование и повторная передача потерянных пакетов, при работе через UDP, программа, пользующаяся RPC, должна самостоятельно обеспечивать надежность. При этом возникает проблема единственного выполнения удаленной процедуры: если по тем или иным причинам от сервера не пришло сообщение-ответ, то, по истечении таймаута, клиент должен повторно передать сообщение-запрос. Возможна ситуация, при которой сервер получит и выполнит оба запроса, что является нарушением логики работы программы. Для защиты от подобных проблем может использоваться, например, механизм нумерации запросов: разные запросы несут в себе разные номера, а повторные запросы — номера исходных запросов. Сервер должен хранить список выполненных номеров запросов и не выполнять запросы повторно.

Спецификация RPC не определяет конкретную процедуру привязки (англ. binding) клиентов к серверам. Возможным способам привязки посвящен отдельный документ — RFC 1833. Конкретная служба (service) RPC идентифицируется номером программы RPC, номером ее версии и транспортным адресом, по которому к ней можно обратиться. Транспортный адрес состоит из сетевого адреса (например, IP-адреса) и селектора транспорта (например, номера TCP-порта). Для успешного обращения к службе, клиент должен знать все перечисленные идентификаторы. При этом номер программы RPC и номер ее версии обычно жестко прописываются в программе клиента, а сетевой адрес либо передается программе при запуске, либо определяется

динамически путем обращения к службе имен (например, DNS). Селектор транспорта обычно определяется динамически при запуске экземпляра сервера. Сервер динамически выделяет транспортный адрес и сообщает его некоторой известной (расположенной по заранее заданному селектору транспорта) службе поиска (англ. lookup service). Клиенты, когда им нужно определить транспортный адрес той или иной службы, обращаются к службе поиска.

Необходимость в использовании отдельной службы поиска связана с тем, что многие транспортные протоколы не могут предоставить большое количество заранее известных селекторов, в то время, как потенциальное количество служб очень велико. Использование службы поиска позволяет занять единственный заранее известный селектор, а селекторы остальных служб назначать динамически.

RFC 1833 определяет три типа служб поиска, все они используют общий номер программы RPC (1000000) и порты TCP/111 и UDP/111.

Стандарты распределенных вычислений Open Source Foundation (OSF) Distributed Computing Environment (DCE) используют RPC как основной механизм выполнения удаленных процедур. В пользу RPC говорят его простота, прозрачность и производительность. То, что модель вызова RPC максимально приближена к модели вызова локальных процедур, позволяет сократить время обучения разработчиков.

8.2 Microsoft DCOM

Модель распределенных компонентных объектов DCOM (англ. Distributed Component Object Model) — объектно-ориентированная технология разработки распределенных программных систем.

Хотя технология COM (англ. Component Object Model) разрабатывалась с учетом будущей поддержки распределенных систем, однако ее первоначальная реализация могла работать только на одном компьютере. Объекты COM могли быть реализованы в DLL или в отдельном процессе, исполняющемся только на

том же компьютере, что и их клиент. В DCOM объекты могут предоставлять свои сервисы и клиентам, выполняющимся на других узлах сети.

Возможность запускать удаленные объекты и вызывать их методы - важное достижение, но требуется большее. В частности, нужен способ контроля за тем, кто имеет право создавать объекты на данной машине, и обеспечение безопасного доступа к этим объектам по незащищенной сети. С этой целью в основу DCOM положен набор сервисов контроля доступа Приложения (включая программы, созданные до DCOM) могут использовать DCOM и работать вполне безопасно без добавления какого-либо кода, связанного с защитой. С другой стороны, приложения, знающие о новых средствах DCOM контроля доступа, могут задействовать их явно.

Сервисы создания объектов — одни из важнейших сервисов предоставляемых COM. Клиенты обычно создают объекты, вызывая библиотеку COM или через моникеры. Эти подходы работают и в DCOM, хотя и с некоторыми новыми особенностями.

Независимо от того, где исполняется объект, клиент обычно создает его и затем получает указатели на необходимые интерфейсы. Чтобы создать удаленный объект, клиент вызывает CoCreatelnstance, передавая идентификатор класса CLSID, идентификатор интерфейса IID (эти два параметра используются и при создании локальных объектов) и имя узла, на котором он должен быть создан. Для объекта, создаваемого на том же узле, в системном реестре по CLSID находится имя файла библиотеки DLL или EXE-файла сервера данного класса объектов, после чего найденный сервер запускается. Для создания удаленного объекта устанавливается связь с удаленным узлом, в ее реестре отыскивается требуемый CLSID, и на удаленном узле запускается соответствующий процесс сервера.

DCOM предоставляет несколько вариантов идентификации узлов в зависимости от используемых сетевых протоколов: доменные имена (DNS), IP-адреса, имена NetBIOS, IPX-адреса.

После запуска удаленного объекта и получения указателей на его интерфейсы, клиент может вызывать методы этих интерфейсов. Вызов метода объекта, реализованного в удаленном сервере, сводится к вызову удаленной процедуры сервера по протоколу объектного RPC (англ. Object RPC). Хотя ORPC включает ряд новых соглашений по взаимодействию клиента с сервером, добавляет несколько новых типов данных и использует некоторые поля пакета особым образом, сама структура пакетов осталась той же, что и в оригинальном DCE RPC.

MS RPC определяет два протокола, из которых выбирается один в зависимости от используемого транспортного протокола. Протокол CN используется поверх транспортных протоколов с установлением логических соединений, например, TCP, надеясь на то, что нижележащий протокол гарантирует надежную доставку данных. Протокол DG используется поверх транспортных протоколов без установления логического соединения, например, UDP. Протокол DG самостоятельно обеспечивает надежную доставку данных.

Для выполнения вызова ORPC, клиент должен указать сетевой адрес удаленного узла (например, IP-адрес), номер порта и комбинацию протоколов (например, CL RPC и UDP).

Для минимизации риска неавторизованного создания и использования объектов, DCOM определяет стандартный способ доступа к сервисам защиты и их использования. Контроль над тем, кто имеет право запускать серверы различных классов на данном удаленном узле, в DCOM называется защитой активизации (англ. activation security). Контроль прав на вызовы клиентами методов уже исполняющихся объектов, в DCOM называется защитой вызовов (англ. call security).

Для обеспечения защиты активизации, в реестре узла имеется флажок, разрешающий или запрещающий удаленный запуск серверов на данном узле, хранятся списки пользователи, имеющие право запуска серверов, и, возможно, списки управления доступом для конкретных классов.

Интерфейсы защиты вызовов DCOM определяют 2 основные возможности: автоматическую и поинтерфейсную защиты. В первом случае клиентский или серверный процесс вызовом функции CoInitializeSecurity устанавливает уровень защиты по умолчанию для всех вызовов этого процесса или вызовов, выполняемых им. Во втором — разные установки защиты можно задать для разных интерфейсов одного объекта или для разных объектов. Можно использовать и оба варианта одновременно, установив значения по умолчанию с помощью автоматической защиты и выполнив затем тонкую настройку, используя поинтерфейсную защиту.

8.3 Технология CORBA

Общая архитектура брокера объектных запросов CORBA (англ. Common Object Request Broker Architecture) — технология, архитектура и набор стандартов для создания распределенных программных приложений на базе использования брокеров объектных запросов ORB (англ. Object Request Broker) для взаимодействия распределенных объектов.

Спецификации CORBA разработаны международным консорциумом OMG (англ. Object Management Group), основной задачей которого является разработка архитектуры и методов реализации программного обеспечения, которое в объектно-ориентированном стиле позволило бы выполнить интеграцию существующих и заново разрабатываемых (не обязательно объектно-ориентированных) информационно-вычислительных ресурсов. Эти ресурсы должны быть оформлены в виде CORBA-объектов, обменивающихся запросами и ответами с другими объектами. Для каждого объекта должен быть описан его интерфейс — набор запросов, на которые этот объект умеет отвечать. Описание интерфейса объекта на языке определения интерфейсов IDL (англ. Interface Definition Language) обрабатывается специальным компилятором, на выходе которого появляется программный код, отвечающий за взаимодействие объекта с брокером ORB. Этот программный код затем

встраивается в серванта (англ. servant — слуга) — серверную программу, реализующую объект.

Инфраструктуру CORBA составляют взаимодействующие программные процессы брокеров ORB, объектные службы (англ. object services) и общие средства (англ. common facilities).

CORBA-объекты передают запросы брокеру, который ищет соответствующего серванта, реализующего тот CORBA-объект, который должен выполнить этот запрос, затем брокеру этого серванта передается запрос. Результаты обработки запроса передаются объекту-инициатору в обратном порядке.

Объектные службы представляют собой набор услуг (интерфейсов и объектов), которые обеспечивают выполнение базовых функций, требуемых для реализации прикладных объектов и объектов категории "общие средства" (например, служба именования объектов, служба долговременного хранения объектов, служба управления транзакциями и т.д.).

Общие средства содержат набор классов и экземпляров объектов, поддерживающих функции, полезные в разных прикладных областях (например, средства поддержки пользовательского интерфейса, средства управления информацией и т.д.). Общие службы подразделяются на две категории: горизонтальные и вертикальные. Горизонтальный набор общих средств включает операции, используемые во многих системах и не зависящие от конкретных прикладных систем. Вертикальный набор служит для поддержки конкретной прикладной системы. Возможна эволюция общих средств, при которой вертикальные средства, оказавшиеся общими для нескольких прикладных систем, мигрируют в набор горизонтальных средств. В число горизонтальных общих средств входят, например, средства поддержки пользовательского интерфейса, средства управления системой, средства управления задачами и т.д. Вертикальные общие средства включают, например, средства поддержки прикладных систем для бизнеса и производства, распределенные моделирующие средства и т.д.

8.4 Вопросы для самопроверки

- В чем состоит основное отличие вызова удаленной процедуры от вызова локальной процедуры?
- Как происходит вызов удаленной процедуры при использовании RPC?
- Как клиент RPC узнает адрес сервера?
- Чем принципиально отличается вызов удаленной процедуры (например, по RPC) и обращение к методу удаленного объекта (например, по DCOM)?
- Зачем в архитектуре CORBA, помимо клиентов и серверов, нужен брокер объектных запросов?

9 Протоколы прикладного уровня

9.1 Структура и информационные услуги территориальных сетей

Территориальная (региональная) сеть — коммуникационная сеть, связывающая географически удаленные друг от друга компьютеры и локальные сети. Как правило, территориальные сети строятся на основе технологий SDH, ATM, или Frame Relay. Обычно территориальная сеть имеет иерархическую структуру и строится как совокупность каналов точка-точка.

Верхний уровень территориальной сети — федеральные узлы, связанные между собой магистральными волоконно-оптическими либо спутниковыми каналами связи. Средний уровень территориальной сети образуют региональные сети, состоящие из региональных узлов. Они связаны с федеральными узлами и, возможно, между собой выделенными высоко- или среднескоростными каналами, такими, как каналы T1/E1, B-ISDN или радиорелейные линии. Нижний уровень — местные узлы (серверы доступа), связанные с региональными узлами преимущественно коммутируемыми или выделенными телефонными каналами связи, хотя заметна тенденция к переходу к высоко- и среднескоростным каналам. Именно к местным узлам подключаются локальные сети малых и средних предприятий, а также компьютеры отдельных пользователей. Корпоративные сети крупных предприятий соединяются с региональными узлами выделенными высоко- или среднескоростными каналами.

К основным услугам территориальных сетей относят:

- услуги передачи файлов,
- услуги электронной почты,
- услуги дистанционного управления,
- услуги конференц-связи.

Для доступа к перечисленным и прочим услугам территориальных сетей разработаны прикладные протоколы, которые будут рассмотрены ниже.

9.2 Протоколы файлового обмена

9.2.1 Протокол передачи файлов FTP

Протокол передачи файлов FTP (англ. File Transfer Protocol) описан в RFC 959 и предназначен для передачи текстовых и двоичных файлов между узлами. Протокол FTP использует два TCP-соединения: стандартный порт 21 служит для управляющего соединения, по которому передаются запросы, а порт 20 — для соединения, по которому передается содержимое файлов. Набор команд, которыми обмениваются клиент с сервером по управляющему соединению, отличается от набора команд, доступного пользователю.

Основные пользовательские команды FTP:

- open <имя узла> — установить соединение с сервером,
- close — закрыть соединение с сервером, остаться в командном режиме,
- quit — выйти из программы,
- help — вывести список команд,
- cd [<имя каталога на сервере>] — перейти в указанный каталог на сервере,
- lcd [<имя локального каталога>] — перейти в указанный каталог,
- dir [<имя каталога на сервере>] — вывести список файлов в каталоге на сервере,
- get <имя файла на сервере> [<локальное имя файла>] — получить с сервера файл по имени,
- mget <шаблон имен файлов> — получить с сервера несколько файлов по шаблону,
- put <имя локального файла> [<имя файла на сервере>] — передать на сервер локальный файл,
- mput <шаблон имен локальных файлов> — передать на сервер несколько локальных файлов по шаблону.

При установлении соединения FTP-сервер выдает приветствие и запрашивает поочередно имя пользователя и пароль. Публичные (“анонимные”) FTP-серверы допускают подключение под именем анонимного пользователя “anonymous” или “ftp” и паролем, совпадающим с адресом электронной почты пользователя.

9.2.2 Простейший протокол передачи файлов TFTP

Простейший протокол передачи файлов TFTP (англ. Trivial File Transfer Protocol) предназначен для высокоскоростной передачи файлов в надежных защищенных сетях. Протокол TFTP передает файлы без проверки полномочий пользователя. Основное достоинство TFTP — небольшой объем программной реализации, что очень важно для встраиваемых систем, систем бездисковой загрузки, операционных системах сетевых устройств, таких, как маршрутизаторы. Во всех этих случаях программа начальной загрузки должна быть записана в ПЗУ, объем которого ограничен.

Для доставки блоков данных в TFTP используется протокол UDP. Блоки имеют фиксированный размер 512 байт. После отправки блока отправитель ожидает подтверждения и, если по истечении заданного интервала времени подтверждение не получено, передает блок повторно. В первом блоке передается имя файла и указатель режима передачи — чтение (от сервера к клиенту) или запись (от клиента к серверу). Получив первый блок, сервер запоминает IP-адрес и порт, с которого он поступил, и ассоциирует последующие блоки, поступающие с этого адреса/порта с указанным именем файла. В последующих блоках передается содержимое файла, блоки нумеруются, начиная с 1, двухбайтовыми целыми числами. В подтверждении передает номер последнего принятого блока. Если последний принятый блок содержит менее 512 байт, это означает, что данный блок — последний в файле.

9.3 Протоколы электронной почты

9.3.1 Простой протокол передачи почты SMTP

Простой протокол передачи почты SMTP (англ. Simple Mail Transfer Protocol) описан в RFC 821 и RFC 822 и предназначен для отправки почты на почтовый сервер. Поступающая по протоколу SMTP на сервер почта раскладывается по личным почтовым ящикам пользователей и хранится до тех пор, пока пользователь ее не получит. В качестве почтовых ящиков могут использоваться либо отдельные файлы, либо специализированная база данных.

Передача почты по протоколу SMTP начинается с установления клиентом TCP-соединения с сервером (стандартный порт — 25). По установленному соединению клиент и сервер обмениваются текстовыми строками: клиент передает запросы, а сервер — ответы. Каждый запрос и ответ завершается переводом строки. Каждый ответ сервера начинается с трехзначного числа, первая цифра которого указывает на успешность выполнения последнего запроса: 2 — успешное выполнение, 3 — ожидается продолжение запроса, 5 — ошибка.

В таблице 10.1 приведена типовая последовательность обмена запросами и ответами во время сеанса связи по протоколу SMTP.

Таблица 10.1 — Пример SMTP-сеанса

№	Клиент	Сервер
1		220 Welcome!
2	HELO mail.server.ru	
3		250 mail.server.ru
4	MAIL FROM: abc@def.ru	
5		250 OK
6	RCPT TO: xyz@server.ru	
7		250 OK
8	DATA	
9		354 Start mail input; end with <CRLF>.<CRLF>
10	<текст сообщения>	
11	.	
12		250 OK
13	QUIT	
14		221 CLOSED

9.3.2 Протокол почтовой службы POP

Протокол почтовой службы POP (Post Office Protocol) предназначен для получения электронной почты из почтового ящика, хранящегося на почтовом сервере. Сейчас используется третья версия этого протокола, POP3, описанная в RFC 1939.

Передача почты по протоколу POP начинается с установления клиентом TCP-соединения с сервером (стандартный порт — 110). По установленному соединению клиент и сервер обмениваются текстовыми строками: клиент передает запросы, а сервер — ответы. Каждый запрос и ответ завершается переводом строки. Каждый ответ сервера начинается либо со строки “+OK”, которая указывает на успешность выполнения последнего запроса, либо со строки “-ERR”, указывающей на ошибку.

В таблице 10.2 приведена типовая последовательность обмена запросами и ответами во время сеанса связи по протоколу POP3.

Таблица 10.2 — Пример POP3-сеанса

№	Клиент	Сервер	Комментарий
1		+OK POP3 server ready	
2	USER xyz		Имя пользователя
3		+OK	
4	PASS mypassword		Пароль
5		+OK	
6	STAT		Запрос количества сообщений
7		+OK 10 2856	Первое число — количество сообщений в ящике, второе — суммарный объем сообщений в байтах
8	RETR 1		Получить первое сообщение
9		+OK 250 octects Received: from mail.def.ru (196.11.22.33) by mail.server.ru with SMTP; 01 Jan 2004 10:12:20 GMT From: abc@def.ru To: xyz@server.ru Date: 01 Jan 2004 10:12:14 GMT <текст сообщения> .	
10	DELE 1		Удалить первое сообщение
11		+OK message 1 deleted	
12	QUIT		Закончить сеанс
13		+OK	

9.3.3 Протокол доступа к Интернет-сообщениям IMAP

Протокол доступа к Интернет-сообщениям IMAP (Internet Message Access Protocol) предназначен как для получения электронной почты из почтового ящика, так и для работы с сообщениями непосредственно на почтовом сервере. Протокол IMAP версии 4.1 описан в RFC 2060 и использует TCP-соединение по стандартному порту 143. При использовании протокола IMAP основным местом хранения сообщений пользователя считается почтовый сервер, а не

клиент, и пользователь заходит на сервер, чтобы читать, сортировать и удалять находящиеся на нем сообщения.

Набор команд, входящих в протокол IMAP, позволяет создавать в почтовом ящике на сервере отдельные папки и раскладывать в них пришедшие сообщения. При использовании IMAP для каждого сообщения, находящегося в почтовом ящике, почтовый сервер сохраняет различную информацию, например, о том, читал ли данное письмо пользователь, был ли написан ответ на него или, наоборот, письмо создается пользователем и подлежит отправке. Имеется специальная команда, позволяющая искать в почтовом ящике сообщения, удовлетворяющие заданным критериям.

Хранящиеся на IMAP-сервере сообщения, могут быть помечены следующими флагами: отвечено, выделено, удалено, прочитано, черновик, новое.

Взаимодействие почтового клиента и IMAP-сервера осуществляется так же, как и работа с POP3-сервером — путем обмена текстовыми строками с запросами и ответами. Интересная особенность протокола IMAP заключается в том, что клиент не обязан дожидаться ответа от сервера, прежде чем отправить следующий запрос. Эта возможность позволяет запускать трудоемкие задачи на сервере (например, поиск писем), параллельно продолжая просматривать сообщения.

Каждый запрос клиента начинается с уникальной метки (обычно строки вида a001, a002 и т.д.). Ответы сервера, указывающие на завершение выполнения запроса, начинаются с той метки, с которой начинался запрос. Остальные строки ответов (данные, передаваемые клиенту и служебные строки ответов, кроме последней) начинаются со служебной метки ‘*’. За меткой в ответе следует одна из трех строк: ‘OK’ — успешное выполнение запроса, ‘NO’ — запрос выполнен не был, ‘BAD’ — синтаксическая ошибка в запросе.

В таблице 10.3 приведена типовая последовательность обмена запросами и ответами во время сеанса связи по протоколу IMAP4.

Таблица 10.3 — Пример IMAP4-сеанса

№	Клиент	Сервер	Комментарий
1		* OK IMAP4 server ready	
2	a001 login xyz pass		Имя пользователя и пароль
3		a001 OK LOGINcompleted	
4	a002 select inbox		Выбор ящика «Входящие»
5		* 10 EXISTS * FLAGS (\Answered \Deleted \Seen \Draft) * 1 RECENT * OK [UNSEEN 7] Message 3 is the first unseen message * OK [UIVALIDITY 1234567876] UIDs valid a002 OK [READ-WRITE] SELECT completed	Количество сообщений в ящике (exists), количество новых (recent) и непросмотренных (unseen) сообщений
6	a003 fetch 8 rfc822.header		Получить заголовок 8-го сообщения
7		* 8 FETCH (RFC822.HEADER {150} Received: from mail.def.ru (196.11.22.33) by mail.server.ru with SMTP; 01 Jan 2004 10:12:20 GMT From: abc@def.ru To: xyz@server.ru Date: 01 Jan 2004 10:12:14 GMT Message-Id: 123@mail.def.ru MIME-Version: 1.0 Content-Type: text/plain; charset=us-ascii) a003 OK FETCH completed	Заголовок 8-го сообщения
8	a004 store 8 +flags \deleted		Удалить 8-е сообщение
8		* 8 FETCH (FLAGS (\Seen \Deleted)) a004 OK +FLAGS completed	
9	a005 logout		Закончить сеанс
10		* BYE a005 OK LOGOUT completed	

9.4 Протоколы дистанционного управления

9.4.1 Протокол виртуального терминала Telnet

Протокол виртуального терминала Telnet описан в RFC 854 и позволяет передавать на сервер коды нажатия клавиш пользователем на клиенте, как если

бы эти клавиши нажимались на консоли сервера, а алфавитно-цифровой вывод сервера на консоль передавать на клиентскую машину. Telnet-сервер рассматривает все подключенные к нему удаленные узлы клиентов как сетевые виртуальные терминалы (англ. Network Virtual Terminal, NVT) строчного типа, работающие в кодах ASCII. Протокол Telnet скрывает различия реальных терминалов, а также обеспечивает возможность согласования более сложных функций (например, эхо-вывод введенных символов, высота и ширина экрана и т.д.). Между разными терминалами могут возникать конфликты в интерпретации значений кодов символов. Так, например, одни терминалы в качестве команды перевода строки используют ASCII-символ с десятичным кодом 13 (CR, Carriage Return, возврат каретки), другие — символ с десятичным кодом 10 (LF, Line Feed, перевод строки), третьи — два символа CR и LF подряд. NVT определяет стандартные значения для управляющих символов, в частности, в качестве команды перевода строки всегда используется два символа — CR и LF. Данные в формате NTV передаются в виде 7-битовых кодов ASCII (байтов с очищенным старшим битом). Значения байтов с установленным 8-м битом зарезервированы для команд.

Telnet-клиент устанавливает TCP-соединение с Telnet-сервером на стандартном порту 23. После этого клиент договаривается с сервером об используемых опциях и переходит в режим ввода. В режиме ввода любой введенный пользователем текст передается серверу (в зависимости от установленного режима, либо посимвольно, либо построчно), а ответы сервера выводятся в окно Telnet-клиента. Для перехода в командный режим используется комбинация клавиш «Ctrl+]». В командном режиме можно использовать следующие основные команды:

- status — показать текущее состояние: имя сервера и режим обмена,
- mode <режим> — установить символьный или построчный режим обмена,
- close — закрыть текущее соединение,
- quit — выйти из программы,

- open <имя узла> [порт] — установить соединение с указанным узлом,
- ? [имя команды] — вывести справочную информацию о команде.

Поскольку протокол Telnet передает всю вводимую пользователем информацию (в том числе, пароли) в открытом виде по сети, она может быть перехвачена злоумышленниками. Использовать Telnet для доступа к удаленным узлам крайне опасно. В качестве замены рекомендуется использовать протокол безопасной командной оболочки SSH.

9.4.2 Протокол безопасной командной оболочки SSH

Протокол безопасной командной оболочки SSH (англ. Secure Shell) предназначен для выполнения тех же функций, что и протокол TELNET — подключения к удаленному узлу и выполнения на нем команд командной оболочки, как если бы они вводились с локальной консоли. Протокол SSH шифрует все передаваемые данные, что надежно защищает данные от перехвата при передаче по открытым сетям. Для аутентификации пользователей SSH позволяет использовать не только пароли, но и открытые ключи, например, RSA.

Помимо интерактивной работы в командной оболочке удаленного узла, протокол SSH позволяет организовывать защищенные тоннели для прозрачного перенаправления UDP-дейтаграмм и TCP-соединений между узлами.

В пакет программ, реализующих протокол SSH, обычно входит также утилита защищенной передачи файлов scp (англ. secure copy, безопасное копирование).

9.5 Виды конференц-связи

Конференц-связь в телефонии — услуга, обеспечивающая сеанс связи сразу с несколькими абонентами. Для компьютерных сетей под конференц-связью понимают групповую коммуникацию между территориально

удаленными друг от друга участниками. В отличие от телефонных конференций, принципиальной особенностью которых является синхронный режим: все участники должны одновременно участвовать в сеансе связи, компьютерная конференц-связь возможна как в синхронном (он-лайн), так и в асинхронном (офф-лайн) режиме.

9.5.1 Виды асинхронной конференц-связи

При обмене сообщениями в асинхронном режиме, участники в большинстве случаев передают друг другу текстовые либо текстово-графические документы, хотя возможна также передача звуковых файлов, видеофайлов и произвольных двоичных файлов.

Простейший вид асинхронной конференц-связи — групповой обмен сообщениями электронной почты: каждый участник вручную направляет свои письма всем остальным участникам. Для организации такого режима не требуется никаких дополнительных программно-аппаратных средств, помимо обычных почтового клиента и почтового сервера. Однако при этом каждый участник вынужден самостоятельно поддерживать список всех участников и своевременно вносить в него изменения.

Второй вид асинхронной конференц-связи — списки рассылки (англ. mailing list): специальное программное обеспечение (сервер списка рассылки, англ. listserv), сходное по функциональности с почтовым сервером, обеспечивает централизованное хранение и модификацию списка участников, среди которых различаются участники, имеющие право только получать сообщения и участники с правом рассылки своих сообщений. Участник, имеющий право рассылки, направляет свои письма серверу списка рассылки, а сервер рассылает их по электронной почте всем остальным участникам. Как правило, внесение и исключение участника из списка рассылки не требует вмешательства администратора, а выполняется сервером автоматически при

получении письма с соответствующей командой. Со стороны участника для работы со списком рассылки достаточно обычного почтового клиента.

Третий вид асинхронной конференц-связи — телеконференции (англ. newsgroups, новостные группы), представляющие собой распределенную систему групповой доставки сообщений, не зависящую от электронной почты. Специальное программное обеспечение (сервер телеконференций, англ. newsserv), хранит всех сообщения, направленные в данную телеконференцию, и предоставляет к ним доступ по протоколу NNTP. Для получения и отправки сообщений, участник телеконференции обычно должен воспользоваться специальной программой — NNTP-клиентом, но некоторые серверы предоставляют и Web-интерфейс к спискам телеконференций, сообщений в них и самим сообщениям. Крупнейшая система телеконференций — Usenet — имеет иерархическое устройство пространства имен. Например, телеконференция с именем comp.databases.ms-sqlserver принадлежит подгруппе comp.databases группы comp. Важная особенность систем телеконференций — их распределенный характер: сообщения хранятся на множестве взаимосвязанных серверов. Каждый сервер, участвующий в Usenet, может поддерживать произвольную часть пространства имен, храня сообщения только тех телеконференций, которые интересуют его владельца. Сообщение, поступившее в любую телеконференцию Usenet через некоторый сервер, будет передано им другим серверам, хранящим сообщения этой телеконференции.

Четвертый вид асинхронной конференц-связи — форумы, организуемые на Web-сайтах. Программное обеспечение форума позволяет через Web-интерфейс определять темы для обсуждения, создавать сообщения и отвечать на сообщения других участников. Одна из полезных дополнительных функций форумов — рассылка по электронной почте уведомлений об ответах и новых сообщениях. В отличие от телеконференций, обычно форум — не распределенная, а локальная система групповой доставки сообщений: он функционирует на одном Web-сайте и не имеет автоматических связей с другими форумами.

Пятый вид асинхронной конференц-связи — электронные доски объявлений (англ. bulletin board system, BBS). Специализированное программное обеспечение электронных досок объявлений устанавливается на компьютер, снабженный модемом, автоматически отвечающим на звонки (режим dial-in), и обычно обладает функциями файлового архива и телеконференций/форума. Для доступа к электронной доске объявлений необходимо установить с ней модемное соединение и запустить программу универсального терминала (HyperTerminal, Telemate, SecureCRT и др.).

Шестой вид асинхронной конференц-связи — системы дневников (англ. weblog, blog). Программное обеспечение систем дневников позволяет множеству пользователей через Web-интерфейс либо с помощью специальной программы-клиента вести личные дневники, списки дел и т.д., комментировать записи в дневниках других пользователей, составлять личные списки пользователей-“друзей” для автоматического формирования ленты последних записей в их дневники. Системы дневников обычно не только поддерживают все функции форумов, но и расширяют их в части большей персонализации интерфейса. Одна из наиболее массовых и известных систем дневников — «Живой журнал» http://www.livejournal.com.

9.5.2 Виды синхронной конференц-связи

Синхронная конференц-связь подразумевает одновременное участие сторон в сеансе связи.

Первый тип синхронной конференц-связи — мгновенный обмен текстовыми сообщениями. Системы, поддерживающие этот тип, часто называют системами Интернет-пейджинга. Наиболее известные представители систем этого класса — ICQ, MSN Messenger и др. Для того, чтобы обмениваться сообщениями с другими участниками такой системы, каждый участник должен либо установить специальную программу-клиент, либо (если производитель предлагает такой вариант) воспользоваться Web-интерфейсом к

системе. Подобные системы поддерживают список участников на центральном сервере и обслуживают подключения/отключения участников, а обмен сообщениями между участниками происходит напрямую, без участия центрального сервера. Передаваемые сообщения немедленно достигают получателя, если только он в данный момент подключен к системе. В противном случае они сохраняются на сервере и выдаются получателю при следующем его подключении. Системы этого типа поддерживают как попарное общение участников, так и совместное общение небольшой группы.

Второй тип синхронной конференц-связи — чаты (англ. chat — болтовня). Чат — это система обмена текстовыми сообщениями для большого количества участников. Классическая система чатов — IRC (англ. Internet Relay Chat), представляющая собой распределенную систему специализированных серверов, для доступа к которым необходимо воспользоваться специальной программой-клиентом (mIRC, Pirch и др.). Общение участников в IRC происходит в отдельных именованных каналах. Как и в Usenet, отдельные серверы могут поддерживать любое подмножество каналов. Кроме того, существуют отдельные IRC-сети со своими серверами, не связанные с остальными. Любой участник может просматривать список каналов на сервере, и подключаться к выбранным каналам, либо создавать свои каналы. Создатель канала может определять политику доступа к нему других участников, назначать пароли для подключения, скрывать или открывать список участников и т.д. Второй распространенный тип чатов — Web-чаты, для доступа к которым участнику достаточно Web-браузера. В простейшем случае Web-чат — это Web-страница, включающая форму ввода сообщений и регулярно (раз в несколько секунд) обновляемое окно со списком последних введенных участниками сообщений.

Третий тип синхронной конференц-связи — голосовые (аудио) конференции. Часто для описания пользовательских Интернет-технологий, поддерживающих аудиоконференции, используется термин «IP-телефония» (англ. Voice over IP, VoIP — голос по IP). Участники таких конференций

должны иметь аппаратные средства для голосового общения (гарнитура или микрофон и колонки/наушники) и установить специальную программу. Одна из известных систем аудиоконференций — Microsoft NetMeeting. В настоящее время наибольшее распространение получила бесплатная система IP-телефонии Skype. Голосовые возможности имеются также в последних версиях ICQ. Основное ограничение на возможность использования аудиоконференций накладывает пропускная способность канала. В зависимости от используемых методов сжатия, минимальная пропускная способность канала для приемлемого качества передачи голоса, находится в пределах 8-16 кбит/с.

Четвертый тип синхронной конференц-связи — видеоконференции. Этот способ связи подразумевает передачу видеоизображений и звука для достижения возможности интерактивного общения. Требования к пропускной способности каналов передачи данных для видеоконференциях еще выше, чем для аудиоконференций — не менее 64 кбит/с для минимального качества изображения и не менее 200 кбит/с для приемлемого качества. Участники таких конференций должны иметь подключаемую к компьютеру видеокамеру, аппаратные средства для голосового общения (гарнитура или микрофон и колонки/наушники). Используются как чисто программные реализации алгоритмов обработки видеосигнала, так и программно-аппаратные решения на основе специализированных плат (кодеков), содержащие цифровые сигнальные процессоры.

Рекомендация ITU-T H.323 «Видеотелефонные системы и терминальное оборудование для локальных сетей с негарантированным качеством услуг» (версия 2 — «Мультимедийные системы связи для сетей с коммутацией пакетов») — определяет стандарты для видео-конференц-связи в локальных, корпоративных и глобальных сетях с коммутацией пакетов.

9.6 Вопросы для самопроверки

- Зачем в протоколе FTP используется два TCP-соединения?

- В каких условиях оправдано применение протокола TFTP?
- Почему для получения электронной почты недостаточно протокола SMTP?
- В чем заключаются преимущества и недостатки протокола IMAP по сравнению с протоколом POP?
- По какой причине протокол Telnet считается небезопасным?
- Что такое конференц-связь? Сравните асинхронные и синхронные виды конференц-связи.
- Чем отличается Web-чат от IRC?

10 Web-технологии

10.1 Структура Web-ориентированного программного обеспечения

Всемирная паутина (англ. World Wide Web, WWW, Web) — это распределенная гипертекстовая информационная система, работающая в сетях TCP/IP. В основе Web лежит механизм, дающий клиентам (браузерам, от англ. browse — просматривать, перебирать) возможность получать от Web-серверов структурированную информацию в виде гипертекстовых HTML-файлов (англ. HyperText Markup Language, язык разметки гипертекста). Отдельный HTML-файл представляет собой текстовый файл, в который внедрены специальные текстовые команды разметки — тэги (англ. tag). Все допустимые команды разметки можно разделить на несколько групп. Команды разметки первой группы формируют гипертекстовую структуру — они содержат ссылки на другие HTML-файлы, графические файлы, звуковые файлы, видеофайлы, файлы архивов, документов и т.д. Команды разметки второй группы описывают внутреннюю структуру данного гипертекстового документа (из каких частей он состоит), а третьей группы — указывают на способы визуального представления частей HTML-файла. Для передачи запросов Web-серверу и получения файлов от него, клиенты используют специальный протокол — Протокол передачи гипертекста (англ. HyperText Transfer Protocol, HTTP).

Протокол HTTP ориентирован на доставку текстовых файлов, и не предназначен для передачи произвольных двоичных файлов. Для преобразования любых двоичных файлов в текст, из которого можно будет затем однозначно восстановить исходный файл, в HTTP используются методы кодирования MIME (англ. Multipurpose Internet Mail Extensions, Многоцелевые расширения электронной почты), разработанные для передачи любых файлов по электронной почте. Этот стандарт можно использовать для кодирования множества разных объектов, таких, как текстовые файлы с разметкой, файлы в

формате PostScript, графические файлы в форматах GIF и JPEG, аудиофайлы, видеофайлы и многое другое. MIME можно использовать и для управления способом представления (например, можно показывать картинку и одновременно воспроизводить звуковой файл).

Браузер — это интерпретатор языка HTML, который получает файлы с сервера и выполняет команды разметки, либо меняя способ изображения текста на экране (размер шрифта, выделение и т.п.), либо запуская соответствующие программы просмотра для объектов других типов. Некоторые из этих объектов могут быть сценариями, написанными на специальных языках программирования (напр., JavaScript, VBScript). Такие сценарии передаются на выполнение интерпретаторам соответствующих языков программирования. Ссылки на другие HTML-файлы (гиперссылки) визуально выделяются, а при их выборе (щелчком мыши или клавишей Enter) браузер загружает и отображает файл, на который указывает ссылка. В процессе переходов по гиперссылкам браузер запоминает их последовательность (историю) и позволяет легко вернуться к ранее просмотренным файлам. Браузеры могут иметь как графический (Microsoft Internet Explorer, Mozilla Firefox, Opera, и т.д.), так и текстовый (Lynx, Links и др.) интерфейс.

Для указания файлов, которые они хотят получить с сервера, браузеры используют Однородные указатели ресурсов (англ. Uniform Resource Locator, URL). URL состоит из трех частей:

- Идентификатор используемого для доступа протокола,
- Имя сервера, хранящего нужный объект,
- Имя файла (страницы) на сервере.

Например, в URL http://www.lenta.ru/index.htm подстрока «http://» указывает на протокол HTTP, подстрока «www.lenta.ru» определяет имя сервера, а подстрока «/index.htm» определяет имя файла на сервере.

С помощью браузеров можно также заполнять формы ввода информации для передачи ее на Web-сервер. Введенная в форму информация передается Web-сервером для обработки другим программам. Такие программы должны

удовлетворять требованиям Общего шлюзового интерфейса (англ. Common Gateway Interface, CGI). Используя формы, Web-сервер может предоставлять универсальный простой, дружественный и очень мощный интерфейс к различным источникам информации. Например, авиакомпания может использовать формы для заказа билетов. Когда вы ввели данные в форму и отправили ее на сервер, соответствующая программа получит ваши данные через CGI и обратится к нужным базам данных для выполнения заказа: БД самолетов, БД расписания полетов, БД счетов, БД прошлых заказов и т.д. Все эти сложные действия будут происходить незаметно для пользователя, который будет работать только с простой формой.

Из Web-серверов наиболее распространен Apache Web Server, работающий как в большинстве Unix-подобных ОС, так и в Microsoft Windows. Фирма Microsoft поддерживает собственный комплекс — Microsoft Internet Information Server (IIS), включающий Web- и FTP-сервер.

10.2 Протокол передачи гипертекста HTTP

Протокол передачи гипертекста HTTP (HyperText Transfer Protocol) описан в RFC 2616. HTTP-сервер ожидает поступления запросов от HTTP-клиентов. Каждый запрос обслуживается независимо от других запросов как этого, так и других клиентов. Процесс обработки запроса состоит из четырех фаз:

- клиент устанавливает TCP-соединение с портом на сервере (стандартный порт - 80);
- клиент передает HTTP-запрос серверу в виде последовательности ASCII-строк;
- сервер передает HTTP-ответ в виде последовательности ASCII-строк;
- клиент или сервер закрывает соединение.

10.2.1 HTTP-запрос

Первая строка HTTP-запроса содержит название метода — GET, POST, HEAD, PUT, OPTIONS, DELETE, TRACE. В основном используются первые два метода. Метод GET запрашивает передачу содержимого ресурса, указанного с помощью уникального идентификатора ресурса (Universal Resource Identifier, URI), то есть имени объекта, находящегося на сервере, следующего за именем метода. В простейшем случае клиент запрашивает файл — текст, изображение и т.п. Кроме того, за URI может скрываться программа, выдающая те или иные данные. Для клиента нет никаких различий, передается ли ему файл, взятый с жесткого диска ("статический") или сгенерированный программой ("динамический").

Метод POST используется для передачи данных клиентом серверу, например, введенных пользователем в поля формы.

Строки HTTP-запроса, начиная со второй и до первой пустой строки должны содержать строки HTTP-заголовка — набор параметров в форме пар "имя: значение". Список возможных имен параметров определен стандартом HTTP и в версии 1.1 состоит из 46 имен.

За пустой строкой может следовать тело HTTP-запроса, если используемый метод допускает его наличие. Так, при использовании метода POST необходимо передаваемые серверу данные поместить именно в тело HTTP-запроса. Метод GET не подразумевает наличия у HTTP-запроса тела.

Простейший HTTP-запрос имеет вид:

GET /

(пустая строка)

(пустая строка)

В ответ на такой HTTP-запрос сервер выдает страницу по-умолчанию.

Более сложный вариант запроса той же страницы по-умолчанию:

GET / HTTP/1.1

Host: 127.0.0.1:80

Accept: text/xml,text/html,text/plain,image/jpeg,image/gif,*/*

Accept-Language: ru,en-us

Accept-Encoding: gzip,deflate

Accept-Charset: windows-1251,utf-8,*

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)

Connection: Keep-Alive

Если в HTML-странице присутствует определение формы, например, такое:

<form name=form1 action=/search method=POST>

<input maxLength=256 size=55 name=q value="">

<input type=submit value="Search" name=btnG>

</form>

Тогда, если пользователь запишет в поле ввода строку «something» и нажмет на кнопку “Search”, то будет сформирован подобный HTTP-запрос:

POST /search HTTP/1.1

Host: 127.0.0.1:80

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; ru-RU)

Accept: text/xml,text/html,text/plain,image/jpeg,image/gif,*/*

Accept-Language: ru,en-us

Accept-Encoding: gzip,deflate

Accept-Charset: windows-1251,utf-8,*

Connection: keep-alive

Referer: http://127.0.0.1/index.htm

q=something

btnG=Search

10.2.2 HTTP-ответ

HTTP-ответ сервера, так же, как и HTTP-запрос, состоит из нескольких частей. Первая строка содержит численный код результата выполнения запроса и его словесное описание. Код состоит из трех цифр, причем первая из них означает, был ли запрос успешен или нет: 2 — успешное выполнение, 3 — перенаправление (ресурс был перемещен), 4 — ошибка в запросе, 5 — ошибка сервера. Вторая и третья цифры уточняют результат.

Строки со второй по первую пустую, как и в HTTP-запросе, содержат HTTP-заголовок — набор параметров.

В случае успешного выполнения метода GET за HTTP-заголовком и пустой строкой следует тело HTTP-ответа - содержимое запрошенного ресурса. В случае выполнения метода POST, HTTP-ответ тела не имеет.

Примерный вид HTTP-ответа:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.1
Date: Sun, 07 Jan 2004 10:13:14 GMT
Connection: Keep-Alive
Content-Length: 55
Content-Type: text/html

<html><head><title>TITLE</title></head><body></body></html>
```

10.2.3 Средства идентификации пользователей в протоколе HTTP

Протокол HTTP не содержит средств для хранения информации о клиентах в промежутке между обращениями. Все обращения клиентов к серверу совершенно независимы друг от друга. Это позволило значительно упростить сам протокол, однако для удобства работы со сколько-нибудь сложными системами, предоставляющими доступ к данным, необходимы

средства, позволяющие различать пользователей, и, возможно, хранить некоторую информацию о них.

Средства эти должны быть поддержаны как Web-серверами, так и браузерами: необходимо, чтобы браузер передавал идентификационную информацию.

В настоящее время применяется два основных способа идентификации пользователей:

- механизм, использующий поля SetCookie и Cookie в заголовке запроса
- механизм явного запроса идентификатора при посещении корневой страницы и динамического создания всех остальных страниц.

Cookie-механизм возможен благодаря ведению браузером базы данных, в которой хранятся небольшие блоки данных, полученные им в полях SetCookie ответов от Web-серверов. Если пользователь запрашивает страницу с сервера, для которого есть такой блок данных, то его содержимое отправляется в составе запроса в поле Cookie. Таким образом, Web-сервер имеет возможность сгенерировать уникальный идентификатор для каждого пользователя, передать его браузеру, а когда этот же пользователь вновь подключится, его браузер сам предъявит этот идентификатор. Этот механизм на самом деле позволяет различать не пользователей, а отдельные установки браузеров.

Механизм явного запроса идентификатора лишен этого недостатка. Создается корневая страница, включающая HTML-форму, в которую пользователь должен ввести свой идентификатор и, возможно, пароль. Обязательным условием является предварительная регистрация пользователя, во время которой проверяется уникальность идентификатора.

10.3 Языки и средства создания Web-приложений

Веб-приложения в большинстве случаев состоят из двух компонентов: серверного и клиентского, отличающихся местом выполнения. Серверный компонент – это по сути CGI-программа, выполняемая при поступлении HTTP-

запроса, генерирующая Web-страницу, состоящую из HTML-разметки и программного кода клиентского компонента.

Для разработки серверных компонентов Web-приложений может быть использован практически любой доступный язык программирования — языки ассемблера, универсальные языки Си, Си++, Си#, Java, Паскаль, Visual Basic, и др., языки описания сценариев Perl, PHP, Python, Ruby и т.д., языки командных оболочек Bash, KSh, WSh и др., и их комбинации. Наибольшее распространение в настоящее время получили специализированный язык описания сценариев PHP. На платформе Microsoft Windows широко используется технология ASP.NET.

Код клиентского компонента либо непосредственно интерпретируется Web-браузером (сценарии на языке JavaScript), либо Web-бразуер запускает специализированную виртуальную машину, которая выполняет этот код (апплеты Java), либо Web-браузер вызывает функции из скомпилированных под конкретную операционную систему и аппаратную платформу модулей (технология ActiveX). Существуют универсальные клиентские ActiveX-компоненты с собственными языками программирования, такие, как Macromedia Flash.

10.4 Вопросы для самопроверки

- Что означает приставка «гипер» в термине «гипертекст»?
- Каким образом пользователь может отличить статический файл, переданный Web-сервером, от сгенерированного CGI-программой?
- Какой метод протокола HTTP позволяет передать на сервер данные, введенные пользователем в формы на Web-странице?
- Как действует механизм Cookie?

Список литературы

1. Олифер В.Г., Олифер Н.А. **Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов**. 3-е изд. — СПб: Питер, 2005. — 960 с.
2. Волков А.Н., Кузин А.В., Пескова С.А. **Сети и телекоммуникации. Учебное пособие для вузов.** — М.: Academia, 2006. — 352с.
3. Галкин В.А., Григорьев Ю.А. **Телекоммуникации и сети.** — М.: МГТУ им Н.Э.Баумана, 2003. — 608с.
4. **Руководство по технологиям объединенных сетей.** 4-е изд. — М.: Вильямс, 2005. — 1040с.
5. Таненбаум Э.С. **Компьютерные сети.** 4-е изд. — СПб.: Питер, 2003. — 992с.
6. Столлингс В. **Современные компьютерные сети.** 2-е изд. — СПб.: Питер, 2003. — 784с.
7. Ломовицкий В.В., Михайлов А.И., Шестаков К.В., Щекотихин В.М. **Основы построения систем и сетей передачи информации. Учебное пособие для вузов.** — М.: Горячая линия — Телеком, 2005. — 382с.
8. Спортак М. **Компьютерные сети и сетевые технологии. Platinum Edition.** — СПб.: Диасофт, 2005. — 720с/
9. Куроуз Дж., Росс К. **Компьютерные сети. Многоуровневая архитектура Интернета.** 2-е изд. — СПб.: Питер, 2004. — 768с.
10. Паркер Т., Сиян К. **TCP/IP для профессионалов.** — СПб.:Питер, 2004. — 864с.
11. Храмцов П.Б., Брик С.А., Русак А.М., Сурин А.И. **Основы Web-технологий. Курс лекций.** — М.: Интернет-университет информационных технологий, 2003. — 512с.

Интернет

1. Русские документы: сетевые технологии http://www.rusdoc.ru/net.shtml
2. Сетевые протоколы http://www.protocols.ru
3. Техническая библиотека компании BiLiM Systems http://www.bilim.com
4. Сервер информационных технологий: сетевые технологии http://www.citforum.ru/nets/
5. Книги по сетям http://document.newmail.ru/la.htm
6. Термины и основные понятия телекоммуникаций http://www.online.ru/it/helpdesk/race/index.htm

Учебное издание

Брейман Александр Давидович

Сети ЭВМ и телекоммуникации. Глобальные сети.

Учебное пособие.