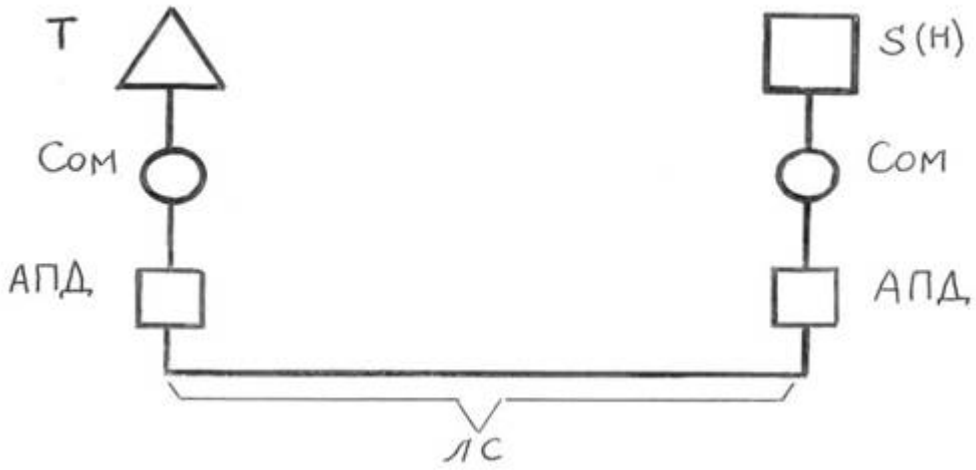
Билет 1. Тракт телеобработки данных (ТОД)

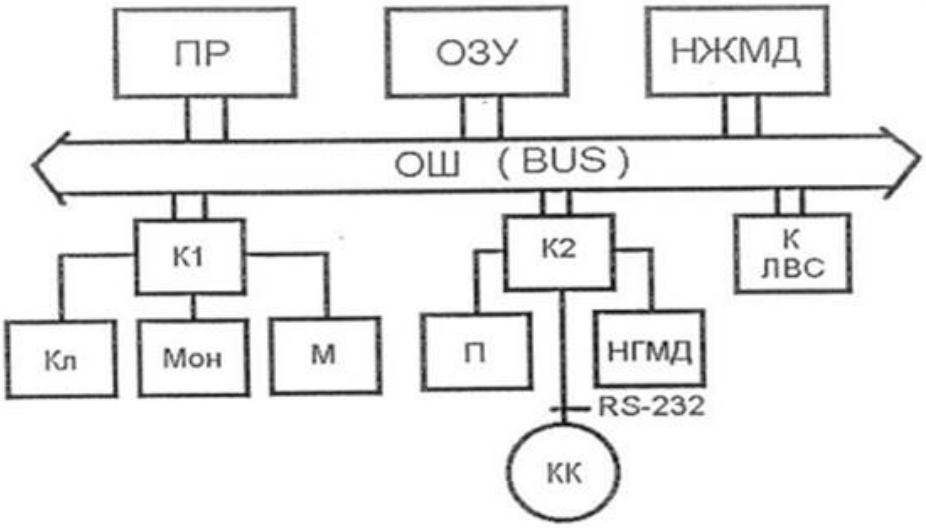
Типовой тракт телеобработки данных (ТОД) представляет фрагмент вычислительной сети, который состоит из следующих узлов:

- ❖Хостмашины (Н) или сервера S
- ❖Коммутирующего узла (Com)
- ❖Аппаратуры передачи данных (АПД)
- ❖Линии связи (ЛС)
- ❖Терминального узла (Т).



Хостмашины, сервера и терминальные узлы представляют собой компьютер, который содержит в своем составе:

- ❖ПР – процессор
- ❖ОЗУ – оперативное запоминающее устройство, хранящее: данные в регистрах (Рег), программы операционной системы (ОС), канальные программы (КПр)
- ❖НЖМД – накопитель на жестком магнитном диске
- ❖ОШ – общая шина
- ❖К1, К2 – многопортовые контроллеры внешних устройств
- ❖КЛ – клавиатура
- ❖Мон – монитор
- ❖М – мышь
- ❖П – принтер
- ❖НГМД – накопитель на гибком магнитном диске.



Модем (модулятор-демодулятор) является устройством, преобразующим последовательные цифровые сигналы в аналоговые и наоборот.

Линии связи образуют физическую среду передачи сигналов, к которой относятся телефонная пара, витая пара, коаксиальный кабель, волоконно-оптический кабель, «эфир» электромагнитных полей.

АПД, включающие аппаратные средства и необходимое программное обеспечение, вместе с линиями связи линии связи обеспечивают реализацию каналов связи, которые по функциональным особенностям разделяются на:
Симплексные каналы связи обеспечивают передачу информации только в одном направлении (телевидение)
Дуплексные каналы связи обеспечивают передачу информации одновременно и независимо в двух направлениях (поддержка диалога)
Полудуплексные каналы связи обеспечивают передачу информации поочередно: то в одном, то в другом направлении (обычно спецсвязь)
При передаче данных могу использоваться 2 принципа связи: коммутация линий связи и коммутация пакетов. При коммутации линий на время передачи данных организуется физическое соединение нескольких линий. При коммутации пакетов в каждом коммутирующем узле имеется буфер, в котором каждое передаваемое сообщение (пакет данных), может ожидать освобождения очередного канала для передачи данных.

Функции тракта ТОД:

❖Каналы связи и АПД выполняют набор функций, обеспечивающий доставку блока цифровых данных на любые (даже удаленные расстояния)

❖Коммутирующий узел выполняет набор функций, обеспечивающий выбор маршрута при реализации режимов коммутации каналов и коммутации пакетов для обмена блоками цифровых данных между терминальным узлом и хостмашиной или сервером.

❖Хостмашина, сервер, терминальный узел для обмена блоками цифровых данных между терминальным узлом и хостмашиной или сервером выполняют 2 набора функций:

- Обеспечение транспортировки больших объемов данных
- Запуск удаленных прикладных программ и получение результатов выполнения обработки.

Существует 7 основных уровней модели OSI:
7-й Прикладной (Application)
6-й Представление (Presentation)
5-й Сеансовый (Session)
4-й Транспортный (Transport)
3-й Сетевой (Network)
2-й Канальный (Data Link)
1-й Физический (Physical)

Функции модели OSI	**Функции ТОД**
7-й Прикладной (Application)	
6-й Представление (Presentation)	Запуск удаленных прикладных программ и получение результатов выполнения обработки.
5-й Сеансовый (Session)	
4-й Транспортный (Transport)	Транспортировка больших объемов данных
3-й Сетевой (Network)	Выбор маршрута при реализации режимов коммутации каналов и коммутации пакетов для обмена блоками цифровых данных между терминальным узлом и хостмашиной или сервером
2-й Канальный (Data Link)	Доставка блока цифровых данных на любые расстояния
1-й Физический (Physical)	

Билет 2. Форматы кадров Ethernet. Схема протокола логического контроля соединения (LLC).

В настоящее время термин Ethernet чаще всего используют для описания всех локальных сетей, работающих в соответствии с принципами CSMA/CD (Carrier Sense Multiple Access/Collision Detection) – множественного доступа с контролем несущей и обнаружением коллизий, что соответствует спецификации Ethernet IEEE 802.3. В модели OSI протокол CSMA/CD относится к доступу к среде. На этом уровне определяется формат, в котором информация передается по сети, и

способ, с помощью которого сетевое устройство получает доступ к сети (или управление сетью) для передачи данных.

CSMA определяет, каким образом рабочая станция с сетевым адаптером «ловит» момент, когда ей следует послать сообщение. Рабочая станция вначале слушает сеть, чтобы определить, не передается ли в данный момент какое-либо другое сообщение. Если слышится несущий сигнал (carrier tone), значит, в данный момент сеть занята другим сообщением – рабочая станция переходит в режим ожидания и находится в нем до тех пор, пока сеть не освободится. Когда в сети наступает молчание, станция начинает передачу.

CD служит для разрешения ситуаций, когда две или более рабочие станции пытаются передавать сообщения одновременно. Если 2 станции начнут передавать свои пакеты одновременно, передаваемые данные належатся друг на друга и ни одно из сообщений не дойдет до получателя. Такую ситуацию называют конфликтом или коллизией (сигналы одной станции перемешаются с сигналами другой). CD требует, чтобы станция прослушала сеть также и после передачи пакета. Если обнаруживается конфликт, станция повторяет передачу пакета через случайным образом выбранный промежуток времени. Затем она вновь проверяет, не произошел ли конфликт. Термин «множественный доступ» подчеркивает тот факт, что все станции имеют одинаковое право на доступ к сети.

Основные спецификации Ethernet

10Base5. Эта спецификация в качестве среды передачи предусматривает толстый коаксиальный кабель на 50 Ом с двумя оболочками. Каждый коаксиальный кабель в сети образует отдельный сегмент. Протяженность сегмента не может превышать 500 м, а число узлов не должно превосходить 100. Отрезок кабеля между соседними узлами должен быть не менее 2.5 м. Это позволяет уменьшить вероятность отражения и появления стоячих волн. Как правило, производители размечают кабель с целью упростить нахождение мест, где станция может быть подключена к сегменту. Сетевой адаптер подключается к кабелю с помощью трансиверного кабеля или трансивера. Длина трансиверного кабеля не должна превышать 50 м.

10Base 2. Эта спецификация предусматривает использование тонкого коаксиального кабеля, а также соединителей типа BNC-T, которые непосредственно связывают сетевой адаптер и кабель Ethernet. Такая схема исключает необходимость применения дорогостоящих трансиверов и трансиверных кабелей. Кроме того, значительно упрощается выполнение самой операции по подключению сетевого адаптера к кабелю. Протяженность сегмента ограничена 185 м, а число узлов – 30.

10BaseT. Эта разновидность Ethernet получила наибольшее распространение. Буква T в названии означает, что средой передачи является неэкранированная витая пара (Unshielded Twisted Pair, UTP). Спецификация предусматривает использование концентратора для подключения пользователей по топологии «звезда». Применение дешевых UTP является одним из основных преимуществ 10BaseT по сравнению с 10Base5 и 10Base2. Подключение узлов к сети осуществляется с помощью телефонных гнезд RJ-45 и RJ-11 и четырехпарного телефонного кабеля UTP. Вилка RJ-45 вставляется напрямую в сетевую плату. Протяженность отрезка кабеля от концентратора до станции не должна превышать 100 м (в случае UTP категории 3) и 150 м (в случае UTP категории 5).

10BaseF. Эта спецификация использует в качестве среды передачи оптоволоконный кабель. Применение оптоволоконной технологии приводит к высокой стоимости комплектующих. Однако нечувствительность к электромагнитным помехам позволяет использовать спецификацию в особо ответственных случаях и для связи далеко расположенных друг от друга объектов.

Существуют 4 основных разновидности кадров Ethernet:

- ❖Ethernet Type II
- ❖Ethernet 802.3
- ❖Ethernet 802.2
- ❖Ethernet SNAP (SubNetwork Access Protocol)

Общий формат кадров Ethernet

Преамбула (56 бит)	Предназначены для синхронизации отправителя и получателя.
Признак начала кадра (8 бит)	
Адрес получателя (48 бит)	Физический адрес устройства в сети, которому адресован данный кадр.

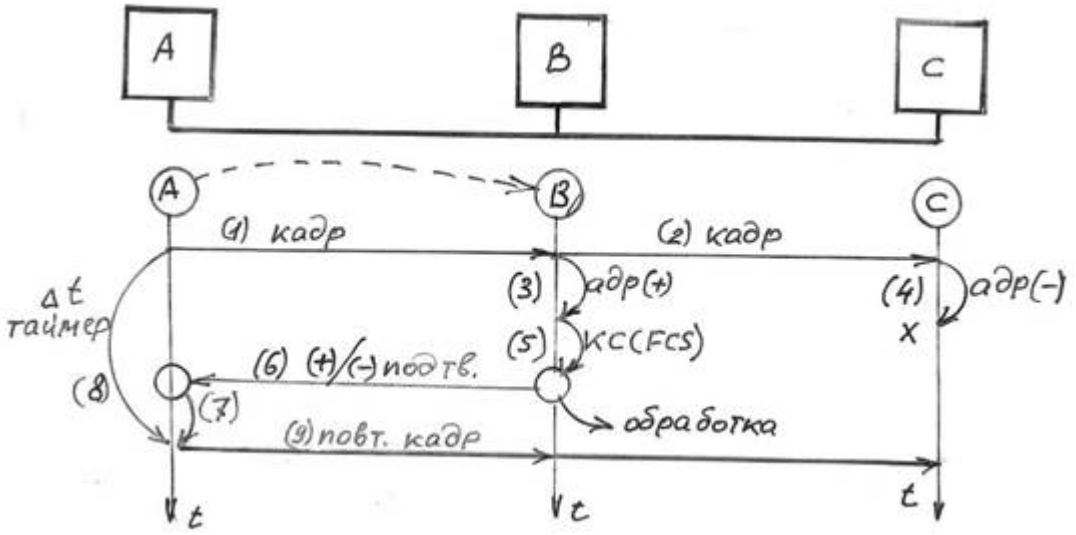
Адрес отправителя (48 бит)	Физический адрес устройства в сети, которое отправило кадр.
Длина/тип (16 бит)	Длина или тип кадра в зависимости от используемого кадра Ethernet.
Данные (переменная длина)	Данные кадра. Чаще всего информация, нужная протоколам верхнего уровня.
Контрольная сумма (32 бит)	Результат вычисления контрольной суммы всех полей за исключением преамбулы, признака качала кадра и самой КС.

Схема протокола логического контроля соединения (LLC)

Стандарт на технологии OSI определяет канальный и физический уровни передачи данных. Канальный уровень ЛВС разбит на 2 подуровня: логического контроля соединения (LLC – Logical Link Control) и контроля доступа к среде (MAC – Medium Access Control). На канальный уровень возлагается ответственность за обеспечение надежной передачи данных между двумя узлами сети. Получая пакет для передачи с более высокого сетевого уровня, канальный уровень присоединяет к этому пакету адреса получателя и отправителя, формирует из него набор кадров для передачи и обеспечивает выявление и исправление ошибок.

Нижний подуровень канального уровня – контроль доступа к среде – при передаче окончательно формирует кадр передачи в соответствии с тем протоколом, который реализован в данном сегменте. При получении пакета этот подуровень проверяет адрес, контрольную сумму и наличие ошибок при передаче.

Верхний подуровень канального уровня – логический контроль соединения – обеспечивает режимы передачи данных, как с установлением, так и без установления соединения.



Станция А формирует и отправляет кадр к станции В (1), который по единой среде канала связи достигает станции С. Одновременно на станции А взводится таймер контроля длительности подтверждения приема. На станции В (3) при проверке адреса получателя устанавливается, что кадр доставлен правильно. На станции С (4) из-за несовпадения адреса назначения кадр удаляется. На станции В (5) проверяется контрольная сумма и, если контрольная сумма совпадает, то формируется (6) положительный кадр (+) подтверждения и кадр поступает на обработку, если контрольная сумма не совпадает, то формируется (6) отрицательный кадр (–) подтверждения и кадр удаляется. На станции А при поступлении отрицательного кадра (–) и при срабатывании таймера (8) осуществляется повторная передача (9) кадра.

Билет 3. Протокол ARP. Формат сообщения ARP.

Протокол ARP (Address Resolution Protocol) – протокол разрешения адреса. Необходимость протокола ARP продиктована тем обстоятельством, что IP-адреса устройств в сети назначаются независимо от физических адресов. Поэтому для доставки сообщений по сети необходимо определить соответствие между физическим адресом устройства и его IP-адресом – это называется **разрешением адресов**.

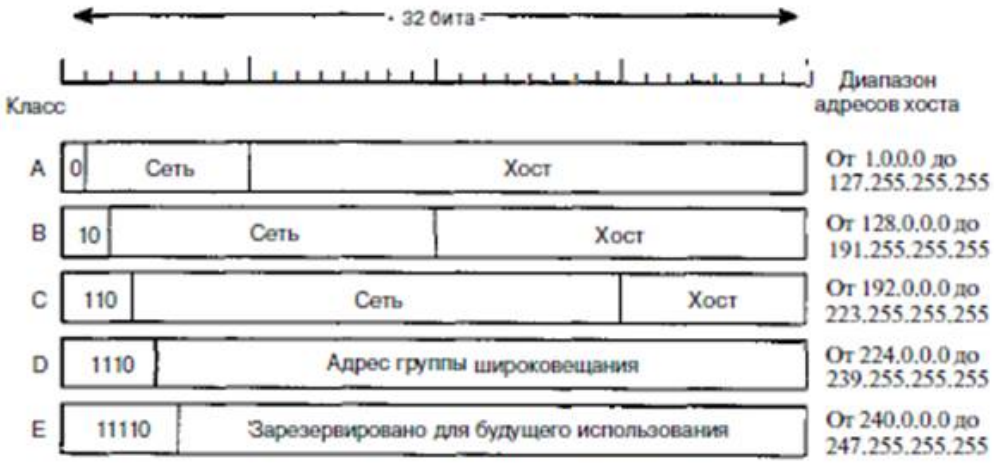
Функционально протокол ARP состоит из двух частей. Одна часть протокола определяет физические адреса при посылке дейтаграммы, другая отвечает на запросы от других устройств в сети. Протокол ARP предполагает, что каждое устройство знает как свой IP-адрес, так и свой физический адрес.
Для того, чтобы уменьшить количество посылаемых запросов ARP, каждое устройство в сети, использующее протокол ARP, должно иметь специальную буферную память, т. е. ARP-таблицу. В ней хранятся пары (IP-адрес, физический адрес).
В ARP-таблице могут быть как статические, так и динамические записи. Динамические записи добавляются и удаляются автоматически. Статические записи могут быть добавлены пользователем. Кроме того, таблица всегда содержит запись с физическим широковещательным адресом (%FFFFFFFFFFFF) для локальной сети. Время жизни каждой записи в таблице 10 минут.
Формат сообщения ARP.

	0 7	8 15	16 24	25 31
1	Тип сети (16 бит)		Тип протокола (16 бит)	
2	Длина аппаратного адрес адреса (8 бит)	Длина сетевого адреса(8 бит)		Тип операции (16 бит)
3	Аппаратный адрес отправителя (32 бит)			
4	+Аппаратный адрес отправителя (16 бит)		IP-адрес отправителя (16 бит)	
5	+IP-адрес отправителя(16 бит)			| Аппаратный адрес получателя (16 бит)
6	+Аппаратный адрес получателя (32 бита)			
7	IP-адрес получателя (32 бита)			

В поле **«Тип сети»** для сетей Ethernet указывается 1. Для других типов сетей значение этого поля определено соответствующими документами RFC. Поле **«Тип протокола»** позволяет использовать сообщения ARP не только для протокола IP, но и для других сетевых, протоколов. Для протокола IP значение этого поля равно 0800 (шестнадцатеричное). В поле **«Тип операции»** для ARP-запросов указывается 1, а для ARP-ответов – 2. Поля **«Длина аппаратного адреса»** и **«Длина сетевого адреса»** позволяют использовать протокол ARP в любых сетях, так как длину адресов можно задать. Протокол ARP работает по следующей схеме: устройство, отправляющее ARP-запрос, заполняет в сообщении все поля, кроме искомого аппаратного адреса. Затем оно рассылает запросы по всей подсети. Поле заполняется устройством, опознавшим свой IP-адрес.
Пример ARP-таблицы

IP-адрес	**Ethernet-адрес (MAC)**
223.1.2.1	08:00:39:00:2F:C3
223.1.2.3	08:00:5A:21:A7:22

Билет 4. Базовая адресация IP4. Имена сетей и узлов. Подсети. Маска подсети. IP-таблица маршрутов.
IP-адреса узла ВС идентифицирует точку доступа модуля IP к сетевому интерфейсу, а не всю машину. Менеджер сети присваивает IP-адреса машинам в соответствии с тем, к каким IP-сетям они подключены. Старшие биты 4-ч байтного IP-адреса определяют номер IP-сети. Оставшаяся часть IP-адреса – номер узла (хост-номер).
Существует 5 классов IP-адресов, отличающиеся количеством бит в сетевом номере и хост-номере. Класс адреса определяется значением его первого октета. Такое распределение обычно называется базовой или полноклассовой адресацией.



Сетевые адреса, являющиеся 32-разрядными числами, обычно записываются в виде четырех десятичных чисел, которые соответствуют отдельным байтам, разделенных точками. Например, шестнадцатеричный адрес C0290614 записывается как 192.41.6.20. Наименьший IP-адрес равен 0.0.0.0, а наибольший – 255.255.255.255.
Адреса класса A предназначены для использования в больших сетях общего пользования. Они допускают большое количество номеров узлов.
Адреса класса B используются в сетях среднего размера, например, сетях университетов и крупных компаний.
Адреса класса C используются в сетях с небольшим числом компьютеров.
Адреса класса D используются при обращениях к группам машин, а адреса класса E зарезервированы на будущее.

Людям удобнее называть машины по именам, а не числам. Например, у машины по имени alpha может быть IP-адрес 233.1.2.1. В маленьких сетях информация о соответствии имен IP-адресам хранится в файлах «hosts» на каждом узле. Конечно, название файла не зависит от конкретной реализации. В больших сетях из файла эта информация хранится на сервере и доступна только по сети.

IP-адрес	Имя узла (название машины)
223.1.2.1	alpha
223.1.2.2	beta
223.1.2.3	gamma
223.1.2.4	delta
223.1.3.2	epsilon
223.1.4.2	iota

IP-сети также могут иметь имена. Пример файла «networks»

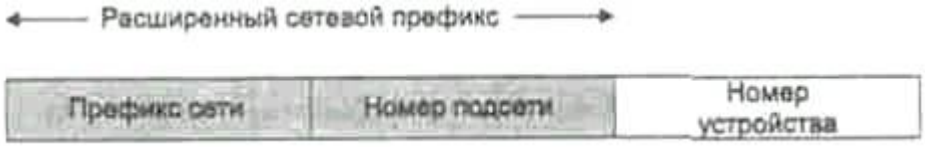
Сетевой номер	Имя сети
223.1.2	development
223.1.3	accounting
223.1.4	factory

Как известно, IP-адрес состоит из двух иерархических уровней. Необходимость во введении третьего уровня иерархии – уровня подсетей – была продиктована возникновением дефицита номеров сетей и резким ростом таблиц маршрутизации маршрутизаторов в Internet. После введения уровня подсети номер устройства разделяется на 2 части – номер подсети и номер устройства в этой подсети.

Маршрутизаторам в частной сети требуется различать отдельные подсети, но для маршрутизаторов Internet все подсети относятся к единственной записи в таблице маршрутизации. Это позволяет администратору частной сети вносить любые изменения в логическую структуру своей сети, не влияя на размер таблиц маршрутизации.
Кроме того, легко решается проблема выделения номеров при росте организации. Организация получает номер сети, а затем администратор произвольно присваивает номера подсетей для каждой внутренней сети. Это позволяет организации расширять свою сеть без необходимости получения еще одного сетевого номера.
Преимущества, которые обеспечивает формирование подсетей внутри частной сети:

- ❖Размер глобальных таблиц маршрутизации в сети Internet не растет
- ❖Администратор может по своему усмотрению создавать новые подсети без необходимости получения новых номеров сетей
- ❖Изменение топологии частной сети не влияет на таблицы маршрутизации в сети Internet, поскольку маршрутизаторы не имеют маршрутов в индивидуальные подсети организации – они хранят только маршрут с общим номером сети.

Если маршрутизаторы в сети Internet используют только сетевой префикс адреса получателя для передачи трафика в организацию, то маршрутизаторы внутри частной сети организации используют расширенный сетевой префикс для передачи трафика индивидуальным подсетям. Расширенным сетевым префиксом называют префикс сети и номер подсети.



Понятие расширенного сетевого префикса, по сути, эквивалентно понятию маска подсети. Маска подсети – это двоичное число, содержащее единицы в тех разрядах, которые относятся к расширенному сетевому префиксу. Маска подсети позволяет разделить IP-адрес на 2 части: номер подсети и номер устройства в этой подсети.
Старшие биты IP-адреса используются рабочими станциями и маршрутизаторами для определения класса адреса. После того как класс определен, устройство может легко вычислить границу между битами, использующимися для идентификации номера сети, и битами номера устройства в этой сети. Однако для определения границ битов, идентифицирующих номер подсети, такая схема не подходит. Для этого как раз используется 32-битная маска подсети, которая помогает однозначно определить требуемую границу. Для стандартных классов сетей маски имеют следующие значения:
255.0.0.0 – маска для сети класса A
255.255.0.0 – маска ля сети класса B
255.255.255.0 – маска для сети класса C

Как модуль IP узнает, какой именно сетевой интерфейс нужно использовать для отправления IP-пакета? Модуль IP осуществляет поиск в таблице маршрутов. Ключом поиска служит номер IP сети, выделенный из IP-адреса места назначения IP-пакета.
Таблица маршрутизаторов содержит по одной строке для каждого маршрута.
Основными столбцами таблицы маршрутов является номер сети, флаг прямой или косвенной маршрутизации, IP-адрес шлюза и номер сетевого интерфейса.
Эта таблица используется модулем IP при обработке каждого отправляемого IP-пакета.

В большинстве систем таблица маршрутов может быть изменена с помощью команды «route». Содержание таблицы маршрутов определяется менеджером сети, поскольку менеджер сети присваивает машинам IP-адреса.

Билет 5. Заголовок дейтаграммы IPv4. Прямая и косвенная IP-маршрутизация.

IP является базовым протоколом всего стека TCP/IP. Он отвечает за передачу информации по сети. Информация передается блоками, которые называются дейтаграммами.

IP является протоколом сетевого уровня. При этом для каждой среды передачи данных определен способ инкапсуляции IP-дейтаграмм. Маршрутизаторы пересылают инкапсулированные дейтаграммы по различным сетям, образуя объединение IP-сетей, по которому каждая рабочая станция может поддерживать связь по протоколу IP с любой другой рабочей станцией.

Протокол IP выполняет фрагментацию и сборку дейтаграмм, если принятый размер кадров в данной сети (или участке распределенной сети) отличается от размера исходных дейтаграмм. В протоколе IP отсутствуют механизмы повышения достоверности передачи данных, управления протоколом и синхронизации, которые обычно предоставляются в протоколах более высокого уровня. Протокол IP получает информацию для передачи от протоколов, расположенных по сравнению с ним на более высоком уровне. К этим протоколам, прежде всего относятся TCP и UDP. После получения информации от них протокол IP передает дейтаграммы через распределенную сеть, используя сервисы локальных сетей.

<table>
<tr><td>Номер версии (4 бита)</td><td colspan="2">Длина заголовка (4 бита)</td><td colspan="2">Тип сервиса (8 бит)</td><td>Общая длина(16 бит)</td></tr>
<tr><td colspan="2">Идентификатор (16 бит)</td><td colspan="2">Флаги (3 бита)</td><td colspan="2">Смещение фрагмента (13 бит)</td></tr>
<tr><td colspan="2">Время жизни (8 бит)</td><td colspan="2">Протокол (8 бит)</td><td colspan="2">Контрольная сумма заголовка (16 бит)</td></tr>
<tr><td colspan="6">Адрес отправителя (32 бита)</td></tr>
<tr><td colspan="6">Адрес получателя (32 бита)</td></tr>
<tr><td colspan="3">Опции (поле переменной длины)</td><td colspan="3">Выравнивание до 32-битной границы</td></tr>
</table>

Поле «Номер версии» указывает на версию используемого протокола IP (IPv4, IPv6).

Поле **«Длина заголовка»** определяет длину заголовка в 32-битовых словах. Заголовок может иметь минимальный размер 5 слов. При увеличении объема служебной информации эта длина может быть увеличена за счет поля «Опции».

Поле «Тип сервиса» определяет способ обслуживания дейтаграммы. Тип сервиса характеризует набор услуг, которые требуются от маршрутизаторов в распределенной сети. Эти параметры должны использоваться для управления выбором реальных рабочих характеристик при передаче дейтаграмм. В некоторых случаях передача дейтаграммы осуществляется с установкой приоритета, который дает данной дейтаграмме по сравнению с остальными некоторые преимущества при обработке. Тип сервиса определяется тремя показателями: малой задержкой при передаче, высокой достоверностью и большой пропускной способностью.

Поле «Время жизни». Его содержимое уменьшается на единицу при прохождении дейтаграммы через маршрутизатор, при обнулении поля дейтаграмма отбрасывается.

Поле **«Идентификатор»** используется для распознавания дейтаграмм, образованных в результате фрагментации. Все фрагменты фрагментированного пакета должны иметь одинаковое значение этого поля.

Поле «Общая длина» указывает общую длину дейтаграммы (заголовок и поле данных). Максимальный размер дейтаграммы может составлять 65535 байт.

Поле **«Флаги»** используется при фрагментации. Нулевой первый бит разрешает фрагментацию, единичный – запрещает. Единичный второй бит указывает на последний фрагмент дейтаграммы.

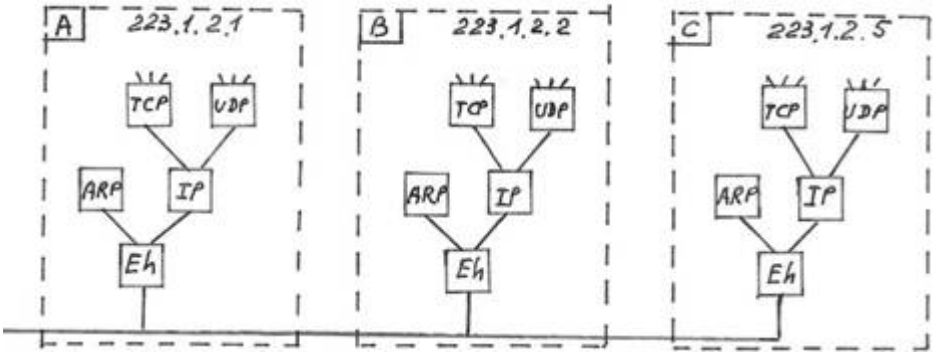
Поле «Смещение фрагмента» используется для указания смещения данных во фрагменте относительно начала исходной дейтаграммы. Чтобы получить смещение в байтах, надо значение этого поля умножить на 8. Первый фрагмент всегда имеет нулевое смещение.

Поле «Протокол» показывает, какому протоколу верхнего уровня принадлежит дейтаграмма. При поступлении дейтаграммы это поле указывает, какому приложению следует ее передать.

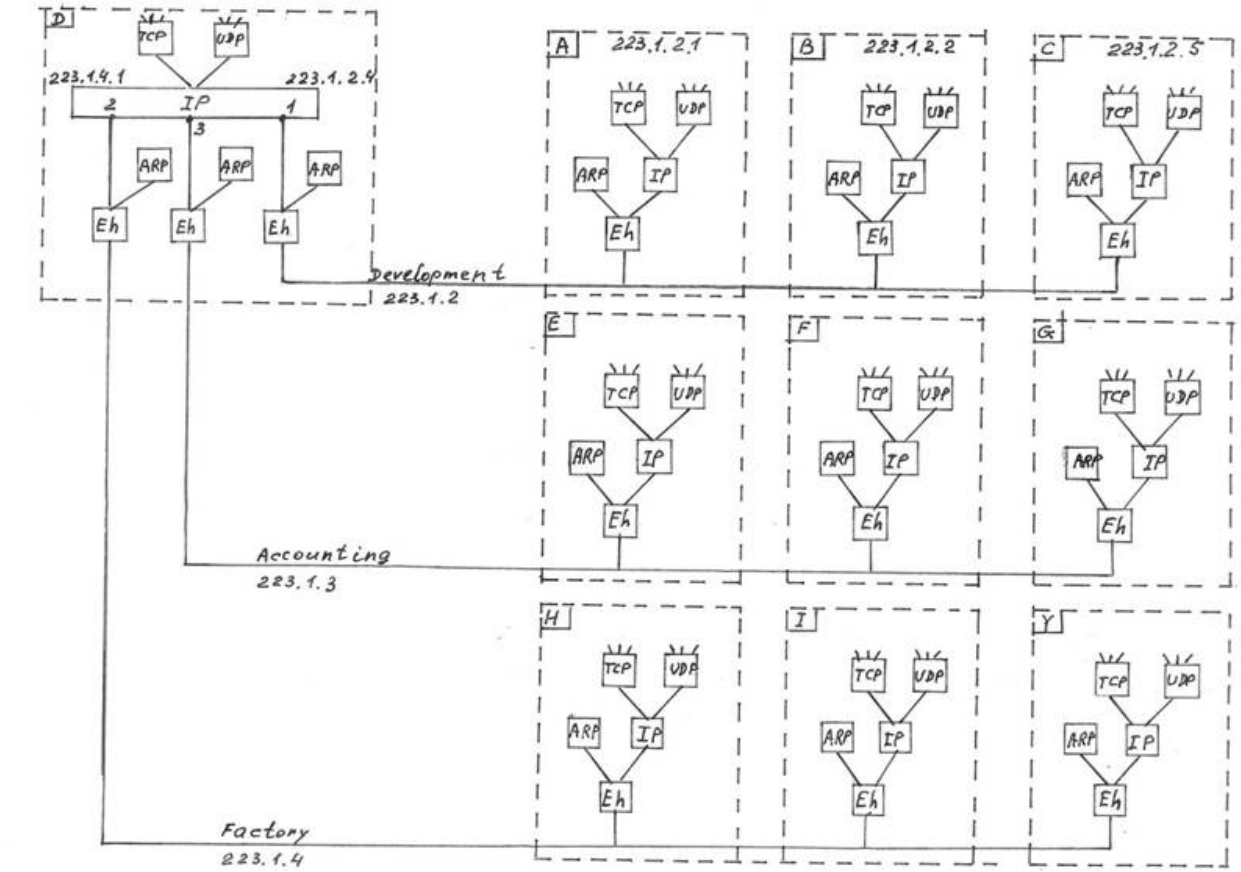
Поле «Контрольная сумма» рассчитывается по всему заголовку. Так как некоторые поля заголовка меняют свое значение, например время жизни, при прохождении дейтаграммы через

маршрутизаторы контрольная сумма проверяется и повторно рассчитывается при каждой модификации заголовка.
Поля «Адрес отправителя» и **«Адрес получателя»** имеют одинаковую длину и структуру. Поля содержат 32-битные IP-адреса отправителя и получателя дейтаграммы.
Поле «Опции» необязательно и обычно используется при настройке сети. В поле могут быть указаны точный маршрут прохождения дейтаграммы в распределенной сети, данные о безопасности, различные временные отметки и т. д. Поле не имеет фиксированной длины, поэтому для выравнивания заголовка дейтаграммы по 32-битной границе предусмотрено следующее поле – поле «Выравнивание». Выравнивание осуществляется нулями.

Прямая маршрутизация



На рисунке показана небольшая IP-сеть, состоящая из трех машин: A, B и C.
Каждая машина имеет стек протоколов TCP/IP. Каждый сетевой адаптер этих машин имеет свой Ethernet-адрес. Менеджер сети должен присвоить машинам уникальные IP-адреса.
Когда A посылает IP-пакет B, то заголовок IP-пакета содержит в поле отправителя IP-адрес узла A, а заголовок Ethernet-кадра содержит в поле отправителя Ethernet-адрес A. Кроме этого, IP-заголовок содержит в поле получателя IP-адрес узла B, а Ethernet-заголовок содержит в поле получателя Ethernet-адрес B.
В этом простом примере протокол IP является излишеством, которое мало что добавляет к услугам, предоставляемым сетью Ethernet. Однако протокол IP требует дополнительных расходов на создание, передачу и обработку IP-заголовка. Когда в машине B модуль IP получает IP-пакет от машины A, он сопоставляет IP-адреса места назначения со своим и, если адреса совпадают, то передает дейтаграмму протоколу верхнего уровня. В данном случае при взаимодействии A и B используется прямая маршрутизация.
Процедура прямой маршрутизации
Узел A посылает IP-пакет узлу B. Этот пакет находится в модуле IP узла alpha, и IP-адрес места назначение равен IP-адресу beta (223.1.2.2). Модуль IP с помощью маски подсети выделяет номер сети из IP-адреса и ищет соответствующую ему строку в таблице маршрутов. Остальная информация в найденной строке указывает на то, что машины в этой сети доступны напрямую через интерфейс номер 1. С помощью ARP-таблицы выполняется преобразование IP-адреса в соответствующий Ethernet-адрес, и через интерфейс 1 Ethernet-кадр посылается узлу B.
Если прикладная программа пытается послать данные по IP-адресу, который не принадлежит сети development, то модуль IP не сможет найти соответствующую запись в таблице маршрутов. В этом случае модуль IP отбрасывает IP-пакет. Некоторые реализации протокола возвращают сообщение об ошибке «Сеть не доступна».
Косвенная маршрутизация



В данном случае сеть Internet состоит из трех сетей Ethernet, на базе которых работаю три IP-сети, объединенные шлюзом D. Каждая IP-сеть включает четыре машины, каждая машина имеет свои собственные IP- и Ethernet-адреса.

Когда машина A посылает IP-пакет машине B, то процесс передачи идет в пределах одной сети. При всех взаимодействиях между машинами, подключенными к одной IP-сети, используется прямая маршрутизация.

Когда машина D взаимодействует с машинами A, E и H, то это прямое взаимодействие, поскольку каждая пара этих машин принадлежит одной IP-сети.

Однако, когда машина A взаимодействует с машинами, включенными в другую IP-сеть, то взаимодействие уже не будет прямым. Машина A должна использовать шлюз D для ретрансляции IP-пакетов в другую IP-сеть. Такое взаимодействие называется «косвенным».

Если машина A посылает машине E IP-пакет, то IP-адрес и Ethernet-адрес отправителя соответствуют адресам A. IP-адрес места назначения является адресом E, но поскольку модуль IP в A посылает IP-пакет через D, Ethernet-адрес места назначения является адресом D.

Процедура косвенной маршрутизации

Узел A посылает IP-пакет узлу E. Этот пакет находится в модуле IP узла A, и IP-адрес места назначения равен IP-адресу узла e (223.1.3.2). Модуль IP выделяет сетевой номер из IP-адреса (223.1.3) и ищет соответствующую ему строку в таблице маршрутов.

Запись в этой строке указывает на то, что машины требуемой сети доступны через шлюз D.

Модуль IP в узле A осуществляет поиск в ARP-таблице, с помощью которого определяет Ethernet-адрес, соответствующий IP-адресу D. Затем IP-пакет, содержащий IP-адрес места назначения E, посылается через интерфейс 1 шлюзу D.

IP-пакет принимается сетевым интерфейсом в узле D и передается модулю IP. Проверяется IP-адрес места назначения, и, поскольку он не соответствует ни одному из собственных IP-адресов D, шлюз решает ретранслировать IP-пакет.

Модуль IP в узле D выделяет сетевой номер из IP-адреса места назначения IP-пакета (223.1.3) и ищет соответствующую запись в таблице маршрутов.

Теперь модуль IP напрямую посылает IP-пакет узлу E через интерфейс номер 3. Пакет содержит IP- и Ethernet-адреса места назначения равные E.

Узел E принимает IP-пакет, и его модуль IP проверяет IP-адрес места назначения. Он соответствует IP-адресу E, поэтому содержащееся в IP-пакете сообщение передается протокольному модулю верхнего уровня.

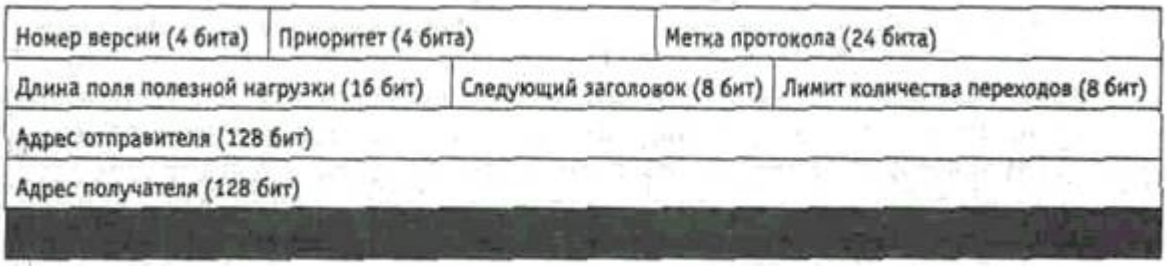
Билет 6. Заголовок дейтаграммы IPv6 и его отличие от заголовка IPv4.

Работа по расширению протокола IP была начата в 1992 году. Необходимость этого диктовалось тем, что практически все ресурсы старой версии протокола IP (IPv4) были исчерпаны. Быстрый рост сети Internet привел к появлению дефицита IP-адресов. Новая версия протокола IP – версия 6 (IPv6) – была принята организацией IETF в 1995 году. В настоящее время осуществляется постепенный переход к протоколу IPv6. Существует несколько фрагментов сети Internet, в которых маршрутизаторы поддерживают обе версии IP.

Основным механизмом, заложенным в схему адресации протокола IPv6, является введение иерархического разделения адресного пространства на уровни. Вместо прежних двух уровней – номера сети и номера устройства, – в протоколе IPv6 используется пять уровней, включая два уровня идентификации провайдеров и три уровня идентификации абонентов в сети.

Префикс	Идентификатор провайдера	Идентификатор абонента	Идентификатор подсети	Идентификатор узла

Для упрощения обработки заголовка дейтаграммы в протоколе IPv6 введены основной и дополнительный заголовки. Основной заголовок присутствует всегда и имеет длину 40 байт. Дополнительный заголовок определяет некоторые необязательные параметры.



Поле «Следующий заголовок» по своему назначению соответствует полю «Протокол» в версии 4 и определяет тип заголовка, который следует за данными. Каждый следующий дополнительный заголовок содержит это поле. Если дейтаграмма не содержит дополнительных заголовков, то значение этого поля определяет протокол – TCP, UDP, RIP или OSPF.

Поле «Лимит переходов» введено для более эффективного определения времени жизни дейтаграмм. В протоколе IPv4 поле времени жизни дейтаграммы уменьшается на единицу (по крайней мере) при прохождении каждого маршрутизатора. При этом время ожидания в очереди не учитывается.

Поле «Приоритет» позволяет отправителю задать приоритет своих дейтаграмм. Возможные 16 значений этого поля разделены на две категории: значения от 0 до 7 определяют трафик, которым маршрутизатор при необходимости может пренебречь, а значения от 8 до 15 указывают на трафик, к которому эти меры применяться не могут (аудио- и видеоинформация, передаваемая с постоянной скоростью в реальном времени).

Используя поля «Приоритет» и «Метка потока» устройства могут идентифицировать дейтаграммы, которым требуется нестандартное обслуживание на маршрутизаторах. В протоколе IPv6 определены следующие типы дополнительных заголовков:

Routing – определяет полный маршрут при маршрутизации от источника. Данный заголовок позволяет отправителю указать список IP-адресов, которые диктуют путь передачи;

Fragmentation – содержит сведения о проведении фрагментации на конечных узлах сети. В протоколе IPv6 фрагментацию не разрешается выполнять на промежуточных узлах; это значительно повышает производительность при маршрутизации;

Authentication – служит для идентификации конечных узлов и обеспечения целостности дейтаграмм;

Encryption – служит для шифрования и дешифровки передаваемых данных;

Hop-by-Hop Option – используется алгоритмом «переход за переходом» при обработке дейтаграмм;

Destination Option – содержит дополнительную информацию для узла назначения.
Каждый дополнительный заголовок содержит тип следующего за ним заголовка, что позволяет создать цепочку заголовков. Основной заголовок является первым в цепочке и не содержит дополнительных заголовков.
Поле «Следующий заголовок» указывает, какой дополнительный заголовок следует за основным. Поле «Следующий заголовок» первого дополнительного заголовка указывает на тип второго дополнительного заголовка и т. д.
Для поддержки качества обслуживания протокол IPv6 работает с «меткой потока» (flow label). *Метка потока* – это признак, который размещается в поле заголовка «Метка протокола» дейтаграммы IPv6. Метка указывает на принадлежность данной дейтаграммы к последовательности дейтаграмм – потоку, который требует определенных параметров обслуживания. Маршрутизаторы обрабатывают потоки на основании значения метки и идентификатора отправителя дейтаграмм.
Отличие от заголовка IPv4
В отличие от протокола IPv4, механизмы разрешения адресов и их настройки в протоколе IPv6 реализованы на сетевом уровне, а не на канальном (протоколы ARP для широковещательных сетей и ATMARP и NHRP для сетей ATM). То есть, протокол обнаружения (NeighborDiscovery), который используется для нахождения маршрутизаторов и соседей, является интегральной частью IPv6 и любые механизмы, предназначенные для адаптации технологии ATM к IPv6, прежде всего, должны работать именно с этим протоколом. Здесь наблюдается принципиальное отличие от протокола IPv4, в котором механизмы разрешения адресов не являются частью основного протокола, а реализуются на канальном уровне – протоколом ARP для широковещательных сетей и механизмами ATMARP и NHRP для сетей ATM.

Билет 7. Протокол RIP и его отличие от OSPF.
Протокол RIP относится к классу протоколов IGP. Этот протокол является одним из первых протоколов обмена маршрутной информацией между маршрутизаторами в IP сети.
Преимуществом протокола RIP является его простота. *Недостатком* — увеличение трафика за счет периодической рассылки широковещательных сообщений.
Протокол RIP стал стандартным протоколом маршрутизации внутри отдельной автономной системы (АС), хотя он существенно ограничивает размер автономной системы. Это связано с тем, что протокол RIP не поддерживает длинные пути, которые содержат более 15 переходов.
Протокол RIP использует алгоритм длины вектора. То есть, каждый маршрутизатор выбирает ближайший соседний маршрутизатор, который расположен на этом самом коротком маршруте до получателя. Выбор осуществляется на основании информации о стоимости путей (выбирается путь с меньшей стоимостью).
Стоимость вычисляется по информации, имеющейся в таблицах маршрутизации всех соседних маршрутизаторов (маршрутизаторы регулярно обмениваются между собой таблицами маршрутизации). Этот алгоритм хорошо работает в небольших сетях. В больших сетях он заполняет сеть широковещательным трафиком.
Протокол не всегда точно и быстро учитывает изменения сетевой топологии, так как маршрутизаторы не имеют точного представления о топологии сети, а располагают только информацией, полученной от своих соседей.
Протокол RIP использует в качестве метрики маршрута количество переходов, то есть число маршрутизаторов, которые должна миновать дейтаграмма, прежде чем она достигнет получателя. Маршрутизаторы с поддержкой протокола RIP всегда выбирают маршрут с наименьшим числом переходов.
Таблица маршрутизации RIP содержит следующие поля:
- ❖IP-адрес целевой сети
- ❖Количество переходов до целевой сети
- ❖Адрес первого маршрутизатора на пути к целевой сети
- ❖Идентификатор соседнего маршрутизатора, который является источником данной адресной информации в таблице маршрутизации
- ❖Таймер для отслеживания времени, прошедшего с момента последнего обновления записи

При включении маршрутизатора таблица маршрутизации заполняется описанием сетей, которые напрямую подключены к данному маршрутизатору.

Затем соседние маршрутизаторы шлют ему свою информацию. Периодически каждый маршрутизатор посылает сообщения об обновлении маршрута всем своим соседям. Эти сообщения содержат информацию из таблицы маршрутизации.
Если объем информации слишком велик и не помещается в одно сообщение протокола RIP, она будет разделена на части и помещена в несколько сообщений. Перед передачей сообщений об обновлении маршрута соседним маршрутизаторам маршрутизатор увеличивает количество переходов до получателя на единицу.
Когда сообщения об обновлении маршрута приходят на маршрутизатор, он обновляет свою таблицу маршрутизации в соответствии со следующими *правилами*:

- ❖Если новое количество переходов меньше, чем текущее (для конкретной записи), маршрутизатор примет новый маршрут. Эта новая запись будет храниться в таблице до тех пор, пока не появится маршрут с еще меньшей метрикой;
- ❖Если передающий маршрутизатор является источником информации для существующей записи, то принявший сообщение маршрутизатор будет использовать новое значение количества переходов, даже если оно больше, чем старое.

Реакция протокола RIP на изменения в топологии сети зависит от того, как маршрутизатор информирует своих соседей о модификации его таблицы маршрутизации.
Однако следует учитывать, что если сетевая топология изменяется, то соседи могут перестать быть соседями. Кроме того, если маршрутизатор выходит из строя, он не может известить своих соседей об изменениях в топологии.
Таким образом, в случае отсутствия сообщений об обновлении маршрутизации предполагаемый маршрут может не отражать произошедшие изменения.
Все маршрутизаторы, участвующие в обмене сообщениями об обновлении маршрута по протоколу RIP, посылают эти сообщения через определенный интервал, который по умолчанию составляет 30 с.
Если маршрутизатор не получает сообщения от маршрутизатора, который отвечает за определенную запись в таблице маршрутизации за временной интервал, равный увеличенному в шесть раз интервалу обмена (по умолчанию 180 с), он предполагает, что либо его соседний маршрутизатор вышел из строя, либо между ними нарушилась связь.
Затем маршрутизатор помечает отказавший маршрут как некорректный и, в конечном счете, удаляет его из своей таблицы маршрутизации.
Когда маршрутизатор получит информацию о новом маршруте от другого своего соседа, он будет использоваться вместо старого удаленного маршрута. Шестикратное увеличение интервала необходимо для того, чтобы избежать исключения маршрута при случайной потере одного сообщения об обновлении.
Кроме того, маршрутизатору чрезвычайно важно оповестить своих соседей о том, что не существует корректного маршрута к определенному получателю. Протокол RIP позволяет выполнить это с помощью стандартных сообщений об обновлении маршрутизации.
Для обозначения недостижимости получателя количество переходов устанавливается равным 16. Это значение можно считать «бесконечностью», так как допустимые маршруты не могут иметь более 15 переходов. Если какая-либо сеть становится недостижимой, все соседние маршрутизаторы установят метрику для этой сети равной 16.

Так как протокол RIP был разработан достаточно давно и практически не изменялся за это время, он обладает существенными недостатками, которые ограничивают его применение в сложных сетях:

- ❖RIP допускает 15 переходов между отправителем и получателем. Если количество переходов превышает 15, получатель рассматривается как недостижимый, что очень сильно ограничивает размеры автономной системы.
- ❖Использование в качестве метрики маршрута количества переходов приводит к тому, что протокол RIP не всегда выбирает самый эффективный и экономный маршрут. Например, может оказаться так, что выбранный маршрут содержит медленный и дорогой канал связи. Протоколы маршрутизации, которые используют в качестве метрики количество переходов, не принимают во внимание такие важные характеристики, как скорость канала, его надежность и т. д.

❖Механизм обмена сообщениями RIP основан на широковещательной рассылке всей таблицы маршрутизации. В средних и больших сетях это может сильно загрузить каналы связи, особенно если распределенная сеть состоит из удаленных сетей, соединяемых по коммутируемым каналам связи.

❖Так как протокол RIP работает по алгоритму вектора расстояния, он обладает медленной сходимостью. В случае, если канал связи выйдет из строя, потребуется несколько минут для того, чтобы маршрутизаторы скорректировали свои таблицы. В течение этого времени могут образоваться петли маршрутизации.

Отличие от протокола OSPF

Предположим, что станции А необходимо передать данные станции Б и администратор вручную настроил метрики для протокола OSPF. Протокол RIP не учитывает при выборе маршрута скорость каналов связи. Поэтому при использовании протокола RIP данные будут передаваться через низкоскоростной (56 Кбит/с) канал связи, так как он имеет наименьшее количество переходов. В отличие от протокола RIP протокол OSPF принимает во внимание скорость канала связи (если она выбрана в качестве метрики). Поэтому будут задействованы именно высокоскоростные каналы, так как их суммарная стоимость составит 20.

Билет 8. Протокол OSPF и его отличие от протокола RIP.

Спецификация протокола OSPF (Open Shortest Path First) описана в документе RFC 1247. Протокол принят в 1991 году. Он ориентирован на применение в больших распределенных сетях. OSPF вычисляет маршруты в сетях IP, работая совместно с другими протоколами обмена маршрутной информацией. Протокол OSPF основан на алгоритме состояния канала. Суть этого алгоритма состоит в том, что он должен вычислить кратчайший путь. При этом «кратчайший» не означает, что путь физически самый короткий. Имеется в виду, что информация пройдет по этому пути быстрее, чем по другим. Маршрутизатор, работающий с этим протоколом, отправляет запросы всем соседним маршрутизаторам, находящимся в одном с ним домене маршрутизации, для выявления состояния каналов до них и далее от них. Состояние канала при этом характеризуется несколькими параметрами, которые называются *метриками.* Метрикой может быть пропускная способность канала, его загрузка на текущий момент, задержка информации при ее прохождении по этому каналу и т. д. Обобщив полученные сведения, этот маршрутизатор сообщает их всем соседям. После этого им строится ориентированный граф, который повторяет топологию домена маршрутизации. Каждому ребру этого графа назначается оценочный параметр (метрика). После построения графа используется алгоритм Дейкстры, который по двум заданным узлам находит набор ребер с наименьшей суммарной стоимостью, то есть, по сути, выбирает оптимальный маршрут. По совокупной информации (полученной и найденной в результате вычислений) создается таблица маршрутизации.

Протокол OSPF отвечает за IP-маршрутизацию и относится к классу протоколов IGP. Он может заменить протокол маршрутизации RIP в больших и сложных сетях.

Протокол OSPF решает многие из перечисленных выше проблем, присущих протоколу RIP. Кроме того:

❖Протокол OSPF обладает большей скоростью сходимости, что предотвращает возникновение петель маршрутизации

❖При работе протокола не генерируется большой сетевой трафик

❖Информация, которой обмениваются маршрутизаторы, проходит процедуру аутентификации. Это позволяет участвовать в процессе маршрутизации только авторизованным маршрутизаторам. Аутентификация устраняет возможность случайного или преднамеренного искажения маршрутной информации

❖Протокол OSPF использует групповую передачу вместо широковещательной. Такой подход позволяет исключить передачу маршрутной информации тем устройствам, которым она не нужна

❖Протокол поддерживает распределение нагрузки (load balancing) по маршрутам, имеющим одинаковые стоимости

❖Протокол поддерживает маски подсетей переменной длины (VLSM). Данные о масках подсетей передаются в сообщениях LSA (Link-State Advertisement, – объявление о состоянии канала), таким образом, маршрутизаторы получают эту информацию динамически, что позволяет более гибко использовать выделенное организации адресное пространство

❖Протокол поддерживает области маршрутизации. Это позволяет администраторам разделять автономную систему на отдельные части с изоляцией сетевого трафика и маршрутизации

❖Протокол OSPF поддерживает передачу внешней маршрутной информации через автономные системы

Протокол OSPF хорошо подходит для современных, больших, динамически изменяющихся сетей.

Отличие от протокола RIP

В отличие от протокола RIP, который по умолчанию рассылает всю свою таблицу маршрутизации каждые 30 с, протокол OSPF посылает информацию о состоянии каналов каждые 30 мин.

При обнаружении изменения сетевой топологии протокол OSPF сразу же посылает небольшие (порядка 75 байт) сообщения.

В начале работы каждый маршрутизатор передает сообщения LSA на все свои порты. Эти сообщения позволяют однозначно идентифицировать передающий маршрутизатор и определить состояние всех его портов: IP-адрес, маску подсети, метрику, присвоенную каналу связи порта, и статус канала связи.

Хотя протокол OSPF решает все проблемы, присущие RIP, он тоже далек от идеала. В очень больших сетях протокол OSPF порождает чрезвычайно много маршрутной информации. Например, в распределенной сети с сотнями маршрутизаторов изменение состояния одного канала (скажем, обрыв линии связи) вызывает распространение тысяч сообщений LSA через всю сеть. База данных состояния канала в таких сетях может достигать нескольких мегабайт.

Билет 9. Заголовок и протокол UDP. Отличие протокола UDP от протокола TCP.

User Datagram Protocol - Протокол Пользовательских Дейтаграмм.

Протокол разработан для предоставления прикладным программам транспортных услуг. UDP, так же, как и IP, обеспечивает негарантированную доставку дейтаграмм получателю и не поддерживает установку соединений. В модели стека TCP/IP он располагается над протоколом IP.

Взаимодействие между прикладными программами и протоколом UDP осуществляется через так называемые *протокольные порты.* Протокольные порты определяют соответствие между абстрактными точками доступа к протоколу UDP и конкретными прикладными программами. Механизм протокольных портов позволяет рабочей станции одновременно поддерживать несколько сеансов связи с удаленными компьютерами и программами в сети. Можно также сказать, что протокольный порт служит для указания программы-получателя информации. Когда рабочая станция получает дейтаграмму с ее IP-адресом, она направляет эту дейтаграмму конкретной программе, используя номер протокольного порта, который определяется во время установки сеанса связи.

Для связи с протокольным портом на другой рабочей станции, отправитель должен знать IP-адрес получателя и номер порта на этой рабочей станции. Каждое сообщение содержит также номер протокольного порта отправляющей рабочей станции. Таким образом, прикладная программа, получающая сообщения, может напрямую ответить отправителю.

Каждое сообщение протокола UDP называется *пользовательской дейтаграммой.* Она состоит из двух частей: заголовка и области данных. Заголовок содержит четыре 16-битных поля, которые определяют протокольный порт отправителя, протокольный порт получателя, длину сообщения и контрольную сумму.

Порт отправителя (16 бит)	Порт получателя (16 бит)
Длина сообщения (16 бит)	Контрольная сумма (16 бит)
Данные	

Поля «Порт отправителя» и **«Порт получателя»** содержат 16-битные номера портов. **Поле «Порт отправителя»** может не использоваться, тогда оно должно содержать нули. **Поле «Длина сообщения»** указывает количество байтов в пользовательской дейтаграмме. При этом учитывается длина заголовка протокола UDP и длина поля данных.

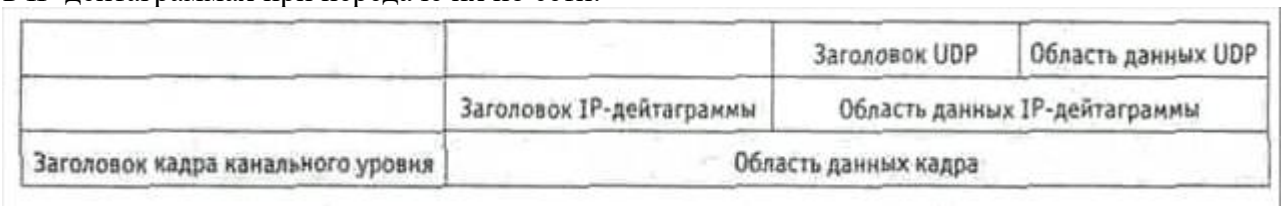
Контрольная сумма пользовательской дейтаграммы может вычисляться, а может и не вычисляться. Нулевое поле «Контрольная сумма» означает, что контрольная сумма не вычислялась. Контрольная сумма, как правило, не вычисляется при работе протокола UDP в высоконадежной локальной сети. Однако в ненадежной сети только контрольная сумма может указать на достоверность и целостность пришедших данных — ведь протокол IP не вычисляет контрольную сумму поля данных IP-дейтаграмм.

Для расчета контрольной суммы пользовательской дейтаграммы необходима дополнительная информация. Для этой цели к началу пользовательской дейтаграммы приписывают псевдозаголовок и добавляют в конец этой дейтаграммы байт, заполненный нулями, так, чтобы число бит во всем сообщении было кратно 16. После этого вычисляется контрольная сумма полученной дейтаграммы. Концевое дополнение из нулей и псевдозаголовок не передаются вместе с пользовательской дейтаграммой. Для вычисления контрольной суммы полученной пользовательской дейтаграммы сначала сохраняется ноль в поле «Контрольная сумма», затем вычисляется 16-битная сумма, включая псевдозаголовок, заголовок самой дейтаграммы и данных. При получении пользовательской дейтаграммы контрольная сумма должна проверяться.

Формат псевдозаголовка

IP-здрес отправителя (32 бита)		
IP-адрес получателя (32 бита)		
Нули (8 бит)	Протокол (8 бит)	Длина UDP (16 бит)

Псевдозаголовок имеет длину 12 байт. **Поле «Протокол»** содержит код типа протокола. Для проверки контрольной суммы получатель должен извлечь эти поля из IP-заголовка, сформировать свой псевдозаголовок и вычислить контрольную сумму. Данные протокола UDP инкапсулируются в IP-дейтаграммах при передаче их по сети.



Если поступило сообщение с номером протокольного порта, которого нет среди используемых протокольных портов, оно удаляется и высылается сообщение протокола ICMP «Получатель недостижим» с кодом «Порт недостижим».

Отличие протокола UDP от протокола TCP

❖*Отсутствие сигналов квитирования.* Перед отправкой пакета UDP, отправляющая сторона не обменивается с получающей стороной квитирующими сигналами. Следовательно, у отправителя нет способа узнать, достигла ли дейтаграмма конечной системы. В результате UDP не может гарантировать, что данные будут действительно доставлены адресату (например, если не работает конечная система или сеть).

❖*Использование сессий.* Ориентированность TCP на соединения поддерживается сеансами между хостами. TCP использует идентификатор сеанса, позволяющий отслеживать соединения между двумя хостами. UDP не имеет поддержки сеансов из-за своей природы, не ориентированной на соединения.

❖*Надежность.* UDP не гарантирует, что адресату будет доставлена только одна копия данных. Чтобы отправить конечной системе большой объем данных, UDP разбивает его на небольшие части. UDP не гарантирует, что эти части будут доставлены по назначению в том же порядке, в каком они создавались в источнике. Напротив, TCP вместе с номерами портов использует порядковые номера и регулярно отправляемые подтверждения, гарантирующие упорядоченную доставку данных.

❖*Безопасность.* TCP более защищен, чем UDP. Во многих организациях брандмауэры и маршрутизаторы не пропускают пакеты UDP. Это связано с тем, что хакеры могут воспользоваться портами UDP, не устанавливая явных соединений.

❖*Управление потоком.* В UDP управление потоком отсутствует, в результате плохо спроектированное UDP-приложение может захватить значительную часть пропускной способности сети.

❖*Нет установки соединения.* UDP является протоколом без организации соединений, поэтому он освобождает от накладных расходов, связанных с установкой соединений. Поскольку UDP не пользуется сигналами квитирования, то задержек, вызванных установкой соединений, также удается избежать. Именно поэтому DNS отдает предпочтение UDP перед TCP — DNS работала бы гораздо медленнее, если бы она выполнялась через TCP.

❖*Скорость.* UDP работает быстрее TCP. По этой причине многие приложения предпочитают не TCP, а UDP. Те же средства, которые делают TCP более устойчивым (например, сигналы квитирования), замедляют его работу.

❖*Топологическое разнообразие.* UDP поддерживает взаимодействия "один с одним" и "один со многими", в то время как TCP поддерживает лишь взаимодействие "один с одним".

❖*Накладные расходы.* Работа с TCP означает повышенные накладные расходы, издержки, налагаемые UDP, существенно ниже. TCP по сравнению с UDP использует значительно больше ресурсов операционной системы, и, как следствие, в таких средах, где серверы одновременно обслуживают многих клиентов, широко используют UDP.

❖*Размер заголовка.* Для каждого пакета заголовок UDP имеет длину всего лишь восемь байтов, в то время как TCP имеет 20-байтовые заголовки, и поэтому UDP потребляет меньше пропускной способности сети.

Билет 10. Протокол TCP. Формат заголовка TCP. Установление соединения. Передача данных.

Transmission Control Protocol (TCP, протокол управления передачей) — один из основных протоколов передачи данных интернета, предназначенный для управления передачей данных.

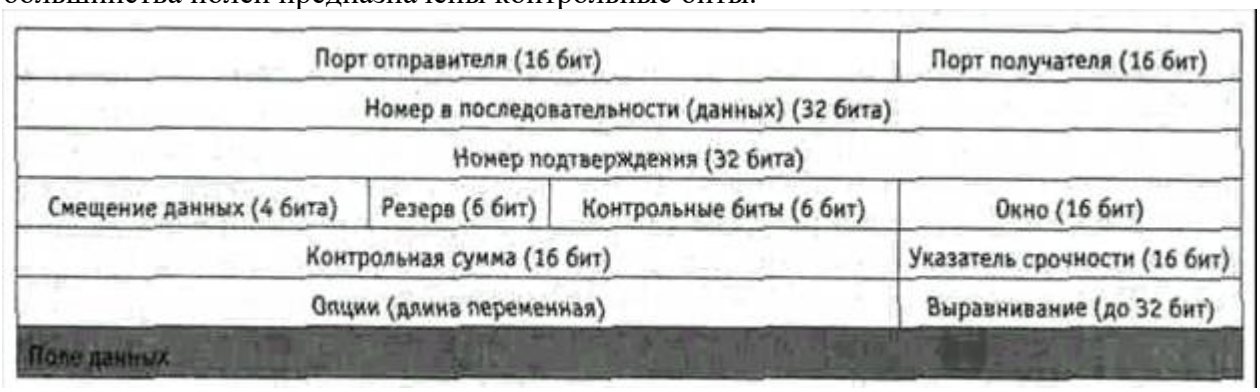
Поступающие к получателю данные буферизуются средствами протокола TCP. Перед отправкой данные также буферизуются.

Формат заголовка TCP

Протокол TCP — это основной транспортный протокол в стеке протоколов TCP/IP. Он обеспечивает надежную передачу потока данных, опираясь при этом на ненадежный сервис транспортировки дейтаграмм, предоставляемый протоколом IP. В сетях IP протокол TCP используется для обработки запросов на вход в сеть, разделения ресурсов (файлов и принтеров), репликации информации между контролерами доменов, передачи списков ресурсов и т. д. На протокол TCP, в частности, возложена задача управления потоками и перегрузками. Он отвечает за согласование скорости передачи данных с техническими возможностями рабочей станции-получателя и промежуточных устройств в сети.

Этот протокол использует только один тип протокольного блока данных (PDU — Protocol Data Unit), который называется *сегментом TCP.* Сегмент состоит из заголовка и поля данных (полезной нагрузки).

Минимальная длина составляет 20 байт. Такая большая величина обусловлена тем, что один и тот же заголовок используется протоколом для самых различных целей. Для определения назначения большинства полей предназначены контрольные биты.



Поле «Номер в последовательности» (Sequence Number) определяет номер первого байта в очереди (последовательности) байтов в текущем сегменте. Исключение составляют случаи, когда установлен бит (флаг) синхронизации SYN. Тогда это поле обозначает начальный номер в последовательности (Initial Sequence Number, ISN), и первый байт данных имеет номер в очереди ISN+1.

Поле «Номер подтверждения» (Acknowledgment Number) содержит следующий номер в последовательности получаемых подтверждений, который ожидает отправитель в ответ на отосланный сегмент. В заголовке в поле «Номер в последовательности» занесен указанный номер подтверждения. При этом должен быть установлен контрольный бит подтверждения АСК. Подтверждения (или, как часто говорят, АСК) посылаются постоянно, как только соединение будет установлено.

Поле «Смещение данных» (Data Offset) определяет количество 32-битных слов в заголовке TCP. Тем самым указывается начало поля данных. Заголовок протокола TCP всегда заканчивается на 32-битной границе, даже если он содержит опции.

Поле «Резерв» (Reserved) должно быть заполнено нулями и предназначено для будущего расширения протокола.

Поле «Окно» (Window) содержит объявляемый размер окна в байтах.

Поле «Контрольная сумма» (Checksum) рассчитывается по сегменту, при этом определяется 16-битное дополнение суммы всех 16-битных слов в заголовке и в поле данных. Если сегмент содержит нечетное количество байтов, то он будет дополнен нулями справа до образования 16-битного слова. Этот выравнивающий байт не передается с сегментом по сети, так как может быть «восстановлен» получателем. Контрольная сумма учитывает также 96-битный псевдозаголовок, который ставится перед заголовком протокола TCP. Псевдозаголовок включает следующие поля из заголовка протокола IP: IP-адреса отправителя и получателя, протокол и длину сегмента. С помощью добавления псевдозаголовка протокол TCP защищает самого себя от ошибочной доставки протоколом IP. Так, если протокол IP доставляет сегмент, не предназначенный данному работающему приложению, то модуль протокола TCP на принимающей стороне обнаружит некорректность доставки.

Поле «Указатель срочности» (Urgent Pointer) сообщает текущее значение указателя срочности. Эта положительная величина определяет смещение относительно номера в очереди данного сегмента. Этот указатель сообщает номер байта, следующего за срочными данными, то есть, начиная с этого байта, данные имеют обычный статус срочности. Поле используется совместно с контрольным битом URG.

Поле «Опции» (Options) имеет переменную длину и может вообще отсутствовать. Опции располагаются в конце заголовка протокола TCP и их длина кратна 8 битам. Протокол TCP должен быть готов обрабатывать все виды опций. Опции используются для решения вспомогательных задач, например, для выбора максимального размера сегмента.

Поле «Выравнивание» (Padding) может иметь переменную длину и представляет собой фиктивное поле, используемое для доведения размера заголовка до целого числа 32-битных слов.

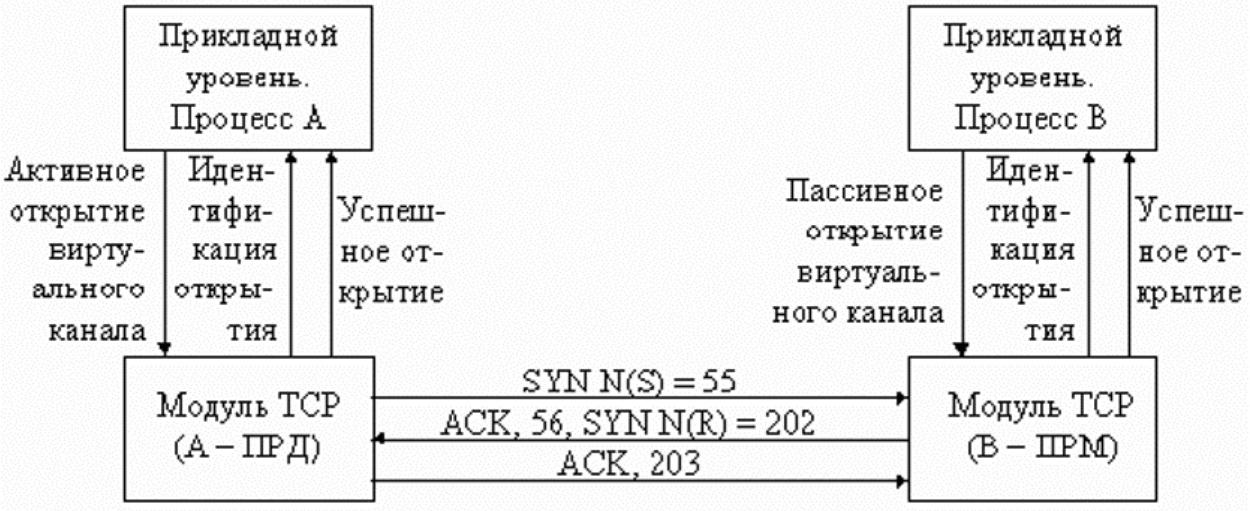
Установление соединения

Состояние системы. Соединения протокола TCP переходят из одного состояния в другое в ответ на определенные события — запросы клиента, приход сегментов с флагами SYN, ACK, RST, FIN —

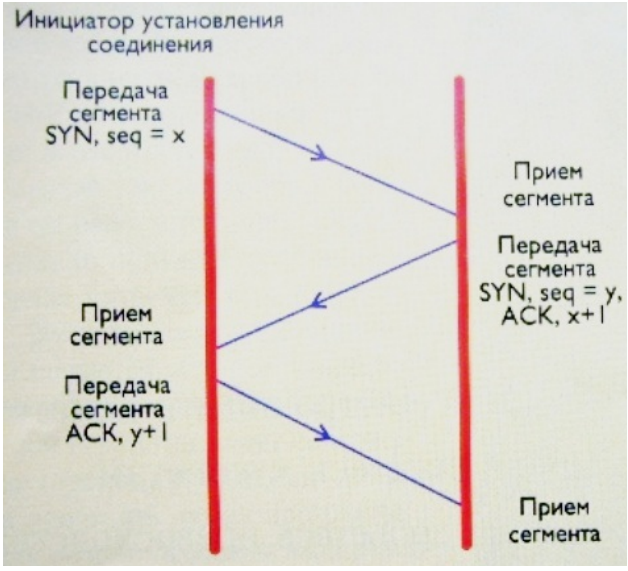
или по истечении заданного времени. Соединение может находиться в одном из следующих состояний:

- ❖**LISTEN** — ожидание запроса на соединение со стороны внешних (чужих) портов и внешних (чужих) программ TCP.
- ❖**SYN-SENT** — ожидание парного запроса на установление соединения (со стороны отправителя запрос уже сделан).
- ❖**SYN-RECEIVED** — ожидание подтверждения после того, как запрос на установление соединения уже принят и отправлен.
- ❖**ESTABLISHED** — соединение установлено. Принимаемые от приложения данные можно передать пользователю.
- ❖**FIN-WAIT-1** — ожидание запроса от чужой программы TCP или подтверждение ранее отправленного запроса на закрытие соединения.
- ❖**FIN-WAIT-2** — ожидание запроса на закрытие соединения со стороны чужой программы TCP.
- ❖**CLOSE-WAIT** — ожидание запроса на закрытие соединения со стороны своего клиента.
- ❖**CLOSING** — ожидание подтверждения запроса о закрытии соединения со стороны чужой программы TCP.
- ❖**LAST-ACK** — ожидание ответного запроса на закрытие соединения на посланный запрос о закрытии соединения, который был ранее отправлен чужой программе TCP.
- ❖**TIME-WATT** — соединение находится в этом состоянии на протяжении времени, достаточного для того, чтобы быть уверенным, что чужая программа TCP получила подтверждение своего запроса на закрытие соединения.
- ❖**CLOSED** — соединение закрыто.
- ❖Основное состояние соединения — ESTABLISHED (соединение установлено). В этом состоянии происходит обмен данными между абонентами.

Фаза 1 — установление соединения.



N(S) - номер байта, с которого начнёт передавать передатчик (ПРД), например, 55;
N(R) - номер байта, с которого будет передавать приёмник (ПРМ), например, 202.



Для открытия виртуального соединения посылается флаг SYN в сегменте, с которого начнется передача (N(S)=55). Приёмник отвечает сегменту, в котором флаг ACK установлен в 1 и указывает номер байта, с которого он начнёт передавать (N(R)=202). В заголовке этого же сегмента в поле «Номер подтверждения» приёмник указывает, что он ожидает от передатчика байт с номером 56. Здесь же передаётся флаг синхронизации SYN. Передатчик (модуль А), получив этот сегмент с подтверждением о готовности приёмника работать, также отвечает сегментом с подтверждением ACK, и в поле «Номер подтверждения» передатчик указывает, что он ожидает от приёмника байт с номером 203.

После этого виртуальное соединение установлено, о чем модули TCP извещают свои прикладные процессы.

Передача данных

Блок управления передачей. Для обеспечения надежной передачи данных по установленным логическим соединениям между прикладными программами протокол TCP должен обеспечивать следующие функции:

- ❖передачу данных
- ❖проверку достоверности данных при передаче
- ❖управление потоком данных
- ❖разделение каналов связи
- ❖обслуживание установленных соединений
- ❖соблюдение установленного приоритета пользователей
- ❖обеспечение соответствующего уровня безопасности

При передаче данных по соединению каждый байт информации нумеруется. Нумерация ведется и в очереди отправления и в очереди получения. Первоначальный номер байта в очереди отправки указывается модулем TCP посылающей стороны, а первоначальный номер байта в очереди приема выясняется во время установления соединения. В это время оба модуля протокола TCP должны синхронизировать друг с другом первоначальные номера байтов в очередях. Синхронизация производится путем обмена сегментами, которые используются при установке соединения. Эти сегменты несут флаг синхронизации SYN и исходные номера для обеих очередей. Синхронизация требует, чтобы каждая сторона послала свой собственный первоначальный о принятии этого номера. Нумеруются и сами сегменты: номером сегмента считается номер первого байта в поле полезной нагрузки этого сегмента.

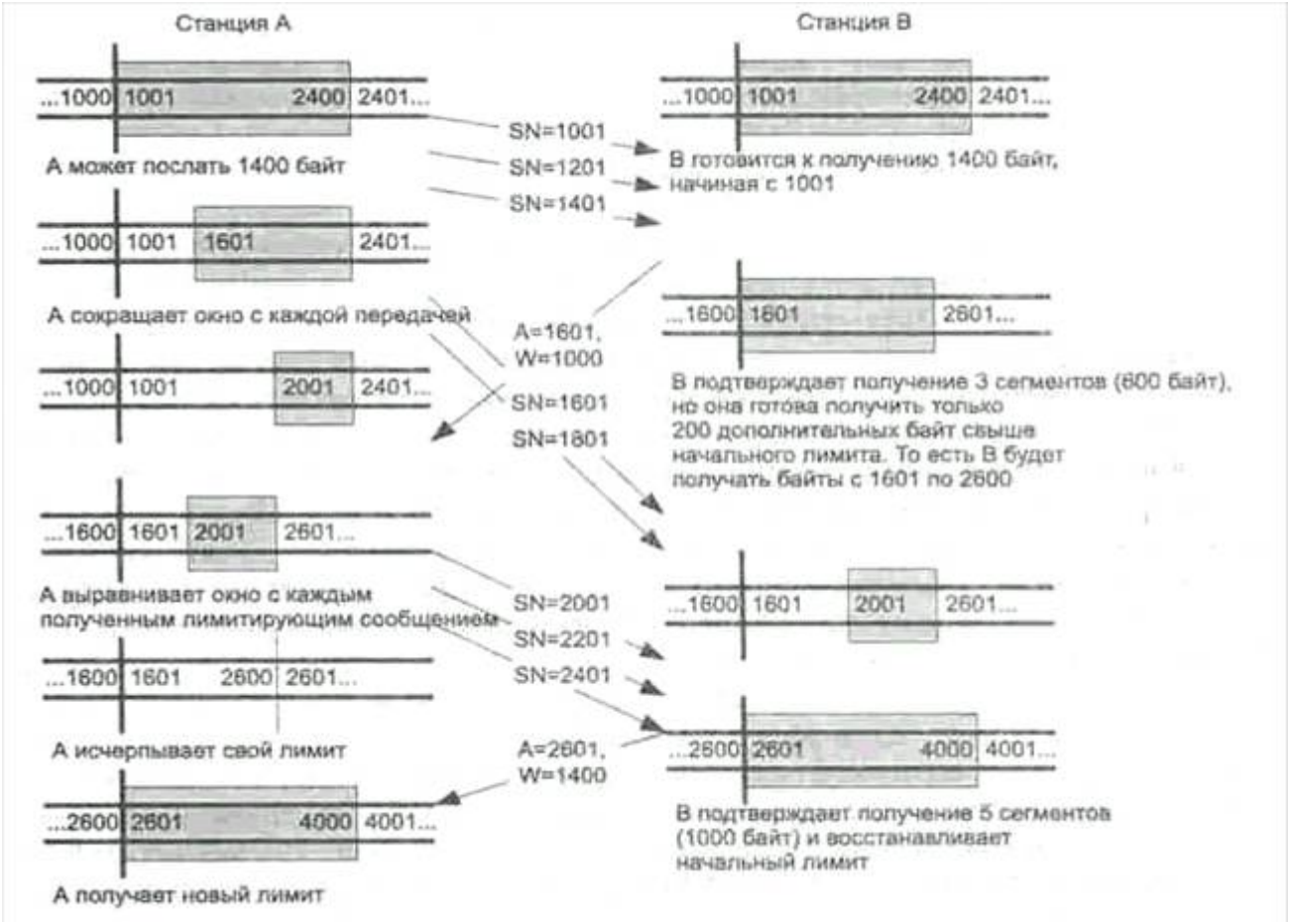
Билет 11. Протокол TCP. Механизм окна TCP. Управление потоком данных.

Плавающее окно

Как и большинство протоколов, осуществляющих управление потоком данных, протокол TCP использует механизм плавающих окон. Протокол TCP несколько отходит от классической схемы.

Основной *недостаток* классической схемы заключается в том, что только один сегмент может быть передан за один сеанс. В данном случае под сеансом понимается посылка сегмента и получение подтверждения на его успешный прием.

Такая схема не позволяет работать с максимальной производительностью. Эффективность может быть значительно повышена, если разрешить передачу множества сегментов за один сеанс с группировкой всех подтверждений для них в одно.
Рассмотрим схему работы этого метода на примере двух станций — А и Б, которым необходимо обмениваться данными.
Станция Б выделяет буферное пространство для приема *n* сегментов. Поэтому станция Б может принять *n* сегментов, а станция А может послать те же *n* сегментов без необходимости ожидания подтверждений об их приеме. Для отслеживания подтверждений о принятых сегментах каждый из них помечается номером в последовательности. Станция Б подтверждает получение сегмента посылкой подтверждения на него, которое содержит номер в последовательности следующего ожидаемого сегмента. Это подтверждение также косвенным образом извещает станцию А о том, что станция Б готова получить следующие *n* сегментов, начиная с указанного номера. Такая схема работы годится для подтверждения получения множества сегментов.
Например, станция Б получает сегменты 2, 3 и 4, но воздерживалась от отправки подтверждения на сегменты 2 и 3 до получения сегмента 4. Посылая подтверждение с номером в последовательности 5, станция Б подтверждает получение сегментов 2, 3 и 4 за один раз. Таким образом, можно сказать, что станция А ведет список номеров в последовательности сегментов, которые ей разрешено посылать, а станция Б поддерживает список номеров в последовательности сегментов, которые она готова принять.
Эти списки называют *окнами сегментов,* а такую схему передачи сообщений часто называют *управлением потоком с использованием плавающего окна.*
Пропускная способность
Рассмотрим методы определения максимально возможной пропускной способности соединения протокола TCP. Пропускная способность зависит от размера окна передачи, задержки и скорости пересылки данных. Используем следующие обозначения:
W — размер окна передачи в байтах;
R — скорость передачи данных (бит/с) по определенному соединению, доступная на стороне отправителя;
D — задержка (в секундах) при передаче данных между отправителем и получателем через определенное соединение.
Управление потоком данных
Управление потоком данных использует механизм плавающего окна, но кроме этого, применяется также более гибкая схема приема/передачи данных и отсылки подтверждений на успешный прием данных. Управление потоком протокола TCP использует так называемую схему с выделением лимита на передачу данных. По этой схеме каждый передаваемый байт имеет свой собственный номер в последовательности (SN). Когда протокол TCP посылает сегмент, он выставляет в поле номера в последовательности номер первого байта в поле данных этого сегмента. На принимающей стороне пришедший сегмент подтверждается сообщением, в котором указывается (*A=i, W=j).*Такая запись имеет следующее значение: если величина *A (ACK)* равна i, это значит, что сообщение подтверждает получение всех байтов, вплоть до номера в последовательности *i—1;* следующие ожидаемые байты имеют номер в последовательности *i.* Кроме того, выдается разрешение на посылку дополнительного окна *W (Window)* из *j* байтов; то есть байтов с номерами в последовательности от i до *i +j - 1.*



Билет 12. Сокеты.

Сокет (Sockets) – функциональный элемент представительского уровня для двунаправленной связи, который может использоваться для взаимодействия с другим процессом на одной и той же машине или с процессом, запущенным на других машинах.

Каждый сокет характеризуется:

- ❖протоколом
- ❖локальным адресом (инициатора соединения)
- ❖удаленным портом

Основы создания сокетов

При создании сокета, необходимо определить три параметра: стиль взаимодействия, пространство имен, и протокол.

Стиль взаимодействия контролирует, как сокет обрабатывает передаваемые данные, и определяет количество партнеров взаимодействия. Через сокеты данные передаются блоками (пакетами). Стиль взаимодействия определяет, как эти пакеты будут обработаны и как они передаются от отправителя к получателю.

Пространство имен определяет, как записаны адреса сокета (socket addresses). Адрес сокета идентифицирует один конец подключения сокета.

Протокол определяет, как передаются данные.

Цикл жизни сервера состоит из:

- ❖создания сокета
- ❖привязки сокета к адресу
- ❖вызова listen, разрешающего соединение с сокетом, вызова accept, принимающего входящие соединения
- ❖закрытия сокета

Системные вызовы

Виды системных вызовов:

socket – создать сокет

closes – уничтожить сокет

connect – создать соединение между двумя сокетами
bind – привязать сокет к порту сервера
listen – настройка сокета для принятия подключений
accept – принять запрос на соединение и создать сокет для процесса взаимодействия
Сокеты представляются дескрипторами файлов.
Сокеты UNIX-domain используются только для взаимодействия двух процессов только на одном компьютере. Сокеты Internet, используются для соединения нескольких процессов на различных машинах, подключенных к сети.
Для соединения процессов через Интернет сокеты используют пространство имен Интернет.

Билет 13. Концепция WWW. Гипертекст. Ссылки. HTML.
Word Wide Web (Web или WWW) предоставляет легкий в управлении графический интерфейс Internet. Эти документы, а также ссылки между ними образуют информационную «паутину» (Web).
Идея Т. Бернерс-Ли заключалась в том, чтобы применить гипертекстовую модель к информационным ресурсам, распределенным в сети, и сделать это максимально простым способом. Он заложил три краеугольных камня системы из четырех существующих ныне, разработав:
- ❖язык гипертекстовой разметки документов HTML (HyperText Markup Lan-guage);
- ❖универсальный способ адресации ресурсов в сети URL (Universal Resource Locator);
- ❖протокол обмена гипертекстовой информацией HTTP (HyperText Transfer Protocol).

Позже команда NCSA добавила к этим трем компонентам четвертый:
- ❖универсальный интерфейс шлюзов CGI (Common Gateway Interface).

Идея HTML – пример чрезвычайно удачного решения проблемы построения гипертекстовой системы при помощи специального средства управления отображением.

Гипертéкст (англ. *hypertext*) — термин, обозначающий систему из текстовых страниц, имеющих перекрёстные ссылки.

В компьютерной терминологии гипертекст — это текст, сформированный с помощью языка разметки (например, HTML) с расчетом на использование гиперссылок.
Ссылки
Гиперссылка (англ. *hyperlink*) — часть гипертекстового документа, ссылающаяся на другой элемент (команда, текст, заголовок, примечание, изображение) в самом документе, на другой объект (файл, каталог, приложение), расположенный на локальном диске или в компьютерной сети, либо на элементы этого объекта.
Гипертекстовые ссылки являются ключевым компонентом, делающим WEB привлекательным для пользователей. Ссылки могут указывать на другой документ, специальное место данного документа или выполнять другие функции, например, запрашивать файл по FTP-протоколу для отображения его браузером. URL может указывать на специальное место по абсолютному пути доступа, или указывать на документ в текущем пути доступа, что часто используется при организации больших структурированных WEB-сайтов.
HyperText Markup Language (HTML) является стандартным языком, предназначенным для создания гипертекстовых документов в среде WEB. Такие документы могут просматриваться различными типами WEB-браузеров. Использование HTML позволяет форматировать документы для их представления с использованием шрифтов, линий и других графических элементов на любой системе, их просматривающей.
Основные положения
Все тэги HTML начинаются с "<" (левой угловой скобки) и заканчиваются символом ">" (правой угловой скобки). Как правило, существует стартовый тэг и завершающий тэг. Завершающий тэг выглядит так же, как стартовый, и отличается от него прямым слешем перед текстом внутри угловых скобок.
Пример тэга заголовка, определяющий текст, находящийся внутри стартового и завершающего тэга и описывающий заголовок документа:
<TITLE> Заголовок документа </TITLE>
Структура документа
HTML-документ представляет собой тэговую модель, то есть совокупность элементов, каждый из которых окружён тэгами.

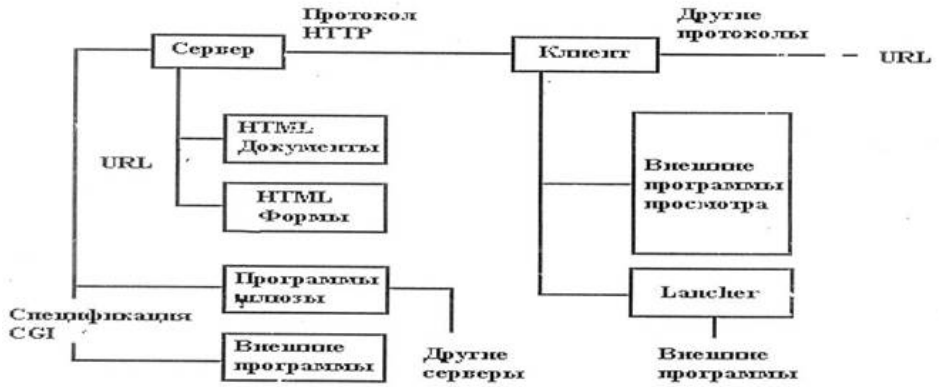
Тэги определяют области действия правил интерпретации текстовых элементов документа.
В своём наиболее общем виде структура документа HTML выглядит следующим образом:

< HTML>
Содержание документа
</HTML>

Команда <HTML> должна быть первой в документе. Она всегда используется в паре с </HTML>, завершающей документ.
Между этими двумя командами располагается текст страницы и другие команды.
Сам элемент HTML состоит из двух частей: *заголовка (HEAD) и тела документа (BODY):*

<HTML>
<HEAD>
Содержание заголовка
</HEAD>
<BODY>
Содержание тела документа
</BODY>
</HTML>

Билет 14. WWW-архитектура. URL. Протокол HTTP.
От описания основных компонентов перейдем к архитектуре взаимодействия программного обеспечения в системе World Wide Web. WWW построена по хорошо известной схеме "клиент-сервер".



Программа-клиент выполняет функции интерфейса пользователя и обеспечивает доступ практически ко всем информационным ресурсам Internet.
Фактически, клиент – это интерпретатор HTML. И как типичный интерпретатор, клиент в зависимости от команд (разметки) выполняет различные функции. В круг этих функций входит не только размещение текста на экране, но обмен информацией с сервером по мере анализа полученного HTML-текста, что наиболее наглядно происходит при отображении встроенных в текст графических образов.
При анализе URL-спецификации или по командам сервера клиент запускает дополнительные внешние программы для работы с документами в форматах, отличных от HTML, например GIF, JPEG, MPEG, Postscript и т. п.
Вообще говоря, для запуска клиентом программ независимо от типа документа была разработана программа Luncher, но в последнее время гораздо большее распространение получил механизм согласования запускаемых программ через MIME-типы.
Другую часть программного комплекса WWW составляет сервер протокола HTTP, базы данных документов в формате HTML, управляемые сервером, и программное обеспечение, разработанное в стандарте спецификации CGI.
База данных HTML-документов – это часть файловой системы, которая содержит текстовые файлы в формате HTML и связанные с ними графику и другие ресурсы. Особое внимание хотелось бы обратить на документы, содержащие элементы экранных форм. Эти документы реально обеспечивают доступ к внешнему программному обеспечению.
Прикладное программное обеспечение, работающее с сервером, можно разделить на программы-шлюзы и прочие. Шлюзы – это программы, обеспечивающие взаимодействие сервера с серверами

других протоколов, например ftp, или с распределенными на сети серверами Oracle. Прочие программы – это программы, принимающие данные от сервера и выполняющие какие-либо действия: получение текущей даты, реализацию графических ссылок, доступ к локальным базам данных или просто расчеты.

URL

Одним из основный понятий концепции WWW стала универсальная форма адресации информационных ресурсов.

Место применения URL – гипертестовые ссылки, которые записываются в тагах < A HREF=URI > и < LINK HREF=URI >. Встраиваемые графические объекты также адресуются по спецификации URI в тагах < IMG SRC=URL > и <FIG SRC=URI >. Реализация URI для WWW называется URL(Uniform Resource Locator). Точнее, URI – это реализация схемы URL, отображенная на алгоритм доступа к ресурсам по сетевым протоколам.

При разработке URI преследовались следующие принципы:

- ❖Расширяемость – новые адресные схемы должны были легко вписываться в существующий синтаксис URI.
- ❖Полнота – по возможности, любая из существовавших схем должна была описываться посредством URI.
- ❖Читаемость – адрес должен был быть легко читаем человеком, что вообще характерно для технологии WWW – документы вместе с ссылками могут разрабатываться в обычном текстовом редакторе.

HTML использует URL (Uniform Resource Locator) для представления гипертекстовых ссылок и ссылок на сетевые сервисы внутри HTML-документа. Первая часть URL (до двоеточия) описывает метод доступа или сетевой сервис. Другая часть URL (после двоеточия) интерпретируется в зависимости от метода доступа. Обычно, два прямых слэша после двоеточия обозначают имя машины: method://machine-name/path/foo.html

URL имеет следующий формат: method://servername:port/pathname#anchor

METHOD Имя операции, которая будет выполняться при интерпретации данного URL. Наиболее часто используемые методы:

file: чтение файла с локального диска. Данный метод используется для отображения какого-либо файла, находящегося на машине пользователя.

http: доступ к WEB-странице в сети с использованием HTTP-протокола.

ftp: запрос файла с анонимного FTP-сервера.

mailto: активизирует почтовую сессию с указанным пользователем и хостом.

telnet: обращение к службе telnet

news: вызов службы новостей, если браузер ее поддерживает.

SERVERNAME Необязательный параметр, описывающий полное сетевое имя машины.

PORT Номер порта TCP на котором функционирует WEB-сервер.

PATHNAME Частичный или полный путь к документу, который должен вызваться в результате интерпретации. Если после сетевого имени машины сразу идет имя документа, то он должен находиться в корневом каталоге на удаленной машине или (что чаще) в каталоге, выделенном WEB-сервером в качестве корневого. Если же URL заканчивается сетевым именем машины, то в качестве документа запрашивается документ из корневого каталога удаленной машины с именем, установленным в настройках WEB-сервера (как правило, это index.html).

#ANCHOR Данный элемент является ссылкой на строку (точку) внутри HTML-документа. Большинство браузеров, встречая после имени документа данный элемент, размещают документ на экране таким образом, что указанная строка документа помещается в верхнюю строку рабочего окна браузера. Точки, на которые ссылается #anchor, указываются в документе при помощи тэга NAME.

Протокол HTTP

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

Основным объектом манипуляции в HTTP является ресурс, на который указывает URI (англ. Uniform Resource Identifier) в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы, но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. (В частности для этого используется HTTP-заголовок.) Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

Структура протокола

Каждое HTTP-сообщение состоит из трёх частей, которые передаются в указанном порядке:

- ❖Стартовая строка (англ. Starting line) — определяет тип сообщения;
- ❖Заголовки (англ. Headers) — характеризуют тело сообщения, параметры передачи и прочие сведения;
- ❖Тело сообщения (англ. Message Body) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой.

Заголовки и тело сообщения могут отсутствовать, но стартовая строка является обязательным элементом, так как указывает на тип запроса/ответа.

1. Стандарт 802.3z. Сеть GigaBit Ethernet (GE)

1) Скорость 1Гбит/с
2) Метод доступа CSMA/ CD
3) Сохранен формат кадра 802.3
4) Используется f/o – оптоволокно

При увеличении скорость до 1 гб/с диаметр сети уменьшился до 40м, что, при использовании хаба означало, что провода должны быть не больше 5 м.

Для борьбы с этим ввели понятие *расширение несущей.* Для небольших кадров после основной информации стали передавать служебные символы, чтобы увеличить размер до 512байт, тем самым увеличив время передачи и дав возможность выявить коллизии. Это позволило расширить диаметр сети до 200м, но при этом при передаче кадров <512 байт приходится передавать бесполезный заполнитель, что сильно уменьшает скорость передачи.

Для этого ввели передачу *блоком кадра*, т.е. передача уже не кадрами, а блоками суммарной длиной ≤8 кбайт. В таком случае расширение несущей используется только если кадры кончились, а 8 кбайт еще нет.

При использовании кодирования 4B5B при большом количестве информации стала появляется постоянная составляющая (например, если высокий уровень сильно чаще, чем низкий), которая приводила к систематическим ошибкам (для UTP) и сжиганию лазера (для f/o)

В связи с этим от кода 4B5B отказались в пользу кода 8B10B. Из символов взяли только те, в которых 5 единиц и 5 нулей, а так же те, в которых 4 нуля и 6 единиц и наоборот. При этом каждому байту поставили в соответствие две кодовых комбинации (одна с 6 единицами, а другая с 6 нулями). При передаче считаются 0, и если их в потоке больше, используется символ с большим количеством единиц.

При использовании NRZ кода максимальная частота стала 625 МГц (максим чередование – при скорости в 1250Мбит/с кажд раз, а скорость 1250Мбит – было 1000 по 8бит, стало 1250 по 10бит)

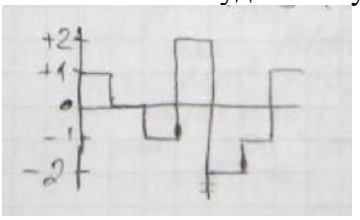
Формат кадра по протоколу Ethernet

1. 7б – преамбула – установить синхрон
2. НО – 1б – нач ограничитель – указание на предстоящий прием кадра
3. 6б – адрес получателя
4. 6б – адрес отправителя (MAX адрес – все FF - броадкаст)
5. 2б – длина поля данных в байтах
6. 1б – адрес точки входа в службы приема и обработки
7. 1б – адрес точки входа в службы отправителя
8. 1б – Control – управление – параметры обработки
9. 1б – OUT – идентификатор организации, контролирующей входы обработки
10. 1б – тип протокола внешнего уровня (IP, Spanning tree)
11. 48б-1492б – данные – если данных меньше, то используется поле заполнения
12. 4б – FCS – контрольное поле

2. Стандарт 802.3ab на витой паре (GigaBit Ethernet UTP)

1) Скорость 1Гбит/с
2) Метод доступа 1-CSMA/ CD
3) Сохранен формат кадра 802.3
4) Используется UTP cat 5, 5e – неэкранированная витая пара

У UTP полоса пропускания только 100 МГц, соответственно возникает сложность с передачей. Для этого информацию передают по всем 4 парам (т.е. на скорости 250 мбит/с на одну пару), что все равно слишком много. Для решения этой проблемы используют систему кодирования PAM5 – амплитудно-импульсную модуляцию 5 уровнями сигнала.



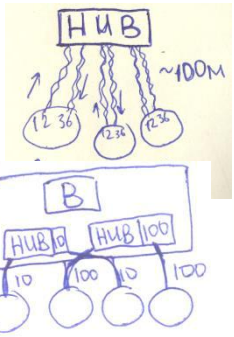
От двоичной СС перешли к пятеричной. Соответственно, вместо 256 (2^8) = 8 бит используют 625 (5^4) = 4 бита (в пятеричной системе) сигналов, из которых выбрали хорошо декодирующиеся символы (не те, что 4444). Таким образом, получили не 250Мбит/с, а 125 бод/сек на одну пару, что дает максимальную частоту 62.5 МГц, что позволяет использовать UTP cat5.

В Ethernet и Fast Ethernet были заняты только 2 пары, соответственно дуплекс мог осуществлятся с помощью незанятых пар. В GE заняты все четыре пары. Для организации дуплексной передачи они одновременно заняты приемом и передачей, для этого в каждом PC установлен сигнальный процессор, который вычитает сигнал самой PC и оставляет только информацию от «соседа».

Формат кадра по протоколу Ethernet

1. 7б – преамбула – установить синхрон
2. НО – 1б – нач ограничитель – указание на предстоящий прием кадра
3. 6б – адрес получателя
4. 6б – адрес отправителя (MAX адрес – все FF - броадкаст)
5. 2б – длина поля данных в байтах
6. 1б – адрес точки входа в службы приема и обработки
7. 1б – адрес точки входа в службы отправителя
8. 1б – Control – управление – параметры обработки
9. 1б – OUT – идентификатор организации, контролирующей входы обработки
10. 1б – тип протокола внешнего уровня (IP, Spanning tree)
11. 48б-1492б – данные – если данных меньше, то используется поле заполнения
12. 4б – FCS – контрольное поле

3. Коммутаторы локальных сетей

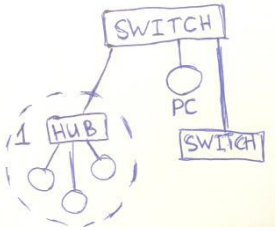


Хаб – если два устр-ва начинают пересылку данных прим одноврем, то конфликт – коллизия – А по 12 передает данные, и хаб отправляет данные А по 36. А видит, что по разным парам идут разные данные – «каша» из сигналов и считает, что косяк

Плюс если хотя бы одно устройство работает на 10Мбит/сек – вся сеть должна тоже быть 10Мбит/сек. Начали делать Dual speed hub – в нем был мост и два хаба с разными скоростями – 10 и 100 Мбит/сек. Чем умнее был хаб – тем дороже. Стоимость порта хаба стала прим = стоимости порта моста. Придумали коммутатор

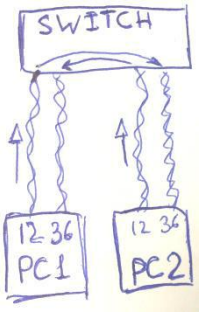
Коммутатор – устройство, работающее на MAC-подуровне (а хаб – на физич), интеллектуальное, осуществляющее пересылку между портами кадров на основе MAC-адресов.

Коммутатор – многопортовой мост, а хаб – разделяемая среда передачи данных.



1. Содержит табл коммутации – соотнесены макадреса и порты
2. Буферизация при неготовности порта
3. Если неизвестно, где нужный мак-передача на все порты
4. Допускает разную скорость на разных портах
5. Коммутация разделяемых сегментов, портов (PC, свитчи)

Сеть, построенная на свитчах, образует единый широковещательный конфликт. Сеть, построенная на хабах - единый домен-конфликт (один передает, другие слушают, конкуренция за сегмент)



6. Дуплексный режим работы – в режиме коммутации портов:

PC1 передает кадр, PC2 начинает передачу прим одновременно

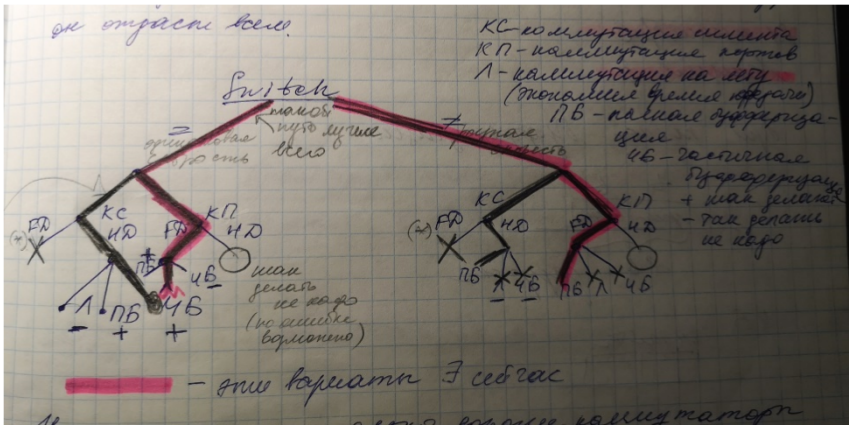
Получается, PC2 передает одно, слышит другое. Может подумать, что конфликт. Чтобы не подумал – надо выставить режим FD (изначально PC думает, что HD). Сегодня все PC и свитчи в FD Если оба в HD – то единый разделяемый сегмент

Если PC передают порт-в-порт – FD. Если подключен хаб – FD не м.б

Способы коммутации:

1. С полной буферизацией – как мост
2. С частичной буферизацией – принимаем первые 64байт, если конфликта не было – то его и не будет – отправляем остальное без буферизации
3. Коммутация на лету – как только получили адрес, сразу перенаправляем. Но если порт занят – будет косяк. Время соизмеримо со временем передачи по HUB

Если в сегменте началась передача и возник конфликт, а свитч начал отправлять на лету – то в итоге отправится только кусок сообщения. Выход – полн или частичн. Если коммутация портов с разными скоростями – только ПБ



С одинак скор – для сегментов – ЧБ Точки в кружках – так делать не нужно, но по ошибке возможно. Сегодня сегменты почти не коммутируют, потому мало ветвей для них – для разн скор ПБ и для одинак – ЧБ

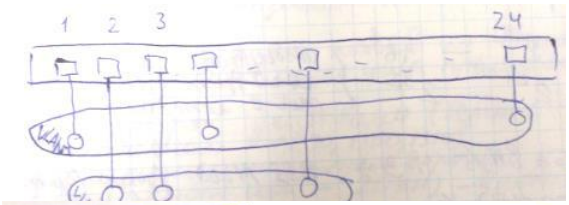
4. Виртуальные локальные сети VLAN

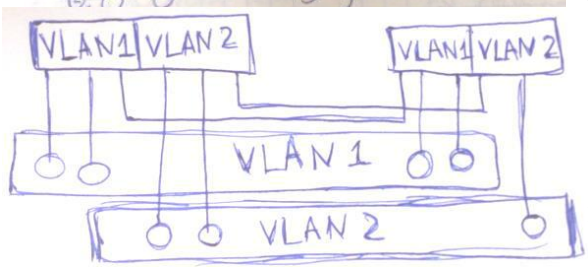
При поступлении бродкаст-кадра на SW – тот рассылает его по всем портам – в итоге цепная реакция и кадр рассылается всюду-всюду. Широковещат домен – часть сети, через которую проходит порожденный где-либо бродкаст кадр. Для того, чтоб уменьшить % бродкаста в сетях, ввели понятие VLAN:

VLAN – вирт лок сеть – совокупность портов коммутаторов, образующая единый широковещат. домен.

2 способа разделения на VLAN – по портам (наиб распростр) и по mac-адресам

1. По портам – 90%





Если в VLAN1 появляется бродкаст кадр – он идет в порты, соответств VLAN1. Внутри свитча табл соотв-вия номера порт и номера VLAN. Если сеть разветвленная из нескольких коммутаторов:

При формировании кадра, если укажем в адресате MAC2 – а отправ-ль MAC1 –

свитч не пропустит из соображ-й безопасности

Т.О. если подсети разные, коммутации не происходит

2. На основе MAC адресов

Составляется таблица коммутации, в которой свитч сам определяет в процессе обучения, какой мак на каком порте, а мы программно задаем соответствие маков и VLAN. Если меняем порт подключения – свитч все равно все помнит

Если свитчей несколько, то надо в КАДЖОМ свитче писать ВСЕ маки и их VLAN.

А место соединения двух коммутаторов – за каким-то портом будет несколько записей в таблице с разными маками и VLAN.

Слишком сложно конфигурировать и администрировать – слишком много забивать вручную. Поэтому используют по портам. Зато в сравнении с портами только одно соединение между коммутаторами, а в портах – их будет много

5. Построение VLAN по стандарту 802.1q

Стандарт, описывающий процедуру тэгирования трафика для передачи информации о принадлежности к VLAN.

Чтобы можно было тэгировать, пришлось менять формат кадра Ethernet

Тэг вставляется после макадресов, перед «Тип протокола», контрольная сумма тоже пересчитывается

|TPID(16b)||Prior(3)||CFI(1b)||VID(12b)||

TPID – идентификатор протокола тэгирования, в данном случае 8100h

Prior – поддержка приоритета трафика 8уровней – на самом деле стандарт 802.1pq

CFI – идентификатор канонического формата: 0 – Ethernet,1- кадр Token Ring,

FDDI VID – идентификатор VLAN, 2^12 VLAN

Максим размер кадра был 1518 байт, стал 1522. Оборудование, если не знает, будет такие кадры отбрасывать. Надо либо принимающую сторону учить их понимать, либо передающую сторону – делать их на 4байта меньше

На коммутаторах - при попадании обычного Ethernet кадра в коммутатор с 802.1q, коммутатор автоматически добавляет 4байта с номером соотв. VLAN источника кадра. При передаче из коммутатора в устройство эта метка удаляется, таким образом устройство может спокойно работать. Такие порты называются untagged – нетеггирующие. А там, где метка добавляется – tagged – тегирующие. Портам надо указывать, кадры с какими метками они не могут принимать.

Формат кадра по протоколу Ethernet

1. 7б – преамбула – установить синхрон
2. НО – 1б – нач ограничитель – указание на предстоящий прием кадра
3. 6б – адрес получателя
4. 6б – адрес отправителя (MAX адрес – все FF - броадкаст)
5. 2б – длина поля данных в байтах
6. 1б – адрес точки входа в службы приема и обработки
7. 1б – адрес точки входа в службы отправителя
8. 1б – Control – управление – параметры обработки
9. 1б – OUT – идентификатор организации, контролирующей входы обработки
10. 1б – тип протокола внешнего уровня (IP, Spanning tree)
11. 48б-1492б – данные – если данных меньше, то используется поле заполнения
12. 4б – FCS – контрольное поле

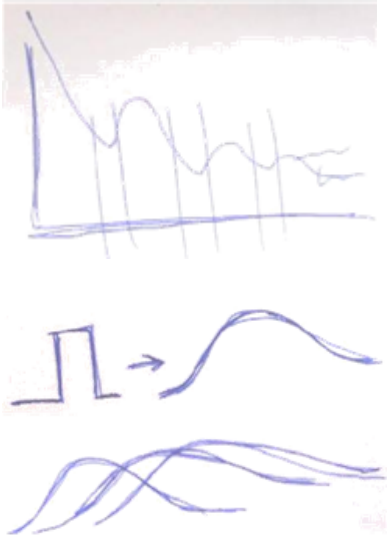
7. Проблемы высокоскор передачи по оптич сетям и пути их решений:

Для дуплексной сети, работающей на коммутаторах, CSMA/CD не используется, т.к. нет коллизий. Значит, ограничение на диаметр сети тоже пропало. Стандарты передачи данных по оптоволокну:

1000Base-SX, 1000Base-LX, 1000Base-HX – различаются по длине волны, которую светит лазер (850-short-диод, 1310-long-лазер, 1550-high-мощный лазер)

Всего используют три длины волны – на которых затухание существенно меньше, чем в окрестности – проэкспериментировали, брали разные показатели типа затухания, коэф. стоячей волны и т.д.

Получили распределение:



S,L,H – окна прозрачности оптоволокна, стоимость увеличивается с возрастанием. Использование:

SX – многомодовое, 50мкм – 550м, старое – 62.5мкм – 220v

LX – одномодовое, 9мкм – 5км

HX – одномодовое, 9мкм – 40км

Причиной затухания в оптоволокне является различие в длинах пути, которые проходит фотон, т.к. он распространяется во все стороны.

А значит, на выходе будет не четкий сигнал, а смазанный. При большом количестве импульсов, чем дальше передаем – тем сложнее выделить нужный нам сигнал Это называется дифференциальной задержкой.

Для того, чтобы ее уменьшить:

1. Чем тоньше – тем меньше (9мкм или 50мкм) (сечение кабеля)
2. Чем больше длина волны – тем меньше проходимый путь – тем меньше (мощность передатчика)

<u>Мультиплексирование по длине волны WDM:</u>

Лучи света с разными длинами волн друг с другом не взаимодействуют. Ставят MUX в начале и в конце – берут маленькую разницу по частотам – каждая волна со своей частотой несет поток данных

Система WDM обеспечивала передачу двух сигналов, одного во втором оптическом окне (около 1310 нм), а другого — в третьем оптическом окне (около 1550 нм). Двухканальная передача осуществляется по одному и тому же волокну, по одному сигналу в каждом окне. Таким образом, пропускная способность оптоволокна удваивается.

Пропускная способность оптоволокна при использовании принципа DWDM многократно увеличивается, поскольку в третьем окне компактно располагается много каналов, обеспечивающих одновременную передачу сигналов по одному одномодовому оптоволокну.

8. Стандарт 802.3ae (10GE):

1) Формат кадра неизменный
2) Отказ от полудуплесной передачи и разделяемой среды
3) Ориентирован на f/o
4) Для синхронизации используется 64b/66b, у которого избыточность около 3%

Используются 5 модификации, в зависимости от используемого окна в длине волны:
10GBASE – SR – ориентирован на MM f/o и длину волны 850 нм. Передача осуществляется на расстояние до 65м. Самое дешевое
10GBASE – LR – используется SM, 1310нм, расстояние до 10 км.
10GBASE – ER – SM, 1550нм, до 40 км
10GBASE - LX4 – MM, 1310нм, WDM – (wave length division multiplexing – разделение по длинам волны. Берут 4 потока с шагом 20нм так, чтобы они попали в одно окно прозрачности), 4 потока, кодирование 8B10B, расстояние до 300 м.
10GBASE - EX4 – SM, расстояние свыше 50 км. Ужасно дорогое и редко используется.
Схожие технологии применяются в глобальных сетях, но они носят буквы SW и LW соответственно.
Формат кадра по протоколу Ethernet

1. 7б – преамбула – установить синхрон
2. НО – 1б – нач ограничитель – указание на предстоящий прием кадра
3. 6б – адрес получателя
4. 6б – адрес отправителя (MAX адрес – все FF - броадкаст)
5. 2б – длина поля данных в байтах
6. 1б – адрес точки входа в службы приема и обработки
7. 1б – адрес точки входа в службы отправителя
8. 1б – Control – управление – параметры обработки
9. 1б – OUT – идентификатор организации, контролирующей входы обработки
10. 1б – тип протокола внешнего уровня (IP, Spanning tree)
11. 48б-1492б – данные – если данных меньше, то используется поле заполнения
12. 4б – FCS – контрольное поле

9. Стандарт 802.3an (10GE UTP):

- особенности кодирования
- витая пара 6А
- дуплексный режим
- формат кадра

Используется UTP cat 6(250 Мгц, 100м)
Используется каждая пара, на каждую пару приходится по 2.5 Гбит/сек, что больше полосы пропускания. Для решения проблемы используют кодирование PAM16 – амплитудою модуляцию 16 уровнями, т.е. 24 бита заменяются 8 словами. Т.е.
$2^{24} – 16^{8}$

Т.к. количество возможных команд сильно больше трамбуемых, мы выбираем только удобные для декодирования комбинации. Таким образом, на одну пару приходится 833 Мбод, что приблизительно равно 417 Мгц. В связи с этим, 10 GE можно построить на cat 6 только если расстояние меньше 55. 100м можно построить только на cat 6a, в которой вокруг пар навивается пластиковый шнур. Из-за этого уменьшаются помехи, наводимые от соседних кабелей.

В Ethernet и Fast Ethernet были заняты только 2 пары, соответственно дуплекс мог осуществлятся с помощью незанятых пар. В GE заняты все четыре пары. Для организации дуплексной передачи они одновременно заняты приемом и передачей, для этого в каждом PC установлен сигнальный процессор, который вычитает сигнал самой PC и оставляет только информацию от «соседа».
Формат кадра сохранен.
Формат кадра по протоколу Ethernet

1. 7б – преамбула – установить синхрон
2. НО – 1б – нач ограничитель – указание на предстоящий прием кадра
3. 6б – адрес получателя
4. 6б – адрес отправителя (MAX адрес – все FF - броадкаст)
5. 2б – длина поля данных в байтах
6. 1б – адрес точки входа в службы приема и обработки
7. 1б – адрес точки входа в службы отправителя
8. 1б – Control – управление – параметры обработки
9. 1б – OUT – идентификатор организации, контролирующей входы обработки
10. 1б – тип протокола внешнего уровня (IP, Spanning tree)
11. 48б-1492б – данные – если данных меньше, то используется поле заполнения
12. 4б – FCS – контрольное поле

11. Назначение и общая характеристика протокола агрегирования канала

Агрегирование в один логический канал – еще одна форма использования избыточных альтернативных связей.
Составляющая агрегированного логического канала – транк
Повышается надежность сети и ее производительность
При отказе одного из транков трафик распределяется между оставшимися
Есть коммутатор, в котором используется только один порт для связи с другим коммутатором. Хотим повысить скорость путем увеличения скорости – тогда придется использовать ПБ вместо передачи на лету или денег не хватит
Есть другое решение – не трогать коммутаторы, а использовать LACP – протокол агрегирования каналов. Чтобы это работало, надо сообщить коммутаторам. При построении покрывающего дерева эти каналы должны рассматриваться как один логический, плюс – принадлежать одной VLAN
Если порты с разной скоростью – хоть и заявлено в стандарте, но работать вряд ли будет.
Допустимое число линий в одном логич канале и число логических каналов определяется производителем.
Статический способ распределения кадров между портами транков – за определенным портом транка потока закрепляется определенный вид кадров. Признаки, по которым распределяют:

1. MAC адреса – у циски – исключ ИЛИ над двумя последними битами мака источника и приемника. Итог – условные номера портов транка. Но реальная нагрузка не учитывается – так что м.б. нерационально
2. IP-адреса пакетов, инкапсулированных в кадры
3. Типы прикладных протоколов – почта, веб-трафик и т.д.

12. Особенности реализации алгоритма покрывающего дерева в коммутаторах RSTP и MSTP:

В дополнению к кол-ву маршрутизаторов, водится новая метрика - скорость интерфейса (при расчете до корневого моста).

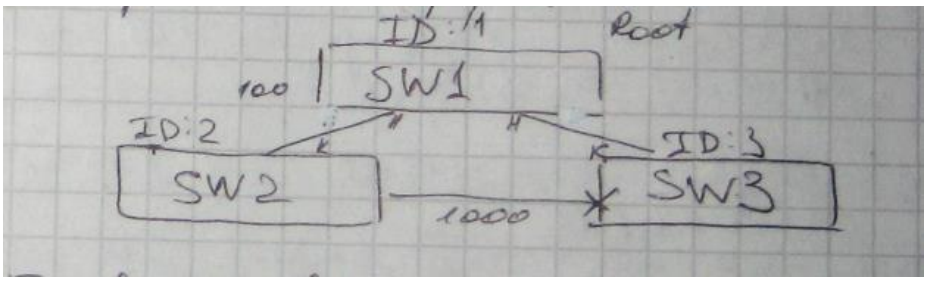
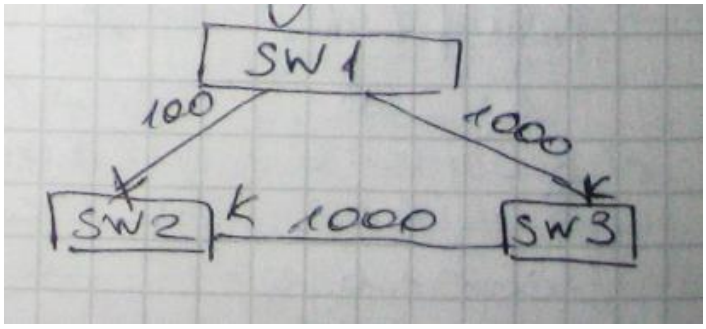


Таблица значений по умолчанию:

10 Тбит/с	2
1Тбит/с	20
100Гбит/с	200
10Гбит/с	2000
1Гбит/с	20000
100Мбит/с	200000



Время сходимости: перестройка дерева может занимать до 50с из-за того, что решение принимал только корневой мост, что очень долго. Соответственно появился **RSTP** – быстрый протокол покрывающего дерева (802.1w). *Коммутаторы стали учитывать тип сегмента,* подключенного к порту. Различаются следующие типы сегментов:

- Сегмент типа «точка-точка». В коммутируемых сетях это единственный тип сегмента; для него у порта существует единственный порт-сосед.
- Разделяемая среда. Стандарт RSTP по-прежнему учитывает существование разделяемой среды, так как формально ее никто не отменял, и все стандарты, включая основной стандарт Ethernet IEE 802.3, описывают работу сегмента этого типа.
- Тупиковая связь (edge port). Связь, которая соединяет порт коммутатора с конечным узлом сети; по этому сегменту нет смысла ожидать прихода сообщений протокола RSTP. Тупиковая связь конфигурируется администратором. В случае подключения к порту тупикового сегмента этот порт не участвует в протоколе RSTP, а сразу после включения переходит в стадию продвижения кадров. Нужно заметить, что в стандарте RSTP начальное заблокированное состояние портов переименовано в состояние отбрасывания.

Исключается стадия прослушивания. Коммутаторы не выдерживают паузу в 15 с для того,чтобы зафиксировать соответствующую роль порта, например корневого или назначенного. Вместо этого порты переходят в стадию обучения сразу же после назначения им роликорневого или назначенного порта.

Сокращается период фиксации отказа в сети — вместо 10 периодов неполучения сообщений Hello он стал равен трем таким периодам, то есть 6 с вместо 20.

Введены новые роли портов — появились **альтернативный** (alternative) и **резервный** (backup) порты. Альтернативный порт является портом-дублером корневого порта коммутатора, то есть он начинает продвигать кадры в том случае, когда отказывает (либо перестает принимать сообщения Hello в течение трех периодов) корневой порт. Резервный порт является портом-дублером назначенного порта сегмента; однако такая роль порта имеет смысл только для сегментов, представляющих собой разделяемую среду.

Принцип работы в общих чертах похож на STP: выбирается корневой коммутатор, к которому, каждый из участвующих в построении дерева коммутатор, ищет кратчайший маршрут (с учётом пропускной способности канала) через соседние коммутаторы (или напрямую). Линии, не попавшие в маршрут, переводятся в режим ожидания и не используются для передачи данных пока работают основные линии.

В случае выхода из строя основных линий, ожидающие линии используются для построения альтернативной топологии, после чего одна из линий становится активной, а остальные продолжают находиться в режиме ожидания.
Раньше только корневой мост посылал сообщения присутствия, теперь каждый коммутатор рассылает соседям сообщения присутствия. Если один из коммутаторов вышел из строя, то его сосед сам запускает реконфигурацию, не дожидаясь, пока информация дойдет до корневого моста. Это в разы быстрее.
MSTP – множественный протокол покрывающего дерева, позволяет создать несколько покрывающих деревьев и приписывать к ним различные виртуальные локальные сети. Обычно создается небольшое количество деревьев, например два или три, чтобы сбалансировать нагрузку на коммутаторы, в противном случае, как мы видели в примере на рис. 14.2 и 14.3, единственное покрывающее дерево может полностью оставить без работы некоторые коммутаторы сети, то есть недоиспользует имеющие сетевые ресурсы. При создании двух покрывающих деревьев можно сконфигурировать приоритеты коммутаторов так, чтобы для одного дерева корневым коммутатором стал коммутатор 1, а для второго — коммутатор 2. Протокол MSTP основан на протоколе RSTP, поэтому обеспечивает быструю реакцию сети на отказы.

14. Общая характеристика протокола DHCP:

При подключении нового компьютера к сети возникает проблема, как ему получить IP-адрес, маску подсети – чтобы определять узлы своей подсети, шлюз по умолчанию – для посылки данных в другие подсети, адрес DNS сервера

Можно бегать и раздавать вручную бумажки, можно использовать DHCP – протокол динамической загрузки узла.

Работа протокола:

1. Устройство, подключившись к сети, отправляет сообщение типа DISCOVER – бродкаст, udp пакет, из порта 68 (клиент) в порт 67 (сервер), свой мак-адрес
2. Все DHCP-серверы, оказавшиеся поблизости, шлют устройству сообщение типа OFFER – предлагают конфигурацию, включающую IP, subnet mask, default gateway и т.д.
3. Устройство выбирает из предложенных конфигураций понравившуюся и шлет тому DHCP-серверу, что ее предложил, сообщение типа REQUEST
4. DHCP-сервер, получив REQUEST, радостно отправляет ACK и добавляет себе запись о выданном айпишнике
5. В конце срока, на который ему выдали IP, устройство делает перезапрос – отправляет re Request, подтверждая, что оно еще тут и никуда не делось
6. Сервер шлет ACK
7. Если айпишник больше не нужен – устройство шлет сообщ типа RELEASE

Чтобы коварные юзеры не подключали всякие роутеры и т.д. – админ может сделать фильтрацию по mac-адресам (мы же знаем начало мак-адресов самых распространенных роутеров). Помогает не от всех, но все же. Эта фильтрация в половине средств реализуется программно – спец библиотеки или функции прям предусмотренные. Или сами ручками айпишники раздаем – но муторно

15. Технология трансляции сетевых адресов NAT

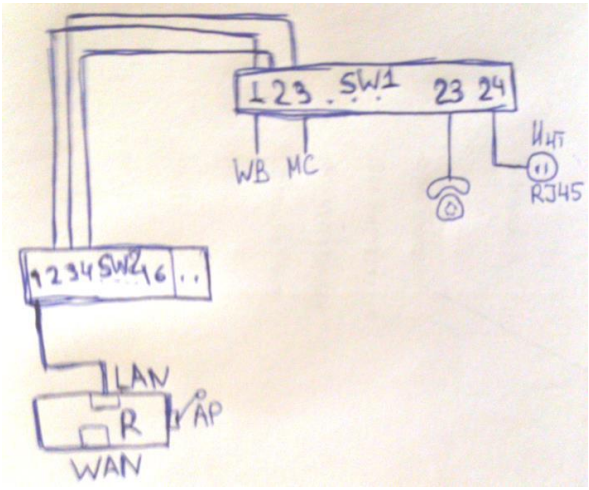
У нас есть сетка на работе из n компьютеров. А глобальных IP-адресов нам выделили M<n. Тогда используем NAT. Для адресации внутренних узлов используем адреса частных сетей, а если хотим отправить пакет во внешний мир – маршрутизатор, связывающий нас с интернетом поддерживает NAT – и заменяет внутренний айпишник на один из предоставленных нам глобальных. Что на что менять – прописывается в спец. таблице.
NAT преобразует множество {IPi} внутренних адресов узлов во множество {IPj} глобальных адресов.
Есть три концепции: 1-в-1, неск-в-1, PAT

1. 1-в-1 – каждому адресу в частной сети поставлен в однозначное соответствие глобальный айпишник.
+: устройство доступно из внешней сети, повышена безопасность
2. неск-в-1 – у нас есть пул глобальных адресов и адресу из частный сети ставится в соответствие доступный в данный момент
3. PAT – все адреса из частной сети отображены в один глобальный. Спасает, когда дефицит адресов, используются различный порты.

Вообще минусы – некоторым приложениям или протоколам требуется точный адрес. И если его нет – то ниче не получится.
Некоторые сервисы могут решить, что множество подключений или запросов от одного айпишника (PAT) – это DoS атака, а не пользователи, жаждущие зайти в аську.

16. Пример подключения локальной сети к интернету:

Надо подключить сетку к интернету. Сетка локальная – в ней могут быть принтеры, телефон, собсна сам интернет

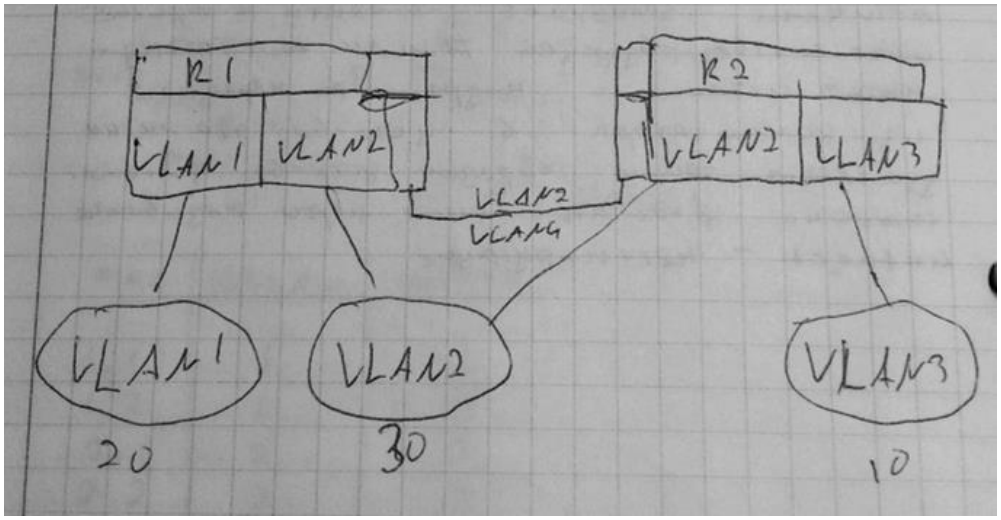


1. Возьмем свитч на 24 порта. Два последних выделим на подключение к Интернет через розетку и подключение телефона. Т.о. 23порт – подключаем телефон, а 24порт – ставим розетку и ура есть интернет.
(Кстати, для телеф связи используется 1витая пара, а для Интернета на скорости 100Мбит/сек – 2витых пары. В итоге можно цеплять на один порт)
2. Первый и третий порты используем для подключения принтеров – цветного и черно-белого

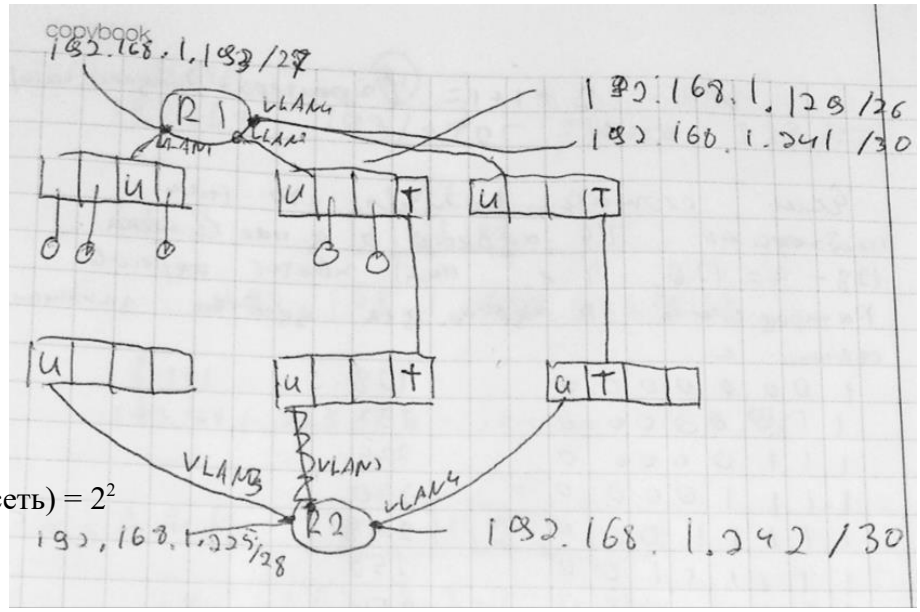
3. Надо поставить роутер, который будет отправлять пакеты из внутр сети во внеш и т.д. + раздавать интернет с точки доступа. Если устройств в локальной сети меньше 4, то нам свитч не нужен. Если больше – то цепляем свитч, напр 16п100BASE-TX + 2п1000BASE-T
4. Соединям свитч с роутером, подключаем наши компы таким образом (порт 1 свитча 2), потом к портам 234 свитча 2 цепляем принтеры и еще один порт – для общения дальнейшего.
5. Когда у нас компы будут подключаться к сети – они должны получать айпишники, маски подсети и т.д. Для этого на роутере поднимаем DHCP сервер, который этой лабудой будет заниматься. По умолчанию сервер будет настроен на сеть 192.168.0.0/24 и адрес порта роутера на выходе в свитчи 192.168.0.1
DHCP при настройке спросит у нас диапазон выдаваемых адресов. Мы выбираем, напр, 192.168.0.200 – 192.168.0.250 – с маской 24. Плюс у принтеров должен быть постоянный айпишник – прописываем привязку к макадресам принтеров
Типа MACWB 192.168.0.251, MACMC 192.168.0.252
Плюс еще когда новый комп посылает бродкастом запрос к DHCP, запрос приходит на роутер и там уже выдается свободный адрес из диапазона. Плюс выдается default gateway – шлюз по умолчанию для связи с внешним миром. Он у нас 192.168.0.1
6. Локальную сетку сделали. Теперь надо связаться с интернетом. Для этого надо поднять NAT-экран, который будет отображать все наши внутренние адреса в один глобальный айпишник, который даст нам провайдер. Кстати, у провайдера тоже стоит NAT – фактически, мы сидим через два NATA.
7. В таблице маршрутизации на роутере может быть только одна строчка
Адрес назнач Маска След.хоп Порт 0.0.0.0 0.0.0.0
след WAN
По маске 0.0.0.0 получаем, что любой пакет с адресом назначения ненашей сетки после наложения маски будет иметь адрес назнач 0.0.0.0 и пойдет во внеш сеть. Адрес след хопа мы получим, запросив DHCP сервер провайдера (раньше) и он нам еще кинет адрес DNS сервера – к которому нам обращаться за преобразованием символьного имени в айпишник.
8. Между роутером нашей сети и провайдера поднимается протокол канального уровня PPPoE. Этот протокол поддерживает аутентификацию – идентификацию с доказательством (мы же должны блокировать, если не платят). Если логин и пароль норм, происходит авторизация. Если поля совпадают, провайдер связывается со своей базой данных, где смотрит, оплатили мы или нет. Если все ок – выдает нам IP для связи с внешним миром.

Ураа, есть интернет

18. Пример построения локальной сети на коммутаторе 2-го и 3-го уровня с 3-мя VLAN:



Для соединения 2 коммутаторов вводится еще одна VLAN, у которой номер 4 и которая подключается через теггированный порт. Если перерисовать картинку, то мы получим:
VLAN 2 подключена к R2 через волнистую линию потому, что мы приняли решения, что за ее маршрутизацию будет отвечать R1.



Пусть у нас есть IP-пул 192.168.1.128/25.
Возникает вопрос, хватит ли нам 7 оставшихся битов на 4 VLAN.
Воспользуемся VLSM
V2 = 30+1+1+1 (бродкаст, сеть, роуетр), т.е. 2^6=64 адреса
V1 = 20+1+1+1 т.е. 2^5=32 адреса
V3 = 10+1+1+1 т.е. 2^4=16 адреса
V4 = 2+1+1 (2 роутера, бродкаст, сеть) = 2^2
Если все сложить, получим, что нам необходимо 116 адресов, а в пуле у нас есть 118.
Т. е. адресов хватает. Распределим IP-адреса:
V2:
192.168.1.128/26
192.168.1.129 (первый, router) – 192.168.1.191 (бродкаст)
V1:
192.168.1.192/27
192.168.1.193 (первый, router) – 192.168.1.223 (бродкаст)
V3:
192.168.1.224/28
192.168.1.225 (первый, router) – 192.168.1.239 (бродкаст) V4:
192.168.1.240/30
192.168.1.241 (первый, router) – 192.168.1.243 (бродкаст)

Таблица маршрутизации в R1

Сеть назначения	Маска		Шлюз	Порт
192.168.1.128	26		-	VLAN2
192.168.1.192	27		-	VLAN1
192.168.1.224	28		192.168.1.242	VLAN3
192.168.1.240	30		-	VLAN4

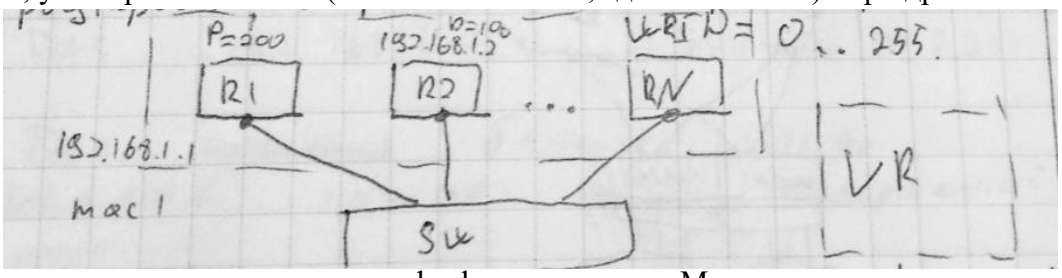
Пусть на роутер пришел пакет с адресом 192.168.1.230 от адреса 192.168.1.200. Первым делом на .230 абонент накладывает свою маску (/27). В результате увидев
.224, абонент поймет, что это не его сеть (у него 192/27). После этого абонент отправляет пакет на роутер по адр. 192.168.1.193/28. Тот смотрит в свою таблицу маршрутизации. Сначала накладывает маску /30, потом накладывает маску 28 и .т.д. Найдя соответствие, переселяет ее по нужному адресу (192.169.1.242). Важно помнить, что в роутере маски определены от большей к меньшему. В роутере R2 есть своя таблица маршрутизации, согласно которой он уже отправит этот пакет. Пусть нужно передать пакет из .130 в .131. При этом .130 подключен к R1, а .123 к R2. Тогда, наложив маску на .130 поймет, что адрес в его сети, воспользуется ARP, пошлет бродкаст, чтобы получить mac и после этого через коммутатор отправит кадр.
Важно понимать, что учитывая специфику билета, используется не роутер, а часть коммутатора третьего уровня, отвечающая за маршрутизацию.

19. Протокол VRP:

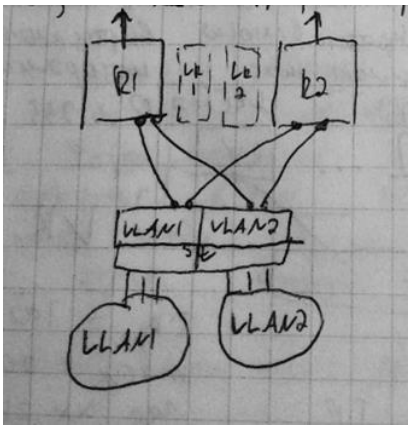
Router-ы бывают:

1) Магистральные – работающие внутри сети
2) Граничные – работают на границе с пользователем.

Отказоустойчивость магистральных роутеров осуществляется протоколами динамической маршрутизации (RIP и OSPF). Отказоустойчивость граничных роутеров осуществляется специально разработанным протоколом VRRP – Virtual Router Redundancy Protocol. Суть в том, что формируется избыточная группа роутеров (как правило, 2), которые объединяются в единую сеть (т.е. становятся широковещательно доступны). Каждому роутеру присваивается IP, и на нем заускается ПО VRRP. После этого пользователю эти роутеры представляются одним виртуальным роутером, у которого свой mac (00.00.SE.00.01.XX, где XX – VRID) и ip-адрес.



Роутерам присваивается приоритет от 0 до 255.Роутер с максимальным приоритетом становится мастер-роутером, остальные становятся backup-роутерами. Мастер-роуетр работает за виртуальный роутер. Кроме того, мастер-роутер с равным промежутком времени отправляет broadcast-пакет по адресу 224.0.0.18, давая понять остальным роутерам, что он функционирует. Если от мастер-роутера нет вестей больше 3 интервалов времени, следующий по приоритету роутер становится мастером.



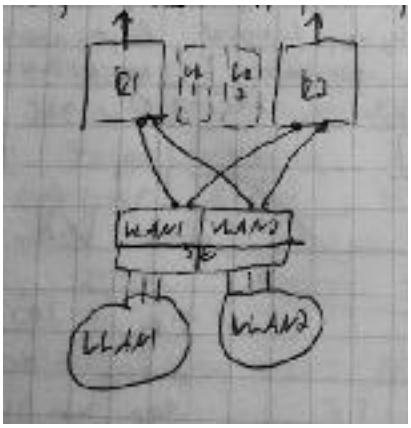
Для пользователя default gateway и DNS будет IP-адрес виртуального роутера.

Проблема технологии в том, что 1 роутер работает, а остальные простаивают. В связи с этим протокол модернизировали так, чтобы один роутер мог входить в разные группы. На рисунке есть два виртуальных роутера, в одном мастер-роутер – R1, а в другом – R2.

20. Пример построения отказоустойчивого маршрутизатора по протоколу VRP:

Пусть необходимо сделать две группы VLAN:

1) VRID = 10, R1 – master, R2 – Backup
2) VRID = 11, R1 – backup, R2 – master



Присваеваем приоритеты:

1. Pr1=200, Pr2=100 2. Pr2=200, Pr1=100

Присваеваем IP

VLAN1 – 192.168.1.0/24 VLAN2 – 192.168.2.0/24

IPR1= 192.168.1.1 IPR1= 192.168.1.2

IPR2= 192.168.1.2 IPR2= 192.168.2.2

У виртуальных роутеров будут MAC-адреса:

00.00.SE.00.01.0A00.00.SE.00.01.0A

Присвоим IP виртуальным роутерам

192.168.1.254 192.168.2.254

Тогда: DNS

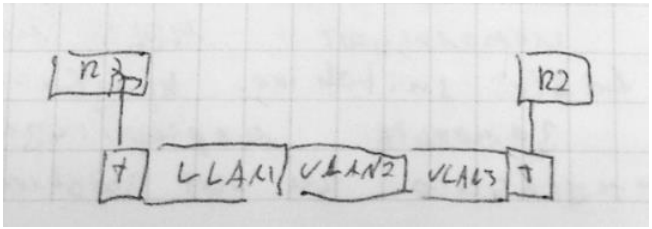
192.168.1.254 192.168.2.254

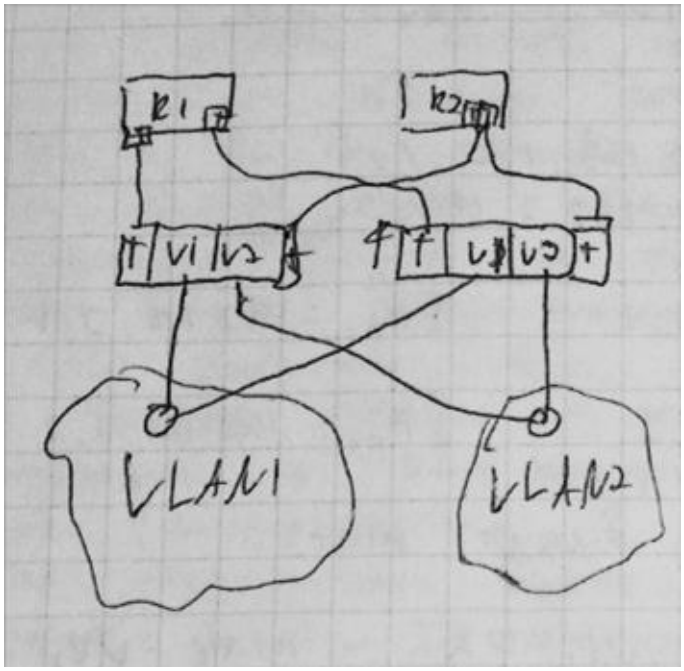
Gateway

192.168.1.254 192.168.2.254

Для соединения большего числа VLAN необходимо сделать теггированный порт.

Для борьбы с отказами каждому пользователю добавляется еще одна интерфейсная карта и настраивается multifinding





22. Технология MPLS

1. Назначение

Объединяет технику виртуальных каналов с функциональностью стека TCP/IP.
Виртуальные каналы – маршрутизация только один раз в сети, потом всем сообщениям присваиваются метки вирт каналов – в итоге адрес у нас не до фига байт, а меньше двух + таблицы маршрутизации гораздо меньше
Вообще сначала создавалось, чтобы можно было использовать протоколы маршрутизации не только TCP/IP, но и другие. Сейчас уже не особо актуально, важнее другое – можно передавать трафик разных протоколов канального уровня, а также способность предоставлять услуги VLAN

2. Основные понятия

LSR – коммутирующий по меткам маршрутизатор – выполняет функции IP-маршрутизатора и коммутатора виртуальных каналов. Это интеграция, а не тупо объединение
Пути, по которым движутся пакеты с метками - LSP

3. Общие принципы функционирования

В одном и том же устройстве поддерживается 2разн способа продвижения пакета – дейтаграммный по IP-адресам и ориентированный на соединения механизм виртуальных каналов. А протоколы маршрутизации используются для определения топологии сети и автоматического построения таблиц IP-маршрутизации и таблиц MPLS-продвижения. Комбинированное устройство может использовать любой из этих способов в зависимости от конфигурац. параметров протокола MPLS.
Метки действуют локально – в пределах каждого устройства LER и LSR – при передаче пакета со входного интерфейса на выходной выполняется смена значения метки

4. LSR, ELSR, LSP

Устройство LSR выполняет все функции IP-маршрутизатора, плюс включены блоки для управления и продвижения данных. Для выбора следующего хопа используем таблицу продвижения
Пограничное устройство LSR в технологии MPLS имеют спец. название – ELSR. Более сложное, принимает от других сетей трафик в форме стандартных IP-пакетов и добавляет к нему метку, направляет вдоль соотв пути к вых устройству LER через несколько LSR – на основе метки
LSP – путь коммутации по меткам, прокладывается с помощью протокола LDP (тогда совпадают с IP-путями) или модификации RSVP (если особое использование трафика)

5. Структура таблиц коммутации LSR, ELSR

Таблица продвижения LSR – формируется с помощью сигнального протокола – LDP (тогда совпадают с маршрутами IP-трафика) или модификация RSVP (в завис-ти от того, как мы обрабатываем трафик) – прокладываются маршруты – пути коммутации по меткам (LSP). Можно формировать вручную

Вх интерфейс	**Метка**	**След хоп**	**Действия**
S0	255	S1	32
S0	33	S2	234

След хоп – выходной интерфейс, на который нужно передать кадр
Действия – новое значение метки по выходу с LSR
Для передачи трафика между двумя устройствами LER нужно установить два LSP – в обоих направлениях. LER направляет входной трафик в один исходящих из LER путей LSR.
Введены классы эквивалентности продвижения – группа IP-пакетов, имеющих одни и те же требования к условиям транспортировки, продвигаются по одному LSP. В LER существует база данных этих классов и приходящий пакет по признакам соотносится к классу и ему присваивается соотв метка
Таблица FTN:**Признаки FEC || Метка**

Признаки – IP-адрес назначения, балансировка загрузки каналов в сети, в соответствии с требованиями VPN – для конкретной VPN создаем отдельн класс, по типу трафика – IP-телефония или веб трафик, по мак-адресу, если это кадр езернет Выходное устройство LER метку удаляет и передает пакет в след сеть в стандартной форме IP пакета.

6. PHP – отбрасывание метки

Усовершенствованный алгоритм обработки пакетов – метка удаляется не последним устройством на пути, а предпоследним. Т.к. если подумать головой, то нафиг нам метка, когда мы последний хоп определили в предпоследнем LSR'е и уже передаем туда, где пакет соотносится с классом, а не с путем. Так экономим одну операцию над этим несчастным кадром. Еще есть стек – но только хз надо или нет. Наличие стека меток позволяет организовать систему LSP с любым количеством уровней иерархии. Кадр включает в себя столько заголовков MPLS, сколько уровней иерархии имеет путь. Дно стека – метка с признаком S=1.

23. Технология LDP. Алгоритм назначения меток. Упорядоченный алгоритм запроса

В сетях, поддерживающих устройства с технологией MPLS, маршрутизаторы помимо составления таблицы маршрутизации должны составить таблицу продвижения – в которой соотнесены метки приходящих сообщений и метки, которые надо назначить по выходе из маршрутизатора в соответствии с LSP.

LSP мы прокладываем двумя способами – либо по протоколу LDP – тогда LSP получается соответствует IP-маршрутам, либо по модификации протокола RSVP – тогда LSP прокладывается, исходя из каких-то там своих соображений.

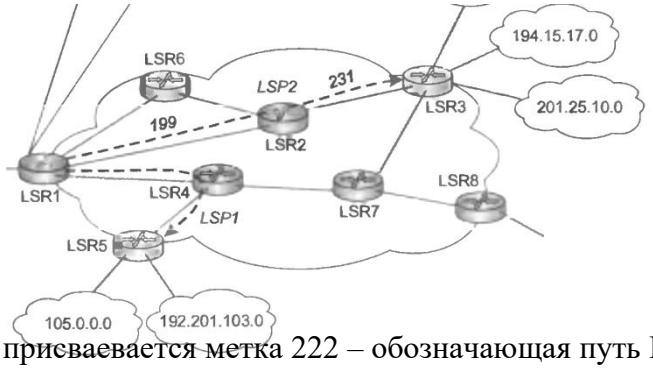
Тут рассматриваем LDP.

Этот протокол ответственен за прокладывание LSPпутей – т.е. фактически за распределение меток между LSR устройствами при необходимости проложить новый LSP к обнаруженной внезапно сети. Вариант работы LDP определяется двумя параметрами:

1. КАК мы распределяем метки – метод распределения распределение от лежащего ниже по потоку по запросу распределение без запроса
2. КАК мы управляем этим распределением – режим управления распределением меток Упорядоченный и независимый

Мы рассматриваем Упорядоченный режим управления распределения меток с запросом устройства вниз по потоку». Фишка в следующем – некое промежуточное устройство LSR не передает метку нового пути устройству LSR, лежащему выше по потоку, до тех пор, пока не получит метку для этого пути от устройства, лежащего ниже по потоку.

Смотрим на примере (кстати, LSR1, LSR5, LSR8,LSR3 – те самые ELSR)



Появилась сетка 194.15.17.0 (справа), которой нет в таблице продвижения – но сразу говорим, что эта сетка есть в таблице IP-маршрутизации.

У LSR 1 есть своя таблица продвижения, где указан признак класса сообщений и их метка. Т.е.

105.0.0.0,192.201.103.0 222

Т.е. сообщениям, идущим в эти сетки, присваевается метка 222 – обозначающая путь LSP1.

И тут вдруг приходит сообщение в сетку 194.15.17.0. LSR1 хз как в нее добраться – у него такой не прописано. Тогда он срочно активизирует протокол LDP для прокладки нового пути и требует метку для этой сети у маршрутизатора, который по IP-ТАБЛИЦЕ следующий на пути к этой сетке – LSR2 – и он у нас получается НИЖЕ по потоку. Но у нас же LDP – поэтому нам сначала надо соединение установить, чтоб общаться.

Сеансы LDP устанавливаются автоматически – все LSR'ы шлют сообщ HELLO – по групповому адресу и порту UDP. А соседний машрутизатор радостно устанавливает соединение по TCP порту 646 (спец для LDP). Так после HELLO у нас все соседние маршрутизаторы пооткрывали друг с другом соединения.

Теперь наш LSR1, видя по табл IP-маршрутизации, что к сетке идем через LSR2, по установленному соединению шлет тому LDP-запрос – НИЖЕ по потоку, как уже сказали. Естесно, что для других сетей у LSR1 другие LSR'ы ниже по потоку. А спрашиваем мы у того, кто ниже – потому что именно тот, кто ниже, будет использовать эту метку и она должна быть уникальной. Запрос LDP:

Запрос метки Длина сообщения Идентиф-р сообще Признак класса Идентиф-р нам нужен, чтоб отличать запросы и определять, на какой вообще ответ пришел

В общем, от LSR1 отправили запрос к LSR2, а тот отправляет к след хопу, если нет такой сетки. И т.д. – пока не дойдем до последнего на пути. А последний LSR знает, что он последний – ему это сказали откуда-то свыше – и назначает метку, которая у него еще не использована.

Ответ на запрос того же вида, что и запрос – ток 1ое поле «отображение метки», а в конце еще сама метка. LSR2 получает этот ответ, записывает себе и шлет дальше к LSR1 – тот тоже записывает и все счастивы.

Логично, что стараются к сеткам прокладывать маршруты так, чтоб их было немного – а не к каждой сетке новый маршрут

Дальше про независимое распределение меток – у нас же метки локальные – используются официально только в паре маршр-р – маршр-р, поэтому LSR2 может никого не спрашивать, а сразу ответить LSR1 меткой из своих неиспользованных, не дожидаясь, пока ему LSR3 ответит.
Распределение меток без запроса – LSR2 видит новую сетку, сам говорит для нее метку – и шлет сразу всем предупреждение – типа я для этой сетки эту метку взял. Так делают все – поэтому LSR может получить несколько разных меток разных путей для одной сетки – и выберет как ему понравится, исходя из своих моральных принципов.
В рамках одного сеанса LDP только один метод распределения меток. В рамках одной сети – могут использоваться оба метода.
Чаще всего LDP работает в режиме независимого управления распределением меток без запроса