1. Компьютерные сети - определение, составные части, назначение, топология, связность, виды сетей.

Компьютерная сеть - это система взаимодействующих компьютеров, которые могут обмениваться данными и ресурсами через различные каналы связи. Сеть может быть организована как в рамках одного учреждения или организации, так и на глобальном уровне, объединяя компьютеры по всему миру.

Составные части компьютерной сети:
Компьютеры - устройства, подключенные к сети, которые могут обмениваться данными и ресурсами.

Средства связи - это физические и логические каналы передачи данных, такие как кабели, оптические волокна, беспроводные каналы связи, сетевые карты и т.д.

Сетевое программное обеспечение - это набор программ и протоколов, которые позволяют компьютерам обмениваться данными и ресурсами.

Серверы - это выделенные компьютеры в сети, которые предоставляют различные услуги и ресурсы, такие как файлы, печать, электронную почту и т.д.

Сетевое хранилище - это устройства, которые предназначены для хранения и организации общих данных в сети.

Назначение компьютерных сетей - это обеспечение обмена данными и ресурсами между компьютерами в сети. Сети используются для обмена файлами, электронной почты, доступа к распределенным базам данных, удаленного управления и многим другим задачам.

Топология сети - это физическое расположение устройств в сети и способ их соединения.

Топология сети - это физическая или логическая структура, определяющая способ, которым устройства в сети связаны и обмениваются данными. В зависимости от того, как организована сеть, можно выделить несколько видов топологий:

- *Звезда (Star)* - все устройства подключены к центральному коммутатору или концентратору.
- *Шина (Bus)* - все устройства подключены к одной линии передачи данных.
- *Кольцо (Ring)* - устройства соединены между собой кольцевой линией.
- *Дерево (Tree)* - сеть имеет иерархическую структуру, где каждый уровень связан с родительским уровнем.
- *Сеть полной связности (Mesh)* - каждое устройство подключено напрямую ко всем другим устройствам в сети.
- *Гибридная (Hybrid)* - комбинация двух или более типов топологий, например, звездообразной и кольцевой.

Каждая из этих топологий имеет свои достоинства и недостатки, и выбор топологии зависит от конкретных потребностей и требований к сети.

Связность - это свойство сети, определяющее степень доступности каждого устройства в сети. Чем выше связность, тем более надежна сеть и лучше она способна передавать данные между компьютерами.

Виды компьютерных сетей:

- *Локальная сеть (LAN)* - это сеть, которая объединяет компьютеры и устройства в пределах ограниченной территории, такой как здание или офис.
- *Глобальная сеть (WAN)* - это сеть, которая объединяет компьютеры и устройства на больших расстояниях, обычно через Интернет.
- *Компьютерные кластеры* - это совокупность связанных компьютеров, используемых для выполнения одной задачи или решения одной проблемы.
- *Метрополитенская сеть (MAN)* - это сеть, которая объединяет компьютеры и устройства в пределах города.
- *Беспроводная сеть (WLAN)* - это сеть, которая использует беспроводные технологии для связи между компьютерами и устройствами.

Вместе с появлением компьютерных сетей появляются новые требования, такие как безопасность данных и защита от взлома, которые необходимо учитывать при проектировании и эксплуатации компьютерных сетей.

2. Сетевой трафик – определение, процесс передачи трафика по звену, скорость передачи, пропускная способность. Пропускная способность сети

Сетевой трафик - это общее название для всех данных, которые передаются в компьютерной сети. Это может быть любая информация, от почты и сообщений до файлов и мультимедийных данных.

Процесс передачи сетевого трафика по звену начинается с того, что данные отправляются с одного узла или компьютера через канал связи в сети. Затем эти данные проходят через несколько промежуточных устройств, таких как маршрутизаторы и коммутаторы, чтобы достичь конечного узла назначения.

Скорость передачи сетевого трафика определяется битами в секунду (bps), килобитами в секунду (kbps), мегабитами в секунду (Mbps) или гигабитами в секунду (Gbps). Скорость передачи зависит от типа канала связи, используемого для передачи данных, а также от общей загрузки сети.

Пропускная способность сети представляет собой максимальное количество данных, которые могут быть переданы через сеть за единицу времени. Она измеряется в битах в секунду (bps), килобитах в секунду (kbps), мегабитах в секунду (Mbps) или гигабитах в секунду (Gbps).

Пропускная способность зависит от нескольких факторов, таких как тип канала связи, топология сети, количество устройств в сети и уровень загрузки сети. Например, локальная сеть, использующая высокоскоростной кабель Ethernet, может иметь пропускную способность до 10 Gbps, тогда как широкополосный Интернет может иметь пропускную способность от нескольких мегабит в секунду до нескольких гигабит в секунду, в зависимости от типа подключения и провайдера услуг.

Важно понимать, что пропускная способность сети может быть ограничена на разных уровнях, например, на уровне локальной сети, маршрутизатора, провайдера услуг Интернета или ограничений, накладываемых программным обеспечением. Если пропускная способность сети недостаточна для передачи данных в режиме реального времени, это может привести к задержкам в передаче данных, потере пакетов и другим проблемам в работе сети. Поэтому, при проектировании и эксплуатации компьютерных сетей необходимо учитывать пропускную способность сети и ее потенциальное использование.

3. Основные сетевые механизмы: мультиплексирование каналов и коммутация. Принципы передачи трафика в сети. Режимы передачи: прямой, косвенный, транзит.

Мультиплексирование каналов и коммутация - это два основных механизма, которые используются для передачи данных в сетях.

Мультиплексирование каналов - это процесс объединения нескольких потоков данных в одном канале связи. Это позволяет использовать доступную полосу пропускания более эффективно и уменьшает затраты на создание новых физических каналов связи. Существует несколько методов мультиплексирования каналов, включая частотное, временное и кодовое мультиплексирование.

Частотное мультиплексирование (Frequency Division Multiplexing, FDM) - это метод передачи нескольких потоков данных через единую линию связи путем разделения частотного диапазона на несколько непересекающихся каналов. Каждый канал выделяется для конкретного потока данных и использует определенный диапазон частот. Это позволяет передавать несколько потоков данных одновременно без взаимного вмешательства.

Временное мультиплексирование (Time Division Multiplexing, TDM) - это метод передачи нескольких потоков данных через единую линию связи путем деления времени на периоды времени, которые выделяются для каждого потока данных. Каждый поток данных получает свой собственный временной интервал или слот, который выделяется на определенный промежуток времени. В каждый момент времени передается только один поток данных, но такая передача происходит настолько быстро, что для пользователя может создаться впечатление, что все данные передаются одновременно.

Кодовое мультиплексирование (Code Division Multiplexing, CDM) - это метод передачи нескольких потоков данных через единую линию связи путем использования уникальных кодов для каждого потока данных. Каждый поток получает свой собственный код, который используется для модуляции сигнала передачи. Приемник использует тот же код для демодуляции и извлечения соответствующего потока данных. Преимущество этого метода заключается в том, что все потоки данных могут быть переданы одновременно без необходимости выделения отдельных частот или временных интервалов.

Коммутация - это процесс перенаправления данных от отправителя к получателю через сеть. Есть три типа коммутации: сообщений, коммутация сообщений и каналов. В канальной коммутации выделяется связь между отправителем и получателем на время передачи данных. В пакетной коммутации данные разбиваются на пакеты, которые передаются через сеть независимо друг от друга. Каждый пакет содержит адрес получателя и информацию о порядке отправки. Таким образом, пакетная коммутация позволяет более эффективно использовать доступную полосу пропускания, т.к. не требует выделения связи на все время передачи данных.

Принципы передачи трафика в сети зависят от режима передачи, который используется в данной ситуации. Режим передачи может быть прямой, косвенный или транзитный.

- В *режиме прямой передачи* данные передаются от отправителя к получателю напрямую через выделенную связь. Этот режим используется, например, при звонке по телефону.
- В *режиме косвенной передачи* данные передаются через несколько узлов сети, каждый из которых перенаправляет данные следующему узлу до тех пор, пока они не достигнут получателя. Этот режим используется, например, при отправке электронной почты.
- В *режиме транзитной передачи* данные передаются через сеть, но не для доставки конечному получателю. Вместо этого они перенаправляются на другой узел сети для обработки или дальнейшей передачи. Такой режим передачи используется, например, при пересылке глобальных информационных пакетов в Интернете.

4. Способы статического мультиплексирования каналов: TDM, FDM. Описание принципов функционирования сетей с коммутацией каналов.

Статическое мультиплексирование - это процесс передачи нескольких потоков данных по одному физическому каналу. Существуют два основных способа статического мультиплексирования каналов: TDM (Time Division Multiplexing) и FDM (Frequency Division Multiplexing).

- *TDM* используется для передачи данных в последовательности, где каждый канал имеет определенный интервал времени для передачи данных. Каждый поток данных прерывается через равные промежутки времени, чтобы позволить другим потокам данных использовать канал. В результате, все потоки данных передаются последовательно на одном канале.
- *FDM* использует различные частотные диапазоны для передачи нескольких потоков данных. Каждый поток данных имеет свой уникальный частотный диапазон, который отличается от других потоков данных. Эти диапазоны разделены, чтобы не пересекаться с другими диапазонами потоков данных, что позволяет каждому потоку данных передаваться через физический канал параллельно.

Сети с коммутацией каналов функционируют путем выделения физического канала для передачи данных между двумя конечными пунктами. Этот физический канал может быть выделен на короткий или длительный период времени, в зависимости от необходимой скорости передачи данных. Когда устройство отправляет данные, оно резервирует необходимый канал для передачи этих данных до тех пор, пока передача не будет завершена. Это позволяет гарантировать доставку данных и предотвращает конфликты между различными потоками данных, которые используют один канал.

Принцип коммутации каналов заключается в выделении выделенного канала для каждого соединения в сети. Когда устройство отправляет данные, они передаются через выделенный канал до получателя, используя предопределенный маршрут.

При использовании *коммутации каналов* устройства должны сначала установить соединение посредством процесса вызова (call setup), который включает в себя набор номера получателя и запрос на выделение выделенного канала для передачи данных. Если соединение установлено успешно, то выделяется канал, которым будут передаваться данные между отправителем и получателем.

Однако если все выделенные каналы уже заняты или отсутствуют подходящие каналы для установления соединения, то процесс установки соединения может завершиться неудачей.

В процессе передачи данных через выделенный канал, скорость передачи данных определяется пропускной способностью канала, а пакеты данных передаются последовательно по выделенному каналу до получателя.

Коммутация каналов обеспечивает очень низкий уровень ошибок и потерь данных, что делает ее наиболее предпочтительной для передачи реального времени и других критически важных данных. Однако, она может быть менее эффективной по сравнению с другими методами коммутации, так как не использует доступные каналы эффективно, что может привести к значительным затратам на оборудование.

Если произошел разрыв соединения при коммутации каналов, то данные могут не доходить до получателя или быть поврежденными. В зависимости от конкретной ситуации и причины разрыва соединения, возможны следующие сценарии:

- *Данные не дошли* до получателя: если соединение было разорвано на стадии передачи данных, то они могут не доходить до получателя. Это может быть вызвано множеством причин, например, обрывом связи или ошибкой в процессе коммутации каналов.
- *Данные были повреждены:* в некоторых случаях данные могут достигнуть получателя, но могут быть повреждены в процессе передачи. Это может произойти из-за ошибок в процессе коммутации каналов, шума на линии связи, интерференции или других факторов.
- *Данные были потеряны*: если соединение было разорвано на стадии установки соединения, то данные могут быть потеряны. Это может произойти, если выделенный канал для передачи данных уже занят, или если возникла другая ошибка в процессе установки соединения.

В любом случае, если соединение было разорвано в процессе коммутации каналов, необходимо принимать меры для решения проблемы и обеспечения правильной передачи данных. Это может включать в себя повторную установку соединения, изменение маршрута, использование другого протокола коммутации или другие меры, зависящие от конкретной ситуации.

5. Статистическое мультиплексирование. Инкапсуляция. Принцип функционирования сетей с коммутацией сообщений. Буферизация. Коммутация Store and Forward.

Статистическое мультиплексирование - это метод передачи данных, который позволяет разделять полосу пропускания физического канала на различные потоки данных в зависимости от текущей загрузки. Он используется для оптимального использования доступной пропускной способности канала и увеличения эффективности передачи данных.

Инкапсуляция - это процесс добавления заголовков и/или некоторой дополнительной информации к передаваемым данным, чтобы обеспечить правильную адресацию и управление передачей данных. Это позволяет гарантировать, что данные будут доставлены по нужному маршруту и в правильном порядке.

Сети с коммутацией сообщений - это сети, в которых сообщения передаются через несколько узлов (например, маршрутизаторов) на пути от отправителя к получателю. При этом каждый узел изучает заголовок сообщения и решает, куда направить сообщение. Это позволяет использовать более сложные алгоритмы маршрутизации для оптимизации передачи данных.

Буферизация - это процесс временного хранения данных в буфере до того, как они будут переданы или обработаны. В сетях связи буферизация часто используется для устранения различных типов задержек, таких как задержки на передачу и задержки на обработку.

Задержки на передачу и обработку сообщений при коммутации сообщений могут быть вызваны различными причинами.

Причины задержек на передачу включают:

- *Ограниченная пропускная способность* канала связи, которая может привести к задержкам в передаче сообщений.
- *Перегруженность сетевого оборудования*, так как большое количество данных может привести к увеличению задержек на передачу.
- *Длинные маршруты*, поскольку каждое устройство на пути сообщения добавляет некоторую задержку.
- *Низкая скорость интернет-соединения*, что может замедлить скорость передачи данных.

Причины задержек на обработку сообщений могут включать:

- *Недостаточные ресурсы* устройства получателя, которые могут привести к замедлению обработки входящих сообщений.
- *Процесс проверки безопасности*, который может замедлить обработку сообщений для обеспечения безопасности передаваемых данных.
- *Обработка ошибок*, которая может занять дополнительное время для коррекции ошибок в сообщении.

Существует несколько методов для исправления задержек на передачу и обработку сообщений при коммутации сообщений:

- Использование сетевых протоколов, которые имеют более высокую производительность, например, TCP/IP, может помочь ускорить передачу данных.
- Оптимизация настроек сетевого оборудования, таких как роутеры, коммутаторы и маршрутизаторы, включая увеличение доступной пропускной способности и уменьшение количества устройств на пути сообщения, чтобы ускорить передачу данных.

- Использование кэширования, такое как использование Content Delivery Networks (CDN), может ускорить время обработки сообщений.
- Использование более мощных компьютеров и устройств получателя может ускорить обработку сообщений.
- Использование алгоритмов сжатия данных для уменьшения размера передаваемых сообщений может помочь сократить задержки при передаче данных.
- Регулярное мониторинг сети, чтобы выявлять и исправлять возможные проблемы, которые могут приводить к задержкам при коммутации сообщений.

Коммутация Store and Forward - это метод коммутации сообщений, при котором каждый пакет данных полностью принимается, сохраняется в буфере и затем пересылается на следующий узел сети. Этот метод гарантирует доставку целостного пакета данных и позволяет устранить ошибки связанные с возможными повреждениями пакетов данных.

В целом, статистическое мультиплексирование, инкапсуляция, сети с коммутацией сообщений, буферизация и коммутация Store and Forward являются эффективными методами передачи данных в сетях связи. Они повышают эффективность передачи данных, гарантируют правильную адресацию и управление передачей данных, а также помогают устранить задержки и ошибки связанные с передачей данных через сети.

6. Принцип коммутации пакетов и его характеристики по сравнению с коммутацией сообщений.

Принцип коммутации пакетов - это метод передачи данных, в котором каждый пакет данных отправляется по маршруту, который определяется на основе адреса назначения в заголовке пакета. Каждый пакет имеет свой уникальный идентификатор, который используется для сборки сообщения при получении пакетов.

Основными характеристиками коммутации пакетов являются:

- *Размер пакета:* пакеты имеют фиксированный размер, что позволяет эффективно использовать доступную пропускную способность канала связи.
- *Широковещательность*: пакеты могут быть отправлены широковещательно на все узлы сети или только на конкретные узлы, что позволяет доставлять данные только тем узлам, которые нуждаются в них.

Широковещательность - это метод передачи информации в компьютерных сетях, при котором сообщение отправляется на все устройства в определенной сети без необходимости указывать конкретного получателя. Это означает, что каждое устройство в сети получит копию сообщения, даже если оно не является адресатом.

Широковещание может быть полезным в многих случаях. Например, когда нужно отправить сообщение всем устройствам в локальной сети для обнаружения новых устройств или поиска определенных сервисов. Это также может использоваться для распространения событий или рассылки уведомлений о каких-либо изменениях в сети.

Однако *широковещание* имеет свои недостатки. При передаче сообщения на все устройства в сети возникает большая нагрузка на сеть, что может привести к заторможенности и падению производительности. Кроме того, поскольку сообщения доставляются на все устройства, они могут стать доступными для неавторизованных пользователей, что повышает уязвимость сети.

В целом, хотя *широковещание* может быть полезным инструментом в некоторых случаях, его использование должно быть ограничено и регулируемым для того, чтобы избежать проблем с производительностью и безопасностью.

- *Гарантированная доставка:* коммутация пакетов гарантирует доставку каждого пакета данных к его адресату, даже если некоторые из пакетов потеряются.

Коммутация пакетов - это метод передачи данных, который разбивает информацию на небольшие куски (пакеты) и отправляет каждый из них отдельно. При этом, каждый пакет может следовать своим маршрутом до получателя, поэтому возможны потери и задержки.

Однако, *коммутация пакетов* гарантирует доставку каждого пакета данных к его адресату, даже если некоторые из пакетов потеряются. Это происходит благодаря тому, что в коммутирующих устройствах (как, например, роутерах) используются методы контроля ошибок и повторной отправки пакетов.

Если какой-то пакет был потерян или поврежден при передаче, то он будет переслан снова или будет запрошена его повторная отправка. Это помогает обеспечить доставку всех пакетов данных до адресата.

Таким образом, коммутация пакетов позволяет эффективно передавать данные в сети, обеспечивая высокую степень надежности доставки, несмотря на возможные потери и ошибки при передаче пакетов.

- *Поддержка QoS (Quality of Service):* коммутация пакетов позволяет предоставлять различную пропускную способность для разных типов трафика, таких как голосовой, видео или данные.

Поддержка QoS (Quality of Service) означает, что сетевое оборудование или программное обеспечение способно контролировать и гарантировать уровень качества обслуживания для конкретного трафика в сети.

Когда на сеть накладывается высокая нагрузка, могут происходить задержки, потери пакетов и другие проблемы, которые могут повлиять на производительность приложений и качество пользовательского опыта. Поддержка QoS позволяет давать приоритет отдельным типам трафика, таким как голосовой или видео-трафик, чтобы обеспечить достаточную пропускную способность и минимизировать задержки для этих приложений.

Например, в голосовых сетях VoIP (Voice over Internet Protocol) поддержка QoS может быть использована для обеспечения достаточной ширины канала и минимальных задержек для голосового трафика, чтобы избежать рассинхронизации и эффекта эха.

Таким образом, поддержка QoS имеет большое значение для обеспечения надежности, производительности и качества обслуживания в сетях с высокой загрузкой и различными типами трафика.
В сравнении с коммутацией сообщений, принцип коммутации пакетов имеет следующие отличительные особенности:

- В коммутации пакетов каждый пакет может проходить по различным маршрутам, в зависимости от текущей загрузки сети, что позволяет более эффективно использовать доступную пропускную способность канала связи.

- Коммутация пакетов обеспечивает гарантированную доставку каждого пакета данных, даже если некоторые из пакетов потеряются, что обеспечивает более высокую надежность передачи данных.
- Коммутация пакетов поддерживает QoS, что позволяет предоставлять различные уровни пропускной способности для разных типов трафика, что повышает качество обслуживания пользователей.

Однако, коммутация сообщений имеет свои преимущества в определенных случаях, таких как при передаче аудио- и видеоданных с фиксированным размером пакета и требованиями к задержкам передачи.

Одним из примеров использования коммутации сообщений с фиксированным размером пакета и требованиями к задержкам передачи является передача потокового видео, например, в системах трансляции онлайн-видео или в видеоконференциях.

В таких приложениях каждый кадр видео разбивается на небольшие пакеты фиксированного размера, которые должны быть доставлены получателю в правильном порядке и с минимальной задержкой. Коммутация сообщений может обеспечить быструю и эффективную передачу таких потоков данных, поскольку фиксированный размер пакетов упрощает процесс маршрутизации и контроля ошибок.

Требования к задержкам передачи также важны для обеспечения высокого качества передаваемых данных. Например, при передаче видео в режиме реального времени, даже небольшая задержка может привести к заметному перекосу в аудио и видео сигналах, что может существенно повлиять на качество пользовательского опыта.

Таким образом, использование коммутации сообщений с фиксированным размером пакетов и контролем задержек может быть полезным для обеспечения высокого качества передачи потокового видео в системах онлайн-трансляций и видеоконференций.

7. Понятие протокола и его составные части: актеры, диалог, сообщения, локальные действия. Протокольные термины: спецификация, реализация, экземпляр, таймаут.

Протокол - это набор правил и процедур, которые определяют способ обмена информацией и управляют взаимодействием между различными устройствами или приложениями в сети. Протоколы используются для того, чтобы гарантировать корректность передачи данных и обеспечить совместимость между различными системами.

Составные части протокола:

- *Актеры (участники)* - это сущности, которые участвуют в обмене сообщениями по протоколу. Актеры могут быть как программными, так и аппаратными компонентами.
- *Диалог* - это последовательность сообщений, которые передаются между актерами в рамках протокола.
- *Сообщения* - это данные, которые передаются между актерами в рамках диалога. Сообщения содержат информацию о команде, запросе или ответе.
- *Локальные действия* - это действия, которые выполняются каждым актером в ответ на получение или отправку сообщений. Например, актер может сохранять полученные данные в свою локальную базу данных или вычислять результат на основе полученных запросов.

Протокольные термины:

- *Спецификация* - это набор требований к протоколу, который определяет его поведение и основные параметры. Спецификация описывает формат сообщений, правила обмена, требования по надежности и т.д.
- *Реализация* - это реализация протокола в виде программного кода или аппаратного устройства. Реализация должна соответствовать спецификации протокола, чтобы обеспечить совместимость между различными системами.
- *Экземпляр* - это конкретный экземпляр протокола, который запущен на определенных устройствах или приложениях. Каждый экземпляр протокола может иметь свои уникальные параметры и настройки.

Экземпляр протокола - это конкретный случай использования протокола в определенной ситуации, когда два или более устройства обмениваются данными в соответствии с правилами, заданными протоколом. Это может быть одноразовое соединение между двумя устройствами или длительная сессия обмена данными.

Примером экземпляра протокола может служить передача электронной почты через протокол SMTP (Simple Mail Transfer Protocol). В таком случае, для отправки сообщения используется экземпляр протокола SMTP, который включает информацию о сервере и его параметры, а также данные о самом сообщении (кому адресовано, тема, содержание и т.д.).

При передаче электронной почты через протокол SMTP, каждый компонент сообщения разбивается на пакеты данных, которые затем передаются по сети от отправителя к получателю. SMTP гарантирует доставку сообщения до получателя, а также обеспечивает контроль ошибок и возможность повторной отправки сообщения в случае неудачной попытки отправки.

Таким образом, экземпляр протокола SMTP позволяет пользователям отправлять электронную почту, используя стандартные правила и процедуры, заданные протоколом SMTP.

- *Таймаут* - это время ожидания ответа на запрос или сообщение перед тем, как актер считается недоступным. Если ответ не получен в течение заданного времени, то протокол может перейти в другое состояние или инициировать другую последовательность действий.

В зависимости от протокола и контекста, примерами последовательностей действий, которые могут быть инициированы при таймауте, являются:

- Повторная отправка запроса: Если ответ на запрос не получен в течение заданного времени, то протокол может повторить запрос для получения ответа еще раз. Это может происходить автоматически без участия пользователей.
- Закрытие сессии: Если актер не отвечает на запросы в течение определенного времени, протокол может закрыть текущую сессию с актером. Например, в случае клиент-серверных систем, сервер может закрыть соединение с клиентом, если не получает от него ответ в течение заранее определенного времени.

- Выдача ошибки: Если ответ на запрос не получен в течение заданного времени, протокол может выдать ошибку о таймауте. Это может произойти, например, если клиент не может подключиться к серверу в течение определенного периода времени.

- Изменение параметров: В некоторых случаях, при таймауте протокол может изменять свои параметры, чтобы адаптироваться к текущим условиям. Например, может происходить изменение частоты опроса или интервалов между запросами.

Таким образом, последовательность действий при таймауте зависит от типа протокола и ситуации, в которой он используется.

8. Способ изображения протокола. Роли актеров в рамках протокола. Модель «клиент-сервер».

Способ изображения протокола - это диаграмма, которая показывает последовательность сообщений, передаваемых между актерами в рамках протокола. Существует несколько способов изображения протокола, но наиболее распространенными являются диаграммы последовательностей и диаграммы состояний.

Диаграмма последовательности показывает последовательность сообщений, передаваемых между актерами в виде вертикальных линий (актеры) и горизонтальных стрелок (сообщения). Диаграмма состояний показывает состояния, через которые проходит каждый актер в рамках протокола и переходы между этими состояниями.

Роли актеров в рамках протокола могут быть различными в зависимости от конкретного протокола. Например, в протоколе HTTP (HyperText Transfer Protocol), актеры могут быть клиентом (браузером) и сервером (веб-сервером), а в протоколе SMTP (Simple Mail Transfer Protocol) - отправителем и получателем электронной почты.

Модель "клиент-сервер" - это модель взаимодействия между компьютерными системами, которая базируется на принципе распределенной обработки. В этой модели одна компьютерная система (сервер) предоставляет определенный сервис или ресурс для других систем (клиентов), которые запрашивают этот сервис или ресурс.

В рамках *модели "клиент-сервер"* клиент и сервер взаимодействуют по определенному протоколу, который определяет формат запросов и ответов, правила аутентификации и шифрования, а также другие параметры. Клиент и сервер могут быть расположены на разных компьютерах или даже в разных сетях, что позволяет эффективно использовать доступные ресурсы и увеличивать масштабируемость системы.

Модель *"клиент-сервер"* имеет несколько *преимущест*в, таких как:

- *Централизованное управление:* сервер является центральным элементом системы, который управляет доступом к ресурсам и контролирует работу клиентов.

- *Удобство использования*: клиенты могут получать доступ к ресурсам с помощью стандартизированных протоколов, что делает использование системы более удобным и простым.

- *Масштабируемость:* система может быть масштабирована путем добавления новых серверов или распределения нагрузки между существующими серверами, что повышает производительность и увеличивает ее масштабируемость.

9. Функционирования протокольных модулей в среде современных операционных систем. Изоляция приложений от аппаратных и системных средств. Понятие сервиса и API. Сетевой сервис в современных ОС. Понятие о модулях сетевого обеспечения (netware).

Протокольные модули - это программные компоненты, которые обеспечивают передачу данных между устройствами или приложениями в сети. Они работают в операционной системе и обрабатывают данные в соответствии с определенными правилами и стандартами, такими как TCP/IP, HTTP и другие.

В *современных операционных системах протокольные модули работают* в изолированных средах, чтобы предотвратить несанкционированный доступ к аппаратным и системным ресурсам. Это означает, что приложения не могут напрямую взаимодействовать с аппаратурой и системными ресурсами, а используют протокольные модули для передачи данных.

Такая *изоляция* позволяет повысить безопасность системы, поскольку предотвращает возможность злоумышленников получить несанкционированный доступ к аппаратуре и системным ресурсам. Кроме того, использование протокольных модулей упрощает написание приложений, поскольку разработчики могут использовать уже готовые стандарты и правила для передачи данных без необходимости создания своих собственных протоколов.

Сервис - это программный компонент, который выполняет определенную функцию в рамках операционной системы или приложения. Например, сервис может обеспечивать доступ к сетевым ресурсам или предоставлять функциональность для работы с файловой системой. Сервисы могут работать как локально на устройстве, так и удаленно через интернет.

API (Application Programming Interface) - это набор программных интерфейсов, позволяющих разработчикам создавать приложения, использующие сервисы ОС. API определяет формат запросов и ответов, которые приложения могут использовать для взаимодействия с сервисами. Обычно API представлены в виде библиотек, которые могут быть статическими или динамическими.

Сетевой сервис в современных ОС представляет собой программный компонент, который обеспечивает работу сетевых протоколов, например, TCP/IP. Сетевой сервис может также включать в себя различные утилиты и инструменты для анализа сетевых соединений, отладки и тестирования сетевых протоколов. Сетевые сервисы могут быть как частью самой ОС, так и установлены отдельно.

Модули сетевого обеспечения (например, netware) - это программные компоненты, которые позволяют реализовывать сетевые протоколы и функции в рамках операционной системы.

Netware-модули могут включать в себя драйверы сетевых интерфейсов, стеки протоколов, а также утилиты для управления сетевыми соединениями и настройки сетевых параметров.

В целом, современные операционные системы обеспечивают широкие возможности для работы с сетевыми протоколами и сервисами.

Это позволяет создавать высокопроизводительные и безопасные сетевые приложения для различных нужд, таких как обмен данными, удаленный доступ к ресурсам и управление сетевым оборудованием. В современных ОС также

присутствуют механизмы контроля за безопасностью и аудитом действий пользователей, которые обеспечивают дополнительную защиту от несанкционированного доступа.

При создании сетевых приложений важным элементом является выбор подходящего API и/или библиотеки для работы с сетью и сетевыми протоколами. Существует множество различных API и библиотек, которые могут использоваться для этих целей, например, POSIX-совместимые API, Winsock API для Windows, а также различные библиотеки, такие как libpcap для работы с сетевым трафиком и OpenSSL для шифрования данных.

Для управления сетевым оборудованием и конфигурации сетевых параметров используются специальные утилиты и интерфейсы командной строки. Например, в Linux можно использовать утилиты ifconfig и ip для настройки параметров сетевых интерфейсов и маршрутизации, а также iptables для настройки правил файрвола.

10. Внутреннее устройство экземпляра сетевого протокола: протокольные модули, сетевой сервис, логическое соединение, примитивы обмена данными, буферы.

Сетевой протокол - это стандартизированный набор правил и процедур, которые определяют, как устройства в сети обмениваются данными. Он представляет собой формат, который определяет, как данные должны быть упакованы, отправлены и получены другим устройством в сети.

Сетевые протоколы обеспечивают эффективную передачу данных между устройствами в сети и позволяют различным устройствам с разными характеристиками связаться друг с другом и обмениваться информацией. Например, наиболее распространенный сетевой протокол TCP/IP (Transmission Control Protocol/Internet Protocol) используется для передачи данных в Интернете. Он разбивает данные на пакеты, отправляет их через сеть и проверяет, что все пакеты успешно доставлены.

Зная сетевой протокол, устройства могут общаться друг с другом и передавать данные без ошибок. Поэтому знание сетевых протоколов очень важно для работы с сетями и интернетом.

Протокольные модули: Это функциональные блоки, реализующие логику работы конкретного сетевого протокола. Протокольные модули могут включать различные алгоритмы и механизмы для обработки данных, например, сжатие, шифрование, контроль целостности и т.д.

- *Сетевой сервис:* Это служба, которая предоставляет доступ к сетевому протоколу через определенный интерфейс. Сетевой сервис может включать в себя функции, такие как управление соединением, передача данных, обработка ошибок и т.д.

- *Логическое соединение*: Это абстрактное понятие, которое описывает установленное соединение между двумя устройствами. Логическое соединение может быть установлено на разных уровнях стека протоколов, и его параметры могут варьироваться в зависимости от конкретного протокола.

- *Примитивы обмена данными:* Это набор функций, которые используются для передачи данных между двумя устройствами, подключенными к сети. Примитивы обмена данными могут включать в себя функции для установки соединения, передачи данных, закрытия соединения и т.д.

- *Буферы:* Это временное хранилище данных, которые будут переданы через сеть. Буферы могут размещаться как на стороне отправителя, так и на стороне получателя, и их размер может быть настроен в зависимости от требований конкретного протокола.

В целом, внутреннее устройство экземпляра сетевого протокола представляет собой сложную систему, которая включает в себя множество компонентов и алгоритмов для эффективной передачи данных. Каждый из компонентов выполняет свою определенную функцию, что позволяет сетевому протоколу работать без сбоев и обеспечивать высокую скорость передачи данных в сети.

11. Логическая модель сетевого сервиса: «черный ящик», SAP, адресация, примитивы приема/передачи данных. Жизненный цикл SAP. Дейтаграммный и потокоориентированный режимы.

Логическая модель сетевого сервиса представляет собой абстрактное описание функций и интерфейсов, которые используются для обмена данными между устройствами в сети. В этой модели выделяют несколько компонентов:

- *"Черный ящик" (black box):* Это абстракция, которая описывает работу сетевого сервиса как единого целого без подробного описания его внутренней структуры и работы протоколов в нем. Таким образом, "черный ящик" скрывает от пользователя детали работы сетевого сервиса и предоставляет только необходимый функционал, который может быть использован для передачи данных.

- *Система адресации (Addressing System):* Это механизм, который обеспечивает уникальную идентификацию устройств в сети и позволяет направлять данные на нужные устройства. Адресация может быть произведена на разных уровнях стека протоколов, в зависимости от конкретной реализации сетевого сервиса.

- *Примитивы приема/передачи данных (Data Transfer Primitives):* Это функции, которые используются для передачи данных между устройствами. Примитивы приема/передачи данных могут включать в себя функции для установки соединения, передачи данных и закрытия соединения.

- *Структура адресного пространства (Service Access Point - SAP):* Это абстракция, которая описывает точку доступа к конкретному сервису в сетевом сервисе. SAP представляет собой структуру данных, которая содержит информацию о конкретном сервисе, такую как протокол и порт, который он использует. SAP может быть использован для установления логического соединения и передачи данных через этот соединение.

Жизненный цикл SAP включает следующие этапы:

В данных этапах сетевой протокол, используемый для обмена данными, зависит от конкретного вида передачи данных и настроек системы. Однако, некоторые из наиболее распространенных протоколов, которые могут использоваться на каждом этапе, перечислены ниже:

- *Создание SAP*: на этом этапе может использоваться протокол UDP (User Datagram Protocol) или TCP (Transmission Control Protocol), который позволяет создать новый SAP.

- *Установка соединения*: для установки логического соединения между устройствами может использоваться протокол TCP/IP.

- *Передача данных*: для передачи данных по установленному логическому соединению может использоваться протокол TCP/IP, который обеспечивает надежную и последовательную передачу данных.

- *Закрытие соединения*: после завершения передачи данных может использоваться протокол TCP/IP для закрытия логического соединения.

Конкретные протоколы, используемые на каждом этапе, могут различаться в зависимости от настроек системы и типа передаваемых данных. Например, если данные должны быть переданы в режиме реального времени, то может использоваться протокол *RTP (Real-time Transport Protocol*). Если данные должны быть переданы в зашифрованном виде, то может использоваться протокол *SSL/TLS*.

Дейтаграммный и потокоориентированный режимы - это два основных типа передачи данных в сетевых сервисах.

- В *дейтаграммном режиме* передачи данных данные разбиваются на небольшие пакеты, которые передаются самостоятельно и могут доставляться в произвольном порядке. Протоколы, работающие в дейтаграммном режиме, обеспечивают быструю передачу данных, но не гарантируют их целостность или доставку в определенном порядке.

- В *потокоориентированном режиме* передачи данных данные передаются как последовательные потоки байтов. Протоколы, работающие в потокоориентированном режиме, обеспечив гарантию целостности данных и их последовательную доставку в правильном порядке, но могут быть менее эффективными в быстрой передаче больших объемов данных. Выбор между дейтаграммным и потокоориентированным режимами зависит от конкретных требований к приложению и сетевому сервису.

Кроме того, в логической модели сетевого сервиса могут присутствовать дополнительные компоненты, такие как управление ошибками (Error Control), обеспечивающее корректное обнаружение и исправление ошибок при передаче данных, или управление потоком (Flow Control), поддерживающее оптимальное использование сетевых ресурсов и предотвращающее перегрузку сети.

В целом, логическая модель сетевого сервиса представляет собой абстракцию, которая позволяет разработчикам и пользователям описывать и понимать функциональность и интерфейсы сетевых сервисов без необходимости знать все детали их внутренней работы и протоколов. Это позволяет создавать более гибкие, расширяемые и удобные в использовании сетевые приложения и сервисы.

12. Специфические проблемы при реализации netware: функциональная совместимость, взаимозаменяемость, гибкая структура. Методы решения: послойная архитектура и стандартизация протоколов.

NetWare (сетевая операционная система компании Novell) была одной из первых сетевых операционных систем, которая предоставляла возможность обмена данными и ресурсами между компьютерами в локальной сети. Однако, при ее реализации возникали некоторые проблемы, связанные с *функциональной совместимостью*, взаимозаменяемости и гибкой структурой.

Одна из основных проблем, связанных с NetWare, заключается в том, что разные версии этой операционной системы могут иметь несовместимые между собой API или протоколы обмена данными. Это означает, что при переходе на новую версию NetWare возникают проблемы совместимости с уже существующими приложениями и сервисами. Для решения этой проблемы была использована послойная архитектура, которая позволяет разделить функциональность операционной системы на различные уровни (например, уровень протоколов, уровень интерфейсов и т.д.), каждый из которых может быть изменен или заменен независимо от других уровней. Таким образом, приложения и сервисы могут использовать только те протоколы или интерфейсы, которые соответствуют их требованиям, без необходимости знать детали работы других уровней.

Еще одна проблема при реализации NetWare связана с *взаимозаменяемостью компонентов* системы. Например, разные производители сетевых карт могут использовать различные драйвера и протоколы, что может привести к несовместимости или неправильной работе системы в целом. Для решения этой проблемы была использована стандартизация протоколов обмена данными, таких как TCP/IP и IPX/SPX, которые можно использовать на разных устройствах и платформах. Это позволяет сделать систему более гибкой и совместимой с различными компонентами.

Гибкая структура NetWare означает, что операционная система может быть настроена и расширена в соответствии с конкретными требованиями пользователей и организаций. Однако, это также может приводить к проблемам совместимости и управляемости системы. Чтобы решить эти проблемы, были использованы различные методы, такие как стандартизация интерфейсов и API, централизованное управление и мониторинг, а также использование модульной структуры, которая позволяет добавлять и удалять компоненты системы по мере необходимости.

Таким образом, для решения проблем при реализации NetWare были использованы методы послойной архитектуры и стандартизации протоколов обмена данными. Это позволило создать более гибкую и совместимую систему, которая может быть настроена и расширена в соответствии с конкретными потребностями пользователей и организаций.

13. Понятие протокольного стека. Структура протокольных стеков. Описание процесса передачи данных через стек протоколов. Преобразования данных: сегментирование, сборка, инкапсуляция, декапсуляция. Мультиплексирование протоколов и экземпляров протоколов.

Протокольный стек - это набор протоколов, расположенных последовательно друг за другом для передачи данных через сеть. Каждый протокол выполняет свою функцию и обрабатывает данные на определенном уровне в стеке протоколов. При передаче данных через сеть каждый уровень протокольного стека выполняет свои задачи, а затем передает данные следующему уровню.

Структура протокольных стеков обычно описывается как слои, где каждый слой отвечает за выполнение конкретной функции:

- *Физический слой (Physical layer)* - отвечает за передачу битов по физическому каналу связи.

- *Канальный слой (Data Link layer*) - обеспечивает надежную передачу данных между устройствами в сети.

- *Сетевой слой (Network layer)* - отвечает за маршрутизацию пакетов данных через сеть.

- *Транспортный слой (Transport layer)* - обеспечивает надежную передачу данных между приложениями на разных устройствах.

- *Сеансовый слой (Session layer)* - устанавливает, поддерживает и завершает соединение между приложениями.

- *Представительский слой (Presentation layer)* - обеспечивает преобразование данных в формат, понятный приложениям.

- *Прикладной слой (Application layer)* - обеспечивает доступ к сетевым сервисам и приложениям.

Конкретные протоколы, используемые на каждом уровне сетевой модели OSI (Open Systems Interconnection), могут различаться в зависимости от конкретной реализации сети, но обычно применяются следующие протоколы:

• Физический слой - Ethernet, Fast Ethernet, Gigabit Ethernet, USB, Bluetooth, Wi-Fi, HDMI.

• Канальный слой - Ethernet, Token Ring, FDDI, HDLC, PPP, Wi-Fi (802.11), Bluetooth.

• *Сетевой слой* - IP (Internet Protocol), ICMP (Internet Control Message Protocol), ARP (Address Resolution Protocol), OSPF (Open Shortest Path First), BGP (Border Gateway Protocol), RIP (Routing Information Protocol).

• *Транспортный слой* - TCP (Transmission Control Protocol), UDP (User Datagram Protocol), SCTP (Stream Control Transmission Protocol).

• *Сеансовый слой* - NetBIOS, SMB (Server Message Block), RPC (Remote Procedure Call), SQL (Structured Query Language), NFS (Network File System).

• *Представительский слой* - ASCII (American Standard Code for Information Interchange), EBCDIC (Extended Binary Coded Decimal Interchange Code), Unicode, MIME (Multipurpose Internet Mail Extensions), SSL/TLS (Secure Sockets Layer/Transport Layer Security).

• *Прикладной слой* - HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol), POP3 (Post Office Protocol version 3), IMAP (Internet Message Access Protocol), DNS (Domain Name System).

Процесс передачи данных через стек протоколов начинается с отправки данных на прикладном уровне. Данные проходят через все уровни стека протоколов, где осуществляются необходимые преобразования и обработка данных на каждом уровне. Например, на транспортном уровне данные могут быть разделены на пакеты или объединены из пакетов, а на сетевом уровне могут быть добавлены заголовки с адресами маршрутизации. Затем данные отправляются на физический уровень, где они конвертируются в биты и передаются по физическому каналу связи.

Преобразования данных, выполняемые на каждом уровне стека протоколов:

- *Сегментирование* - разделение данных на более мелкие блоки для передачи на транспортном уровне.
- *Сборка* – объединение фрагментов данных на транспортном уровне.
- *Инкапсуляция* - добавление заголовков и другой информации к данным на каждом уровне стека протоколов для передачи.
- *Декапсуляция* - удаление заголовков и другой информации с данных на каждом уровне стека протоколов после их получения.
- *Мультиплексирование протоколов и экземпляров протоколов* позволяет передавать несколько потоков данных одновременно через один физический канал связи. Это достигается за счет того, что каждый протокол имеет свой уникальный номер порта или идентификатор, который используется для маршрутизации данных на приемной стороне. Кроме того, в протокольном стеке можно использовать несколько экземпляров протоколов, где каждый экземпляр обрабатывает данные отдельно друг от друга, что позволяет изолировать различные потоки данных и предотвращать их взаимное влияние.

Протокольный стек является ключевой концепцией в сетевых технологиях и используется для передачи данных через локальные и глобальные сети. Правильная настройка протокольного стека является важной задачей при проектировании и настройке сетей. Необходимо учитывать особенности каждого слоя протокольного стека и выбирать протоколы, которые лучше всего подходят для решения определенных задач сетевой связи.

14. Эталонные модели сетевой архитектуры: ЭМВОС, Internet Protocol Suite (TCP/IP). Методология их создания. Перечисление и функциональное описание уровней этих моделей.

Эталонные модели сетевой архитектуры являются стандартом, описывающим, как компьютеры в сети должны взаимодействовать друг с другом. Два наиболее известных примера эталонных моделей сетевой архитектуры - это ЭМВОС и Internet Protocol Suite (TCP/IP).
ЭМВОС (OSI) - это международная эталонная модель сетевой архитектуры, разработанная Международной организацией по стандартизации (ISO). Модель состоит из семи уровней, каждый из которых выполняет определенную функцию при передаче данных между устройствами в сети.
Методология создания эталонных моделей заключается в том, что они разрабатываются комитетами экспертов, состоящими из инженеров и ученых, которые работают в области компьютерных сетей. Каждый уровень в эталонной модели определяет свою функциональность и назначение.

ЭМВОС (сокращение от английского "Enterprise Model for Very Open Systems") - это модель компьютерных сетей, которая описывает уровни функциональности и связи между ними. Уровни модели ЭМВОС для компьютерных сетей включают:
Уровни ЭМВОС:

- *Физический уровень* - отвечает за передачу битов данных по физической среде. Он определяет характеристики кабелей, интерфейсов и протоколов передачи данных.
- *Уровень канала* - обеспечивает безошибочную передачу данных между двумя устройствами на физическом уровне. Здесь решаются вопросы синхронизации передачи данных, контроля ошибок и управления доступом к каналу.
- *Сетевой уровень* - отвечает за коммутацию данных между различными сетями. На этом уровне работают маршрутизаторы и коммутаторы, которые позволяют настраивать маршруты передачи данных и управлять зонами доступа.
- *Транспортный уровень* - обеспечивает надежную и эффективную передачу данных между приложениями. На этом уровне используются протоколы, которые гарантируют доставку данных в правильном порядке и без потерь.

- *Уровень сеанса* - устанавливает и поддерживает соединения между приложениями на разных компьютерах. Этот уровень позволяет обеспечить надежность передачи данных и управлять сеансами связи.

- *Уровень представления* - отвечает за кодирование и декодирование данных, передаваемых между приложениями. На этом уровне определяется формат данных и протоколы их представления.

- *Прикладной уровень* - содержит приложения, работающие на компьютерах и использующие сеть для передачи данных.

Internet Protocol Suite (TCP/IP) - это стандартная модель сетевой архитектуры, которая используется в Интернете. Модель состоит из четырех уровней, каждый из которых выполняет определенную функцию при передаче данных между устройствами в сети.

Уровни модели TCP/IP:

- *Уровень доступа к сети*: этот уровень определяет, как данные передаются между устройствами в локальной сети.

- *Интернет-уровень:* этот уровень определяет, как данные передаются между удаленными сетями. Он использует протокол IP для маршрутизации данных по сети.

- *Транспортный уровень:* этот уровень обрабатывает передачу данных между приложениями на разных устройствах в сети. Он использует протоколы TCP и UDP для пересылки данных.

- *Прикладной уровень:* этот уровень обрабатывает приложения, которые используются для работы с данными.

Модель TCP/IP является более простой и популярной, чем модель ЭМВОС, поскольку она используется в Интернете и других сетях. Однако, модель ЭМВОС все еще широко используется в некоторых областях, таких как телекоммуникации.
Если вам необходимо выбрать между ЭМВОС и Internet Protocol Suite (TCP/IP) для использования в сети, то рекомендуется обратить внимание на следующие факторы:

- *Архитектура:* ЭМВОС имеет монолитную архитектуру, где все функции и службы расположены на центральном сервере, тогда как TCP/IP использует распределенную архитектуру, где функции и службы распределены по всей сети. Если вы планируете создавать сеть с большим числом узлов, то TCP/IP может быть более подходящим выбором.

- *Применение:* ЭМВОС была разработана для использования в локальных сетях, тогда как TCP/IP может быть использован на любом типе сети. Если вы планируете использовать сеть только для локальной связи, то ЭМВОС может быть более подходящим выбором.

- *Доступность:* TCP/IP является открытым стандартом, который доступен для использования и разработки любым желающим, тогда как ЭМВОС является закрытым продуктом, используемым только в ограниченном числе организаций. Если вы хотите иметь больше свободы в выборе оборудования и программного обеспечения, то TCP/IP может быть более подходящим выбором.

- *Надежность*: Обе модели имеют свои преимущества и недостатки в плане надежности. Однако, в целом, TCP/IP сегодня является более распространенным и широко используется в интернете и других сетевых приложениях, что говорит о его хорошей надежности.

В целом, выбор между ЭМВОС и TCP/IP зависит от конкретных потребностей вашей сети. Если у вас есть большое количество узлов, то TCP/IP может быть более подходящим выбором, тогда как если вы планируете использовать сеть только для локальной связи, то ЭМВОС может быть более удобным решением.

15. Сетевой протокол IPv4 и его базовые понятия: сеть (network), интер-сеть (internet), узел (host), интерфейс, multihomed host, шлюз. Виды сетей: P2P и multipoint. Режимы передачи: unicast, broadcast и multicast. Принцип передачи пакетов по интер-сети.

Протокол IPv4 является одним из наиболее распространенных сетевых протоколов, используемых в Интернете и других компьютерных сетях. Он предназначен для управления адресами и маршрутизации данных между устройствами внутри сети и между различными сетями (интер-сеть).

Сеть (network) в терминах IPv4 обычно описывается как группа узлов (host), связанных между собой через общие каналы передачи данных. Каждый узел имеет свой уникальный IP-адрес, который позволяет ему быть идентифицированным и общаться с другими узлами в этой сети.

Интер-сеть (internet) - это объединение нескольких локальных сетей в одну большую сеть. Каждая сеть в интер-сети может иметь свои собственные IP-адреса и свою собственную схему адресации, но все они должны соответствовать общим принципам IPv4, чтобы обеспечить корректную маршрутизацию данных между сетями.

Узел (host) - это устройство в сети, которое может отправлять и принимать данные. Это может быть компьютер, сервер или любое другое сетевое устройство.

Интерфейс - это средство подключения узла к сети, которое обычно представлено в виде сетевой карты или другого аналогичного устройства. Каждый интерфейс имеет свой собственный уникальный IP-адрес, который используется для идентификации узла в сети.

Multihomed host - это узел, который имеет более одного интерфейса и, следовательно, более чем один IP-адрес. Это может быть полезно для обеспечения надежности и повышения производительности связи между узлами.

Когда устройство имеет только один сетевой интерфейс и подключено к одной сети, оно зависит от этой сети для связи с другими устройствами и ресурсами. Если эта сеть выходит из строя, то устройство теряет связь с другими устройствами и перестает функционировать.

В случае с *multihomed host*, устройство имеет несколько сетевых интерфейсов, каждый из которых может быть подключен к разным сетям. Это позволяет использовать несколько путей для связи с другими устройствами и ресурсами, что повышает надежность и доступность соединений.

Например, если одна из сетей выходит из строя, устройство может продолжать функционировать, используя другую сеть. Также *multihomed host* может использоваться для повышения скорости передачи данных, например, при использовании двух сетевых интерфейсов параллельно.

Multihomed host также может быть использован для разделения сетевых трафиков, например, для разделения сети на группы по сетевой безопасности или для разделения рабочих и личных сетей.

В целом, *multihomed host* предоставляет более высокий уровень доступности и надежности соединений, а также большую гибкость в управлении сетью.

Шлюз (gateway) - это устройство, которое используется для соединения двух или более сетей разных типов. Шлюз позволяет узлам в одной сети общаться с узлами в другой сети, пересылая данные между этими сетями.

Когда узел в одной сети хочет отправить данные на узел в другой сети, он посылает эти данные своему шлюзу. Шлюз принимает данные, проверяет их адрес назначения и определяет, в какую сеть нужно отправить данные. Затем шлюз пересылает данные в сеть назначения, где они доставляются до узла-получателя.

Процесс пересылки данных между сетями возможен благодаря тому, что шлюз имеет минимум два сетевых интерфейса, каждый из которых подключен к разным сетям. Шлюз может использовать информацию, содержащуюся в заголовках пакетов данных, чтобы определить, какие данные должны быть пересланы в какую сеть.

Для обеспечения безопасности и контроля над передаваемыми данными, шлюз также может выполнять функции маршрутизации, фильтрации пакетов по адресам и протоколам, аутентификации и шифрования данных. Это позволяет создавать более надежные и безопасные сетевые соединения между различными узлами и сетями.

В целом, использование шлюза позволяет узлам в разных сетях общаться друг с другом и обеспечивает эффективную передачу данных между различными сетями.

Существует два типа сетей в терминах IPv4 - это P2P (point-to-point) сети и multipoint (многоточечные) сети.

P2P (point-to-point) сеть является прямой связью между двумя узлами. Это обычно соединение двух компьютеров через кабель, как в случае с использованием Ethernet-кабеля.

Multipoint (многоточечные) сети - это сеть, которая имеет более одного узла и используется для обмена данными между ними. Такие сети часто используются в офисах, где несколько компьютеров подключены через один роутер или коммутатор.

Существует три режима передачи данных в терминах IPv4 - unicast, broadcast и multicast.

Unicast - это режим передачи данных, при котором данные отправляются только на один конкретный узел в сети.

Broadcast - это режим передачи данных, при котором данные отправляются на все узлы в данной сети. Этот режим используется для широковещательных сообщений, таких как запросы ARP (Address Resolution Protocol) или DHCP (Dynamic Host Configuration Protocol).

Multicast - это режим передачи данных, при котором данные отправляются на группу узлов в сети, которые были заранее выбраны для этого. Каждая группа имеет свой уникальный IP-адрес, который используется для идентификации группы узлов. Этот режим используется для потокового мультимедийного контента, онлайн-игр и других приложений, когда несколько узлов должны получать одинаковый поток данных.

Принцип передачи пакетов по интер-сети в IPv4 основан на использовании маршрутизаторов. Каждый маршрутизатор имеет таблицу маршрутизации, которая содержит информацию о том, какие сети доступны через данный маршрутизатор и какие маршруты лучше использовать для доставки пакетов. Когда пакет отправляется из одной сети в другую, он проходит через различные маршрутизаторы на пути к конечному узлу.

Каждый пакет содержит IP-адрес источника и назначения, а также другую информацию о пакете. Когда маршрутизатор получает пакет, он сравнивает IP-адрес назначения с таблицей маршрутизации и отправляет пакет по соответствующему маршруту. Если нужный маршрут не найден, маршрутизатор может либо отбросить пакет, либо отправить его на шлюз по умолчанию для дальнейшей обработки.

В целом, IPv4 является важным протоколом для управления адресами и маршрутизации данных в компьютерных сетях, и его принципы работы играют ключевую роль в функционировании Интернета и других сетевых систем.

16. Адресация в сетях IPv4: назначение, адресуемый элемент, структура адресов IPv4, способ записи. Диапазоны и блоки IP адресов. Свойства адресов, образующих блок, префиксы и суффиксы. Способы записи блоков адресов: netmask и CIDR.

Адресация в сетях IPv4 используется для идентификации устройств, подключенных к компьютерной сети, а также для маршрутизации пакетов данных между этими устройствами. В IPv4 каждое устройство в сети имеет уникальный 32-битный адрес (четыре октета), который записывается в десятичной системе счисления.

IPv4 адрес состоит из двух частей: сетевой и хостовой. Сетевая часть адреса определяет сеть, к которой принадлежит устройство, а хостовая - устройство внутри этой сети. Для разделения сетевой и хостовой частей используется специальная маска подсети, которая указывается после IPv4-адреса и содержит набор единиц и нулей.

Адрес IPv4 *записывается* в виде четырех чисел, разделенных точками. Каждое число представляет собой значение байта (8 бит), и может быть от 0 до 255. Например, адрес 192.168.1.1 является допустимым IPv4-адресом

Структура адресов IPv4 выглядит следующим образом:

XXX.XXX.XXX.XXX

Где каждый XXX представляет один октет адреса, который может принимать значения от 0 до 255. Например, адрес 192.168.1.1 является допустимым IPv4-адресом.

IPv4-адреса можно разделить на группы, или диапазоны, в зависимости от их назначения. Некоторые наиболее распространенные диапазоны IPv4-адресов включают:

Существует несколько наиболее распространенных *диапазонов IP-адресов*, которые используются для различных целей. Рассмотрим каждый из них более подробно:

- Приватный диапазон адресов 10.0.0.0 - 10.255.255.255

Этот диапазон адресов используется для приватных сетей и не маршрутизируется в Интернете. Это означает, что устройства, подключенные к сети с такими адресами, не могут быть напрямую доступны из глобальной Интернет-сети, а только из локальных сетей, использующих тот же диапазон.

- Приватный диапазон адресов 172.16.0.0 - 172.31.255.255

Этот диапазон адресов также используется для приватных сетей и не маршрутизируется в Интернете. Он предназначен для использования в больших организациях, которые нуждаются в большем количестве адресов, чем доступно в диапазоне 10.0.0.0/8.

- Приватный диапазон адресов 192.168.0.0 - 192.168.255.255

Этот диапазон адресов также используется для приватных сетей и не маршрутизируется в Интернете. Он широко используется в домашних и небольших офисных сетях, поскольку предоставляет достаточное количество адресов для таких сетей.

- Диапазон адресов 169.254.0.0 - 169.254.255.255 (APIPA - Automatic Private IP Addressing)

Этот диапазон адресов используется, когда устройство не может получить IP-адрес от DHCP-сервера. В этом случае устройство автоматически назначает себе адрес из этого диапазона, чтобы поддерживать связь в локальной сети.

APIPA - это сокращение от Automatic Private IP Addressing. Это технология автоматического назначения адреса в локальной компьютерной сети, которая используется, когда устройство не может получить IP-адрес от DHCP-сервера. Если компьютер не может связаться с DHCP-сервером, то он автоматически присваивает себе IP-адрес из диапазона 169.254.0.0/16 (от 169.254.0.0 до 169.254.255.255), который является зарезервированным для APIPA. При использовании APIPA устройства могут продолжать работать в локальной сети, не зависимо от наличия DHCP-сервера, что обеспечивает повышенную отказоустойчивость и надежность сети. Однако, так как адреса APIPA не уникальны в глобальной Интернет-сети, они не могут быть использованы для доступа к ресурсам в Интернете или на других сетях.

Отказоустойчивость - это способность системы продолжать работу в случае ее частичного или полного отказа. Она означает, что система может сохранять свою функциональность и доступность для пользователей даже при возникновении непредвиденных сбоев, ошибок или атак.

DHCP-сервер (Dynamic Host Configuration Protocol) - это сервер, который автоматически назначает IP-адреса и другие настройки сетевых параметров устройствам в компьютерной сети. DHCP-сервер позволяет управлять динамическим распределением IP-адресов, что упрощает конфигурацию сетевых устройств и уменьшает вероятность ошибок при настройке сети вручную.

DHCP-серверы используются практически во всех типах сетей, от локальных домашних сетей до крупных корпоративных сетей. Когда устройство подключается к сети, оно отправляет запрос на получение IP-адреса на DHCP-сервер, который затем назначает ему свободный адрес из пула доступных адресов. Кроме того, DHCP-сервер может также предоставлять информацию о шлюзе по умолчанию, DNS-серверах и других параметрах сети.

Каждый из этих диапазонов имеет свои особенности и предназначен для определенных целей. Приватные диапазоны адресов обеспечивают безопасность и конфиденциальность данных в локальных сетях, а диапазон APIPA помогает устройствам продолжать работу в сети даже в случае отсутствия доступного DHCP-сервера.

Диапазоны IP-адресов, которые используются для выхода в Интернет, называются *публичными адресами*. Эти адреса представляют собой уникальные идентификаторы, которые могут быть использованы для доступа к ресурсам в Интернете.

Публичные IP-адреса, как правило, выдаются провайдерами интернет-услуг, которые резервируют определенный диапазон адресов для своих клиентов. Некоторые из наиболее распространенных диапазонов публичных IP-адресов, которые используются для выхода в Интернет, включают:

- 126.255.255.255
- 128.0.0.0 - 191.255.255.255
- 192.0.0.0 - 223.255.255.255

Эти диапазоны IP-адресов были выделены Интернет-ассигнационным блоком (IANA) для использования в публичных сетях и доступа к Интернет.

Но следует отметить, что эти диапазоны IP-адресов не являются единственными публичными адресами, используемыми для выхода в Интернет. Провайдеры интернет-услуг могут использовать и другие диапазоны IP-адресов в зависимости от региона, страны или других факторов.

IPv4-адреса также могут быть объединены в блоки, которые представляют собой набор адресов, которые могут быть использованы в одной или нескольких сетях. Блок адресов может быть записан в виде начального и конечного адресов или в виде адреса и маски подсети.

Адреса, образующие блок, имеют ряд свойств:

- Все адреса в блоке имеют одинаковый префикс (начальную часть), который определяет сеть.

- Размер блока задается количеством бит, которые занимает префикс.

- Количество адресов в блоке можно вычислить как 2 в степени числа бит в хостовой части.

- Адреса, образующие блок, используются для распределения IP-адресов на подсети.
- Блоки могут быть разделены на более мелкие блоки при необходимости увеличения количества доступных адресов или уменьшения размера подсетей.
- Блоки IPv4 адресов могут быть классифицированы в соответствии с их размером и назначением (например, A, B, C и т.д.).
- Блоки могут быть использованы для организации VLAN (Virtual Local Area Network) в сетях Ethernet.

VLAN (Virtual Local Area Network) - это технология, которая позволяет разделить одну физическую сеть на несколько логических, независимых друг от друга виртуальных сетей.

В каждой VLAN может быть своя собственная настройка безопасности и управления доступом. VLAN позволяют группировать устройства в сети по функциональной принадлежности, а не только по физическому расположению в сети.

Маска подсети - это набор битов, которые определяют, какие биты IPv4-адреса являются сетевыми, а какие - хостовыми. Маска подсети записывается так же, как и IPv4-адрес, в виде 32-битного числа, которое состоит из четырех октетов (например, 255.255.255.0). Эта маска означает, что первые 24 бита IPv4-адреса являются сетевыми, а последние 8 бит - хостовыми.

Префикс и суффикс - это термины, которые используются для обозначения частей блока адресов. Префикс - это часть блока адресов, которая определяет сеть, к которой принадлежат устройства. Суффикс - это часть блока адресов, которая определяет количество доступных хостов в этой сети.

Пример записи блока адресов: 192.168.1.0/24.
Здесь 192.168.1.0 является адресом сети, а /24 - маской подсети, указывающей, что первые 24 бита IPv4-адреса являются сетевыми, а оставшиеся 8 бит - хостовыми. Это означает, что в данном блоке адресов может быть до 256 доступных хостов (2^8), так как последний октет адреса используется для идентификации конкретного устройства.

IPv4-адресация является одним из ключевых компонентов компьютерных сетей. Она позволяет устройствам обмениваться данными между собой и использовать Интернет-ресурсы. Для правильной настройки сети необходимо понимание структуры IPv4-адресов, префиксов и масок подсетей, а также способов их использования при настройке маршрутизации данных в сети.

Существуют два основных способа записи блоков адресов в IP-сетях - это запись netmask и запись CIDR.

Запись netmask (маска сети)

Netmask представляет собой битовую маску, которая определяет диапазон IP-адресов, принадлежащих заданной сети. Она записывается в виде четырёх чисел от 0 до 255, разделённых точками, например: 255.255.255.0. Каждый из этих чисел представляет собой количество битов, которые используются для определения адреса сети. Таким образом, маска сети 255.255.255.0 означает, что первые 24 бита адреса используются для определения адреса сети, а оставшиеся 8 битов - для определения адресов устройств внутри этой сети.
В записи *Netmask*, количество первых бит, отведенных под сетевую часть IP-адреса, определяется значением маски, где единицы обозначают сетевую часть, а нули - хостовую.

Например, в маске "255.255.255.0" (в двоичной системе - "11111111.11111111.11111111.00000000") первые 24 бита (три октета) отведены под сетевую часть, а последние 8 бит (один октет) отведены под хостовую часть. Таким образом, в данном случае количество первых бит, отведенных под сетевую часть, равно 24.

Запись CIDR (Classless Inter-Domain Routing)

CIDR - это метод записи диапазона IP-адресов в более компактной форме. Он состоит из IP-адреса и числа, которое обозначает количество битов, используемых для определения адреса сети. Например, 192.168.1.0/24 означает, что первые 24 бита адреса используются для определения адреса сети. В этой записи число после слеша (/) обозначает количество битов в маске сети.

CIDR-запись может быть использована для записи любого диапазона IP-адресов, включая как одиночные адреса, так и блоки адресов различных размеров. Например, 192.168.1.0/24 представляет собой блок из 256 адресов (от 192.168.1.0 до 192.168.1.255), а 10.10.0.0/16 - блок из 65 536 адресов (от 10.10.0.0 до 10.10.255.255).

В целом, CIDR-запись более удобна и компактна, чем запись netmask, поэтому она широко используется в современных сетях. Однако, для понимания работы сетей, важно иметь понимание и описательную способность обеих систем записи блоков адресов.

CIDR - это аббревиатура от английского выражения "Classless Inter-Domain Routing", что можно перевести как "Безклассовая междоменная маршрутизация". Это метод разбиения IP-адресов на более мелкие подсети без использования классов адресации, который был применяем в ранних версиях протокола IP. Использование CIDR позволяет более гибко и эффективно использовать доступные адреса IP, улучшает масштабируемость сетей и повышает эффективность маршрутизации.

Класс адресации - это метод, используемый для разделения IP-адресов на группы, которые затем могут использоваться для определения подсетей и маршрутизации трафика в сети. Ранее существовали три класса адресации: класс A, класс B и класс C. Каждый класс имел определенный диапазон IP-адресов, который был предназначен для определенных типов сетей.

17. Виды адресов IPv4: обычные и особые. Разновидности обычных адресов: глобально уникальные (белые), частные (серые) и специального назначения. Метод обеспечения уникальности «белых» IP адресов: IANA->RIR->ISP->потребитель. Правила назначения блоков адресов на сети.

Pv4-адреса могут быть разделены на две категории: обычные и особые.

Обычные адреса - это IPv4-адреса, которые используются для идентификации конкретного устройства в сети и для маршрутизации данных в Интернете. Обычные адреса могут быть разделены на три категории: глобально уникальные (белые), частные (серые) и специального назначения.

1.1. Глобально уникальные (белые) адреса

Глобально уникальные адреса, также известные как белые адреса, являются уникальными в рамках всего Интернета. Они предоставляются организацией IANA (Internet Assigned Numbers Authority) и распределяются региональными регистраторами интернет-ресурсов (RIR), которые затем передают эти адреса поставщикам услуг Интернета (ISP) и другим сторонам, нуждающимся в них. Затем ISP назначает эти адреса своим клиентам.

1.2. Частные (серые) адреса

Частные адреса, также называемые серыми адресами, используются в локальных сетях и не могут быть использованы для доступа к Интернету напрямую. Они предназначены для использования внутри сети, и служат для идентификации устройств в локальной сети. Частные адреса подразделяются на три диапазона: 10.0.0.0 до 10.255.255.255, 172.16.0.0 до 172.31.255.255 и 192.168.0.0 до 192.168.255.255.

1.3. Специального назначения адреса

Существует ряд IPv4-адресов, которые зарезервированы для специальных нужд. Некоторые из них используются для тестирования и отладки сетей, а другие - для определенных задач, таких как маршрутизация или мультикаст.

Задача мультикаста - это передача одного и того же сообщения (данных) от одного отправителя к нескольким получателям одновременно в рамках сети. В отличие от unicast, где данные передаются только одному адресату, и broadcast, где данные передаются всем узлам сети, multicast позволяет выбрать определенную группу узлов для передачи данных. Задача мультикаста широко используется в различных приложениях, таких как видеоконференции, онлайн-игры, потоковое видео и другие.

Особые адреса - это адреса, которые используются для специальных целей, например, для широковещательной передачи данных или для идентификации устройства, на котором выполняется запрос. Они не используются для идентификации конкретного устройства в сети и не могут быть назначены устройствам в качестве обычных адресов.

Широковещание (broadcast) - это метод передачи данных в компьютерных сетях, при котором сообщение отправляется одним узлом и доставляется всем остальным узлам в сети. Данные отправляются на специальный адрес, который называется широковещательным адресом (broadcast address), например, 255.255.255.255 для IPv4.

Метод обеспечения уникальности «белых» IP-адресов

- Процесс распределения IP-адресов начинается с *выделения блоков адресов организацией IANA* (Internet Assigned Numbers Authority), которая ответственна за глобальное управление IP-адресами.
- Затем эти блоки адресов передаются региональным реестрам интернет-адресов (RIR - Regional Internet Registries):
- *RIPE* Network Coordination Centre (RIPE NCC) для Европы, Ближнего Востока и некоторых стран бывшего Советского Союза;
- *RIR* затем *распределяет эти блоки адресов между поставщиками услуг Интернет (ISP - Internet Service Provider). ISP* получают блоки адресов от RIR и используют их для своих потребностей. Например, один ISP может использовать блок адресов для сети своих клиентов.
- *Клиенты провайдеров, в свою очередь, получают IP-адреса от своих поставщиков услуг Интернет и используют их для подключения к сети*, например, через свой домашний роутер или компьютер.

Таким образом, процесс распределения IP-адресов включает несколько уровней, начиная с глобальных организаций и заканчивая конечными пользователями. Каждый уровень отвечает за свою часть процесса и обеспечивает уникальность IP-адресов в пределах своей зоны ответственности.

Правила назначения блоков адресов на сети зависят от региона, в котором находится организация или пользователь, нуждающийся в назначении IP-адреса. Обычно, ISP выделяют своим клиентам блоки адресов, достаточных для их потребностей. Это может быть блок из нескольких IPv4-адресов для домашнего пользователя или блок из тысяч IPv4-адресов для крупной компании.

В случае частных (серых) IP-адресов, правила использования не так жесткие. Частные адреса могут использоваться в любой локальной сети без необходимости получения их от IANA, RIR или ISP. Любая организация может использовать частные адреса для своих локальных сетей. Рекомендуется использовать частные адреса только для локальных сетей и использовать механизм Network Address Translation (NAT) для доступа к Интернету.

Механизм NAT - это способ, который позволяет устройствам в локальной сети выходить в Интернет через единственный общий IP-адрес.

Вот как это работает: когда ваше устройство отправляет запрос на сервер в Интернете, оно использует свой локальный IP-адрес (например, 192.168.1.2). Но этот адрес не уникален и не может быть использован для доступа к Интернету.

Важно отметить, что наличие уникального IPv4-адреса становится все более необходимым, поскольку число доступных адресов быстро исчерпывается. Поэтому большинство провайдеров интернет-услуг переходят на использование IPv6, которые обеспечивают гораздо больший адресный пространство, позволяя значительно расширить возможности подключения к Интернету.

18. Структура пакета IPv4 и назначение его основных полей (какую функцию они выполняют).

IPv4 (Internet Protocol version 4) является протоколом маршрутизации пакетов данных в Интернете. Пакет IPv4 состоит из заголовка и полезной нагрузки.

Заголовок IPv4 имеет фиксированную структуру и содержит следующие поля:

- *Версия (Version)* - определяет версию протокола (IPv4 или IPv6). Размер поля 4 бита.
- *Длина заголовка (Header Length)* - указывает размер заголовка. Размер поля 4 бита.
- *Тип сервиса (Type of Service)* - используется для определения приоритета пакета и качества обслуживания. Размер поля 8 бит.

Приоритет пакета и качество обслуживания (Quality of Service, QoS) - это параметры, которые используются для управления трафиком в сети, чтобы обеспечить более эффективное использование ресурсов.

Существует несколько типов приоритетов пакетов, включая:

- *High Priority (Высокий приоритет):* такие пакеты получают наивысший приоритет и должны быть обработаны первыми. Это может включать голосовой или видео трафик, который требует низкой задержки и минимального пакетной потери.
- *Normal Priority* (Нормальный приоритет): это наиболее распространенный тип приоритета пакетов. Они не имеют особого приоритета и обрабатываются по мере возможности. Данные с Normal Priority могут включать электронные письма, файлы, загрузки веб-страниц и другие типы трафика, которые не требуют быстрой обработки и доставки.
- *Low Priority* (Низкий приоритет): такие пакеты получают наименьший приоритет и могут быть отложены до обработки других пакетов высшего приоритета. Данные с Low Priority могут включать резервирование пропускной способности для создания резервной копии данных, обновления программного обеспечения или любые другие процессы, которые не критически зависят от времени. В случае низкого приоритета такие данные будут обрабатываться после всех остальных пакетов, что может привести к задержкам.

Качество обслуживания определяется политиками управления трафиком, которые могут учитывать приоритетность пакетов, скорость передачи и другие факторы.

Некоторые типы качества обслуживания включают:

- *Best Effort* (Лучшие усилия): это означает, что пакеты обрабатываются по мере возможности без какого-либо приоритета.
- *Guaranteed* (Гарантированное): пакеты с гарантированным качеством обслуживания будут обработаны первыми и иметь наименьшую задержку при передаче данных в сети.
- *Controlled Load* (Контролируемая нагрузка): это политика управления трафиком, которая предоставляет определенный уровень качества обслуживания для определенного потока трафика, чтобы минимизировать задержки и потери пакетов.

Различные типы приоритетов пакетов и качества обслуживания используются для оптимизации производительности сети в зависимости от ее конкретных требований.

- *Длина пакета (Total Length)* - указывает общий размер пакета (заголовок + полезная нагрузка) в октетах. Размер поля 16 бит.
- *Идентификатор (Identification)* - уникальный идентификатор пакета, используемый для объединения фрагментированных пакетов. Размер поля 16 бит.

В сетевой связи, пакеты данных могут быть разделены на фрагменты для передачи по каналу связи.

Идентификатор (Identification) - это число или последовательность символов, которая уникально идентифицирует каждый пакет в сети. Этот идентификатор используется для объединения фрагментированных пакетов и установления связи между ними при доставке. Без использования идентификатора, фрагментированные пакеты могут быть перепутаны при доставке, что может привести к ошибкам и потере данных.

- *Флаги (Flags)* - используются для указания наличия фрагментации пакета и определяют положение фрагмента в исходном пакете. Размер поля 3 бита.
- *Смещение фрагмента (Fragment Offset)* - смещение фрагмента относительно начала пакета в 8-байтных блоках. Размер поля 13 бит.
- *Время жизни (Time to Live)* - задает время жизни пакета (количество промежуточных маршрутизаторов, которые могут обработать пакет). Размер поля 8 бит.
- *Протокол (Protocol)* - указывает протокол верхнего уровня (TCP, UDP, ICMP), который используется для передачи данных внутри пакета. Размер поля 8 бит.
- *Контрольная сумма заголовка (Header Checksum)* - используется для проверки целостности заголовка пакета. Размер поля 16 бит.
- *IP-адрес источника (Source IP Address)* - IP-адрес отправителя пакета. Размер поля 32 бита.
- *IP-адрес назначения (Destination IP Address)* - IP-адрес получателя пакета. Размер поля 32 бита.
- После заголовка следует *полезная нагрузка*, которая может содержать данные прикладного уровня (например, HTTP, FTP, SMTP).

Таким образом, каждое поле в заголовке IPv4 выполняет определенную функцию, необходимую для маршрутизации и доставки пакета в сети Интернет.

19. Коммутация пакетов IPv4: таблицы маршрутов, алгоритм коммутации, варианты действий, выполняемых IP модулем.

Коммутация пакетов IPv4 - это процесс передачи пакетов между узлами сети, который осуществляется на уровне сетевого протокола. Для того чтобы пакеты могли быть доставлены от отправителя к получателю, необходимо использовать таблицы маршрутов и алгоритм коммутации.

Таблицы маршрутов содержат информацию о том, каким образом следует доставлять пакеты в конечный узел. В таблице маршрутов для каждого возможного адреса назначения указывается следующий шаг, который должен выполнить коммутатор для передачи пакета. Например, если пакет должен быть доставлен из одной сети в другую, то в таблице маршрутов будет указан адрес следующего коммутатора, через который должен быть передан пакет.

Алгоритм коммутации включает в себя несколько этапов:

- *Получение пакета* - коммутатор получает пакет на свой интерфейс и проверяет его заголовки.
- *Определение адреса назначения* - коммутатор проверяет адрес назначения пакета и сравнивает его с записями в таблице маршрутов.
- *Выбор порта* - на основе информации из таблицы маршрутов коммутатор выбирает порт, через который должен быть передан пакет.
- *Отправка пакета* - коммутатор отправляет пакет на выбранный порт и переходит к следующему пакету.

В зависимости от ситуации, IP модуль может выполнить одно из нескольких действий:

- *Прямая доставка* - если адрес назначения пакета принадлежит устройству, на котором работает коммутатор, то пакет передается напрямую, без поиска в таблице маршрутов.
- *Доставка по подсети* - если адрес назначения пакета принадлежит устройству, находящемуся в той же подсети, что и отправитель, то коммутатор использует таблицу маршрутов для определения порта, через который нужно передать пакет.
- *Доставка по маршруту* - если адрес назначения пакета находится в другой подсети, то коммутатор использует таблицу маршрутов для определения следующего коммутатора, через который нужно передать пакет.
- *Отбрасывание пакета* - если запись в таблице маршрутов не найдена или возникла ошибка при обработке пакета, коммутатор может отбросить пакет или отправить его обратно к отправителю с сообщением об ошибке.

В целом, коммутация пакетов IPv4 осуществляется на основе таблиц маршрутов и алгоритма коммутации. IP модуль выполняет различные действия в зависимости от адреса назначения пакета и информации из таблицы маршрутов, что позволяет передавать пакеты по всей сети Интернет.

20. MTU. Функции фрагментации пакетов IPv4 и Path MTU discovery.

MTU (Maximum Transmission Unit) - это максимальный размер пакета, который может быть передан через сеть без фрагментации. Размер MTU определяется техническими характеристиками сетевых устройств и протоколов, которые используются в сети.

Фрагментация пакетов IPv4 - это процесс разделения большого пакета на несколько меньших фрагментов для передачи через сеть с меньшим MTU. Фрагментация позволяет передавать большие пакеты по сети, но она также может вызывать проблемы, например, задержки в доставке пакетов и потерю данных.

Path MTU discovery (PMTUD) - это процесс автоматического определения максимального размера пакета, который может быть передан без фрагментации через сеть от источника к назначению. PMTUD используется для избежания фрагментации IP-пакетов во время их прохождения через различные сети с разными MTU (Maximum Transmission Unit).

В ходе PMTUD исходный узел начинает отправку IP-пакетов с определенным значением DF (Don't Fragment) - это битовый флаг в заголовке IP-пакета, которое указывает на запрет фрагментации пакета. Если пакет не может пройти через какую-то сеть на своем пути из-за ограничения MTU, сетевое устройство на этом участке отправляет обратно сообщение ICMP с ошибкой "Fragmentation Required and Don't Fragment was Set", содержащее MTU этой сети. Исходный узел затем изменяет размер пакета до максимально допустимого значения и повторяет процедуру до тех пор, пока пакет не достигнет своего назначения.

Функции фрагментации пакетов IPv4 включают следующие шаги:

При получении пакета коммутатор проверяет его размер и сравнивает его с MTU порта, через который пакет должен быть отправлен.

- Если размер пакета больше, чем MTU порта, то коммутатор разделяет пакет на несколько фрагментов по размеру MTU порта.
- Каждый фрагмент получает уникальный идентификатор, который используется для объединения фрагментов обратно в оригинальный пакет.
- Фрагменты отправляются через сеть к адресу назначения.
- Получатель собирает все фрагменты обратно в единый пакет на основе их идентификаторов.

Таким образом, фрагментация пакетов IPv4 и Path MTU discovery являются важными функциями для эффективной передачи данных в сети. Фрагментация позволяет передавать большие пакеты через сеть, но она также может вызывать проблемы. Path MTU discovery помогает избежать фрагментации и определяет оптимальный размер MTU в сети.

21. Обмен служебными сообщениями в процессе функционирования протокола IPv4. Протокол ICMP: виды сообщений, формат сообщений. При каких обстоятельствах применяются.

Протокол IPv4 (Internet Protocol version 4) используется для обмена пакетами данных в сетях TCP/IP. В процессе функционирования этого протокола могут передаваться различные служебные сообщения с помощью протокола ICMP

(Internet Control Message Protocol).

Протокол ICMP предназначен для отправки сообщений об ошибках и событиях, происходящих в процессе передачи данных. Он может использоваться для проверки связи между устройствами в сети и определения наилучшего маршрута для передачи данных.

Виды сообщений, которые могут быть отправлены через протокол ICMP, включают в себя:

- *Echo Request (Ping)* - запрос эхо-сообщения для проверки доступности удаленного узла. Это один из наиболее часто используемых типов ICMP-сообщений.
- *Echo Reply* - ответ на запрос эхо-сообщения. Если удаленный узел доступен и работает корректно, то он должен отправить этот тип ICMP-сообщения в ответ на запрос эхо-сообщения.
- *Destination Unreachable* - сообщение об ошибке, которое отправляется обратно отправителю, если удаленный узел недоступен или если маршрут для доставки пакета не может быть найден.
- *Time Exceeded* - сообщение об ошибке, которое отправляется обратно отправителю, если пакет не может быть доставлен в срок. Это может произойти, например, если пакет зациклился в сети или если его маршрут был изменен.
- *Redirect* - сообщение, которое указывает на более эффективный маршрут для доставки пакета.

Формат сообщений ICMP состоит из заголовка и данных. Заголовок содержит информацию о типе и коде сообщения, а также контрольную сумму для проверки целостности сообщения. Данные могут содержать дополнительную информацию, связанную с сообщением.

Применение протокола ICMP зависит от конкретных обстоятельств, но обычно он используется для диагностики сетевых проблем, проверки доступности удаленных узлов и определения наилучшего маршрута для передачи данных. Например, команда ping использует протокол ICMP для отправки запросов эхо-сообщений и получения ответов от удаленного узла. Также протокол ICMP может быть использован для приема и обработки сообщений об ошибках, когда возникают проблемы в процессе передачи данных.

Протокол ICMP (Internet Control Message Protocol) используется в сетевых технологиях для обеспечения коммуникации между различными устройствами, а также для диагностики и устранения неполадок в сетях.

Основное применение протокола ICMP:

- *Проверка доступности хостов*: Одним из наиболее распространенных способов проверки доступности удаленного хоста является использование команды ping, которая отправляет эхо-запросы с помощью протокола ICMP. Когда удаленный хост получает эти запросы, он должен отправить в ответ эхо-ответы.
- *Обнаружение маршрутов:* Протокол ICMP может быть использован для обнаружения маршрутов в сети. Если происходит изменение маршрута, то протокол ICMP может отправить сообщение об ошибке "Redirect" с информацией о новом маршруте.
- *Определение MTU*: Протокол ICMP также может быть использован для определения максимального размера передаваемых данных (MTU). Если пакет имеет размер большего, чем MTU, то он может быть разделен на несколько фрагментов. В этом случае протокол ICMP будет отправлять сообщения "Fragmentation needed", указывающие на необходимость фрагментации пакета.
- *Обнаружение ошибок:* Протокол ICMP может использоваться для обнаружения ошибок в процессе передачи данных. Если возникают проблемы при отправке или получении пакетов, то устройства могут отправлять сообщения "Destination Unreachable" или "Time Exceeded", чтобы оповестить отправителя о возникших проблемах.
- *Расширенные функции:* Существуют и другие типы сообщений ICMP, которые могут быть использованы для выполнения расширенных функций, таких как определение IP-адреса ближайшего маршрутизатора (traceroute).
- *Протокол ICMP* играет важную роль в обеспечении стабильности и надежности сетевых соединений. Благодаря своей гибкости и широкому спектру функций, он может быть использован в различных ситуациях для диагностики и устранения проблем в сетях.

22. Средства элементарной диагностики IP сетей: ping и traceroute. Описать принцип действия и приблизительную методологию применения.

Средства элементарной диагностики IP сетей - это инструменты, которые помогают определить работоспособность сетевого устройства и проблемы в сетевой связи. Два наиболее распространенных инструмента для этой цели - это ping и traceroute.

Ping - это утилита, которая используется для проверки доступности узла в сети и измерения времени отклика устройства на отправленный ему запрос. Принцип работы Ping заключается в отправке ICMP-пакета (Internet Control Message Protocol) на указанный IP-адрес и ожидании ответа. Если устройство получает ICMP-запрос, то оно отправляет в ответ ICMP-ответ, содержащий время прохождения пакета до устройства и обратно. При этом Ping может отправлять несколько пакетов за раз и выводить статистику по их получению. В результате, если устройство отвечает на ICMP-запросы, значит, оно доступно, а время отклика позволяет определить скорость передачи данных между устройствами.

Методология применения Ping очень проста. Его можно использовать следующим образом:

- Открыть командную строку (в Windows) или терминал (в MacOS/Linux).
- Ввести команду ping и IP-адрес устройства, которое требуется проверить (например, ping 192.168.0.1).
- Дождаться завершения выполнения команды Ping и прочитать результаты.

- Изучить количество отправленных и полученных пакетов, а также время отклика устройства.

Traceroute - это сетевая утилита, которая используется для определения маршрута, по которому проходят пакеты данных в сети Интернет. Она работает по принципу последовательной отправки ICMP-запросов (Internet Control Message Protocol) на удаленный узел с постепенным увеличением значения TTL (Time To Live), начиная с единицы.

Каждый маршрутизатор на пути следования пакета уменьшает значение TTL на единицу. Если значение достигает нуля, то маршрутизатор отбрасывает пакет и отправляет обратно сообщение "Time exceeded" (время жизни истекло) отправителю. Traceroute фиксирует время прохождения каждого маршрутизатора и выводит информацию о задержке (ping time) и IP-адресе каждого узла на маршруте.

Таким образом, Traceroute позволяет узнать количество и порядок маршрутизаторов, через которые проходит пакет до достижения конечного узла, а также определить проблемные сегменты сети, где возможны задержки или потери пакетов. Это делает эту утилиту полезной при диагностике проблем с сетью и выборе оптимального маршрута для передачи данных.

Методология применения Traceroute включает следующие шаги:

- Открыть командную строку (в Windows) или терминал (в MacOS/Linux).
- Ввести команду traceroute и IP-адрес устройства, до которого требуется определить маршрут (например, traceroute 192.168.0.1).
- Дождаться завершения выполнения команды Traceroute и прочитать результаты.
- Изучить список устройств (маршрутизаторов), через которые проходит пакет, и время, затраченное на его прохождение через каждое из устройств.

В целом, Ping и Traceroute являются важными инструментами для диагностики IP-сетей. При правильном использовании они помогают определить доступность устройства, проверить скорость передачи данных и определить маршрут передачи данных в сети.

TTL (Time To Live) - это поле в заголовке IP-пакета, которое указывает на количество маршрутизаторов или сетевых устройств, через которые может пройти пакет, прежде чем он будет отброшен. Каждый раз, когда пакет проходит через маршрутизатор, значение TTL уменьшается на единицу. Если значение TTL достигает нуля, то маршрутизатор должен отбросить пакет и отправить обратно ICMP-сообщение об ошибке.

Таким образом, значение TTL задает максимальное количество сетевых устройств, которые пакет может пройти, и предотвращает зацикливание пакетов в сети. Обычно значение TTL устанавливается в начале маршрутизации и увеличивается по мере прохождения пакета через сеть. Значение TTL может быть изменено при пересылке пакета через другой маршрутизатор.

Значение TTL может быть различным для разных типов пакетов и настраивается в зависимости от конкретных потребностей сети. Например, если TTL установлен очень высоким значением, то это может привести к затратам ресурсов сети, так как пакет может проходить через большое количество маршрутизаторов. Если же TTL установлен слишком низким, то пакет могут быть отброшен раньше времени и не достигнуть своего назначения.

В целом, значение TTL является важным параметром, который влияет на прохождение пакетов через сеть и может быть использован для диагностики проблем в сетевой связи.

23. Особенности функционирования IP на многоточечных сетях (звеньях). MAC адреса: структура, назначение, особые адреса. Роль MAC адресов при передаче кадров по многоточечному звену в различных режимах: unicast, broadcast, multicast.

IP (Internet Protocol) является протоколом сетевого уровня, который обеспечивает маршрутизацию пакетов данных через интернет и другие сети. Однако для передачи пакетов данных по физической сети используются другие протоколы, включая Ethernet.

Ethernet - это стандартный протокол локальных сетей (LAN), который использует метод доступа CSMA/CD (Carrier Sense Multiple Access with Collision Detection). В Ethernet каждое устройство имеет свой уникальный адрес MAC (Media Access Control), который используется для идентификации устройства на физическом уровне сети.

Метод доступа CSMA/CD (Carrier Sense Multiple Access/Collision Detection) - это один из методов, используемых в сетях Ethernet для контроля доступа к среде передачи данных. Он обеспечивает доступ к среде передачи для нескольких устройств, позволяя им конкурировать за передачу данных.

Принцип работы *метода CSMA/CD* заключается в том, что перед началом передачи каждое устройство производит прослушивание среды передачи на наличие других передач. Если среда свободна, то устройство начинает передачу своих данных. Если же в момент передачи данные другого устройства уже находятся на среде передачи, возникает коллизия (collision).

Каждое устройство, обнаружив коллизию, немедленно прекращает передачу и отправляет специальный сигнал (jam signal), оповещая другие устройства о возникновении коллизии. После этого все устройства должны подождать случайный интервал времени перед следующей попыткой передачи. Это помогает избежать повторной коллизии, так как вероятность возникновения коллизии на очередной попытке передачи будет меньше.

Таким образом, метод доступа CSMA/CD обеспечивает эффективное использование среды передачи данных в Ethernet-сетях, позволяя нескольким устройствам конкурировать за доступ к ней и предотвращая возникновение коллизий при одновременной передаче.

MAC-адрес состоит из 48 бит (6 байт) и записывается в формате шестнадцатеричного числа, разделенного двоеточиями. Первые три байта (24 бита) представляют производителя оборудования, а оставшиеся три байта (24 бита) уникальны для каждого устройства производителя.

Назначение MAC-адреса заключается в том, чтобы обеспечить уникальную идентификацию устройства на физическом уровне, что позволяет определять отправителя и получателя кадров данных в сети Ethernet. Каждый раз, когда устройство отправляет кадр данных, оно указывает MAC-адрес получателя в заголовке кадра.

При передаче кадров данных по многоточечной сети Ethernet (звену) используются различные режимы: unicast, broadcast и multicast.

Unicast - это режим передачи данных, при котором кадр отправляется только одному устройству с определенным MAC-адресом получателя. В этом случае отправитель указывает MAC-адрес получателя в заголовке кадра.

Broadcast - это режим передачи данных, при котором кадр отправляется всем устройствам в сети Ethernet. Для этого используется специальный широковещательный MAC-адрес (FF:FF:FF:FF:FF:FF), который указывается в заголовке кадра.

Multicast - это режим передачи данных, при котором кадр отправляется группе устройств, имеющих определенный MAC-адрес. Для этого используется специальный MAC-адрес мультикастной группы, который указывается в заголовке кадра. Устройства, которые принадлежат к этой группе, получают копию кадра данных.

Когда компьютеры общаются в локальной сети, они используют протокол Ethernet. Каждое устройство в этой сети имеет уникальный идентификатор - MAC-адрес. Когда один компьютер хочет отправить данные другому компьютеру, он указывает MAC-адрес получателя в специальном поле заголовка пакета данных.

Но порой возникает необходимость отправлять данные нескольким устройствам одновременно. Например, когда нужно транслировать видео на всех компьютерах в классной комнате, или когда компьютеры должны получить обновление программного обеспечения из централизованного источника. В этом случае используется механизм мультикастинга.

Мультикастинг - это режим передачи данных, при котором один и тот же пакет данных отправляется одновременно нескольким устройствам. Мультикастинговые пакеты имеют специальный IP-адрес, называемый мультикастным адресом. Этот адрес определяет группу устройств, которая должна получить эти данные.

Каждая группа устройств, подписавшаяся на мультикастный адрес, имеет свой уникальный MAC-адрес, который называется MAC-адресом мультикастной группы. Он начинается с префикса "01:00:5E", за которым следует 23 бита номера группы. При передаче мультикастингового пакета компьютер указывает этот MAC-адрес в качестве получателя. Таким образом, все устройства, подключенные к этой группе и имеющие соответствующий MAC-адрес, получают эти данные одновременно.

В стандарте Ethernet и связанных сетевых технологиях существует несколько особых MAC-адресов. Вот несколько примеров:

- *MAC-адрес широковещательной группы* (broadcast): FF:FF:FF:FF:FF:FF. Он используется для передачи данных всем устройствам в сети.
- *MAC-адрес мультикастной группы:* начинается с префикса "01:00:5E", за которым следуют еще 23 бита, представляющие номер группы. Он используется для передачи данных нескольким устройствам одновременно, подключенным к определенной группе.
- *MAC-адрес Loopback-интерфейса*: 00:00:00:00:00:00. Он используется для локального тестирования сетевых приложений и оборудования.
- *MAC-адрес настроенного интерфейса:* 02:XX:XX:YY:YY:YY. Он используется для определения настроенных *MAC*-адресов виртуальных машин или других вспомогательных механизмов.

Виртуальные машины обычно имеют виртуальные сетевые адаптеры, которые могут быть сконфигурированы для использования определенных MAC-адресов. Для этого используются различные методы в зависимости от конкретной системы виртуализации.

Например, в системе виртуализации VMware можно задать MAC-адрес виртуального адаптера вручную или автоматически генерировать его. В Hyper-V от Microsoft также предусмотрена возможность задания статического MAC-адреса для виртуальных адаптеров.

Кроме того, вспомогательные механизмы, такие как контейнеры, также могут иметь свои собственные сетевые интерфейсы с уникальными MAC-адресами, которые могут быть настроены при создании контейнера.

В целом, настройка MAC-адресов виртуальных машин и других вспомогательных механизмов будет зависеть от выбранной системы виртуализации и специфических требований к инфраструктуре.

- *MAC-адрес собственного интерфейса*: это MAC-адрес сетевого интерфейса, присвоенный производителем оборудования. Он используется для уникальной идентификации сетевого оборудования в локальной сети.

Таким образом, MAC-адреса играют важную роль при передаче кадров данных по многоточечной сети Ethernet. Они обеспечивают уникальную идентификацию устройств на физическом уровне и позволяют определять отправителей и получателей кадров данных в различных режимах передачи.

24. Протокол ARP: назначение, место в протокольном стеке, формат пакетов. ARP таблица. Взаимодействие с IP модулем. Выполнение процедуры поиска MAC адреса (ARP-resolve). Алгоритм обработки ARP-пакета.

Протокол ARP (Address Resolution Protocol) используется для разрешения (определения) MAC-адреса устройства по его IP-адресу в локальной сети. Это необходимо для передачи данных между устройствами на физическом уровне.

ARP работает на сетевом уровне модели OSI и является частью протокольного стека TCP/IP. Он позволяет обеспечить соответствие между логическими адресами (IP-адресами) и физическими адресами (MAC-адресами) в локальной сети.

Формат пакетов ARP состоит из следующих полей:

- *Тип аппаратного адреса* (Hardware Type) - определяет тип сетевого интерфейса: Ethernet, Token Ring, FDDI, и т. д.
- *Тип протокола* (Protocol Type) - указывает тип сетевого протокола, для которого выполняется разрешение адреса.
- *Длина аппаратного адреса* (Hardware Address Length) - размер аппаратного адреса в байтах.
- *Длина протокольного адреса* (Protocol Address Length) - размер протокольного адреса в байтах.

- *Код операции* (Operation Code) - определяет тип ARP-сообщения: запрос или ответ.
- *Аппаратный адрес отправителя* (Sender Hardware Address) - MAC-адрес отправителя.
- *Протокольный адрес отправителя* (Sender Protocol Address) - IP-адрес отправителя.
- *Аппаратный адрес получателя* (Target Hardware Address) - MAC-адрес получателя. В ARP-запросах это поле не заполняется, а в ARP-ответах содержит MAC-адрес устройства, для которого запрашивался IP-адрес.
- *Протокольный адрес получателя* (Target Protocol Address) - IP-адрес устройства, для которого выполняется разрешение MAC-адреса.

ARP таблица - это таблица, в которой хранятся соответствия между IP-адресами и MAC-адресами устройств, доступных в локальной сети. ARP таблица обычно заполняется автоматически при обмене данными между устройствами, но может быть также заполнена вручную.

Взаимодействие с IP-модулем происходит следующим образом: при передаче пакета на уровне IP, IP-модуль проверяет, находится ли получатель в локальной сети. Если да, он выполняет процедуру ARP-resolve для определения MAC-адреса получателя и добавляет его в заголовок пакета.

Алгоритм обработки ARP-пакета выглядит следующим образом:

- Получение ARP-пакета.
- Проверка типа аппаратного адреса (обычно Ethernet).
- Проверка длины адресов.
- Проверка типа протокола (обычно IPv4 или IPv6).
- Проверка длины протокольных адресов.
- Обновление ARP таблицы, если необходимо.
- Отправка ответного ARP-пакета, если получен запрос на свой IP-адрес.

Процедура поиска MAC адреса (ARP-resolve) выполняется в следующем порядке:

- Проверка ARP таблицы на наличие записи для нужного IP-адреса.
- Если запись найдена, отправляется запрос на этот MAC-адрес.
- Если запись не найдена, отправляется широковещательный ARP-запрос (broadcast). В запросе содержится IP-адрес запрашиваемого устройства.
- Устройства, которые имеют соответствующий IP-адрес, отвечают собственным MAC-адресом.
- Полученный MAC-адрес добавляется в ARP таблицу.

Таким образом, протокол ARP выполняет важную функцию в локальной сети, обеспечивая определение MAC-адреса по IP-адресу и обновление ARP таблицы. Взаимодействие с IP модулем позволяет выполнять автоматический поиск MAC-адресов устройств в локальной сети.

25. Функциональное описание канального уровня ЭМВОС - перечислить и описать функции канального уровня, описать взаимодействие с другими уровнями ЭМВОС, привести классификацию модулей канального уровня.

Канальный уровень в модели ЭМВОС (OSI) - это второй уровень, который отвечает за передачу данных между соседними узлами сети. Основная задача канального уровня состоит в обеспечении надежной передачи данных, управлении потоком и обнаружении ошибок.

Функции канального уровня:

- Контроль доступа к среде передачи данных
- Формирование кадров данных
- Управление потоком передачи данных

- Обнаружение и исправление ошибок в переданных данных
- Канальный уровень взаимодействует с другими уровнями

ЭМВОС в следующих аспектах:

- Связывается с физическим уровнем для получения доступа к физической среде передачи данных.
- Передает данные на сетевой уровень, формируя кадры данных и добавляя информацию для контроля ошибок.
- Получает от сетевого уровня данные для передачи и управления потоком.

Модули канального уровня могут быть классифицированы по различным параметрам, таким как:

- Режим работы: полудуплексный или дуплексный.

Полудуплексный режим подразумевает, что передача данных возможна только в одном направлении за один период времени. Например, если два устройства обмениваются данными через кабель, каждое из них должно ждать своей очереди на передачу.

В дуплексном режиме связи данные могут передаваться в обоих направлениях одновременно, что повышает эффективность обмена информацией между устройствами.

- Тип среды передачи данных: проводное или беспроводное.

Проводная среда передачи данных использует кабели и соединители для передачи информации между устройствами.

Беспроводная среда передачи данных использует радиоволны или инфракрасные лучи для передачи данных без использования проводов.

- Корпус: внешний или встроенный модуль.

Внешний корпус означает, что устройство имеет отдельный физический корпус, который можно подключить к другому устройству.

Встроенный модуль подразумевает, что устройство является частью другого устройства и не имеет отдельного корпуса. Например, сетевая карта может быть встроенной частью материнской платы компьютера или внешним устройством, подключаемым через USB порт.

Ряд протоколов помогает обеспечить надежную передачу данных и управление потоком на канальном уровне, включая:

- *HDLC* - это протокол управления канального уровня, который обеспечивает надежную передачу данных между двумя узлами в сети. Он используется для управления потоком данных, управления ошибками и управления доступом к среде передачи данных.
- *Протокол контроля передачи* (Transmission Control Protocol, TCP) - гарантирует доставку пакетов данных между двумя узлами в сети.
- *Протокол управления передачей* (Flow Control Protocol) - управляет скоростью передачи данных в сети.
- *Протокол обнаружения ошибок* (Error Detection Protocol) - определяет ошибки при передаче данных по сети.
- *Протокол коррекции ошибок* (Error Correction Protocol) - исправляет ошибки при передаче данных по сети.
- *Протокол управления доступом к среде передачи* (Media Access Control Protocol, MAC) - управляет доступом пользователей к среде передачи данных.
- *Протокол проверки целостности данных* (Cyclic Redundancy Check, CRC) - используется для проверки целостности передаваемых данных.
- *Протокол управления потоком* (Flow Control Protocol) - управляет объемом данных, которые отправляются через сеть.
- *Протокол управления ошибками* (Error Control Protocol) - управляет обработкой ошибок при передаче данных по сети.
- *Протокол автоматической пересылки* (Automatic Repeat Request, ARQ) - устраняет ошибки, возникающие при передаче данных по сети.
- *Протокол проверки паритета* (Parity Check Protocol) - используется для обнаружения ошибок при передаче данных по сети.
- *Протокол контроля скорости передачи* (Rate Control Protocol) - управляет скоростью передачи данных в сети.
- *Протокол управления кадрами* (Frame Control Protocol) - управляет формированием и передачей кадров по сети.

- *Протокол управления квитированием* (Acknowledgment Protocol) – отправляет подтверждение получения пакета данных.
- *Протокол управления выборкой* (Selective Repeat Protocol) – повторно передает только те пакеты данных, которые были потеряны или повреждены при их передаче.
- *Протокол управления окнами* (Windowing Protocol) – используется для управления скоростью передачи данных, основываясь на количестве доступных ресурсов в сети.
- *Протокол управления приоритетами* (Priority Control Protocol) - управляет приоритетами передаваемых данных, чтобы гарантировать более высокий уровень обслуживания для наиболее важных данных.
- *Протокол управления потерями* (Loss Control Protocol) – предотвращает потерю пакетов при передаче данных по сети.
- Протокол управления ресурсами канала связи (Channel Resource Control Protocol) - используется для оптимизации использования доступных ресурсов в сети.
- *Протокол управления буфером* (Buffer Control Protocol) – управляет работой буфера, который временно хранит передаваемые данные, чтобы обеспечить надежную передачу данных.
- *Протокол управления задержкой* (Delay Control Protocol) – контролирует задержки при передаче данных по сети.

26. Кадрирование - описание проблемы, методы ее решения. Кодопрозрачность и способы ее достижения при различных методах кадрирования.

В контексте компьютерных сетей, *кадрирование* может использоваться для изменения размера передаваемых данных. Например, когда данные передаются через сеть, часто необходимо уменьшить их размер, чтобы ускорить процесс передачи и сэкономить пропускную способность сети. Однако, при этом могут возникнуть некоторые проблемы.

Одной из главных проблем кадрирования в контексте компьютерных сетей является потеря информации. Если мы обрезаем данные, то теряем часть информации, которая могла бы быть важна для корректной обработки или анализа этих данных. Кроме того, при изменении размера данных могут возникать ошибки искажения или повреждения данных.

Для решения проблемы потери информации при кадрировании в компьютерных сетях можно использовать различные методы. Один из таких методов - *сжатие данных*. Сжатие позволяет уменьшить размер данных без его обрезания, сохраняя все данные в исходном виде. Для этого используются различные алгоритмы сжатия данных, например, GZIP или LZ77.

Кроме того, можно использовать технику *фрагментации данных*. Фрагментация позволяет разбить данные на более мелкие части, которые затем передаются по сети. Это может помочь уменьшить нагрузку на сеть и ускорить процесс передачи данных. Однако, при использовании этого метода необходимо обеспечить целостность данных, чтобы они были корректно восстановлены после передачи.

Кодопрозрачность также является важным аспектом кадрирования в контексте компьютерных сетей. Кодопрозрачность означает, что изменения, сделанные при кадрировании, не должны повлиять на код или данные, связанные с изображением. Для достижения кодопрозрачности можно использовать сжатие данных с сохранением формата, либо использовать протоколы передачи данных, которые поддерживают целостность данных, например, TCP.

В целом, кадрирование - это важный процесс при работе с данными в компьютерных сетях, который может помочь ускорить процесс передачи данных и сэкономить пропускную способность сети. Однако, при его использовании необходимо учитывать потерю информации и кодопрозрачность, чтобы избежать ошибок и искажений данных, связанных с изображением.

27. Контроль правильности передачи данных - принцип контроля, кодовое пространство, метод обнаружения ошибок, способ построения механизма контроля. Эффективность контролирующего механизма – избыточность, обнаруживающая способность.

Контроль правильности передачи данных - это процесс проверки целостности и достоверности передаваемых данных. Цель контроля правильности передачи данных состоит в том, чтобы обнаруживать и исправлять ошибки в данных, которые могут возникнуть в процессе передачи по сети или хранения на устройствах.

Один из основных принципов контроля правильности передачи данных - *использование контрольных сумм*.

Контрольная сумма - это число, которое вычисляется на основе содержимого передаваемых данных. При получении данных, контрольная сумма вычисляется повторно и сравнивается с исходной контрольной суммой. Если они не совпадают, значит, данные были повреждены, и необходимо запросить их повторную передачу.

Для вычисления контрольной суммы используют *кодовые пространства*, которые представляют множество всех возможных кодов, которые могут быть использованы для передачи или хранения информации. Каждый код представляет собой последовательность символов (обычно битов), которая соответствует определенному значению или символу. *Кодовое пространство для вычисления контрольной суммы* - это множество всех возможных последовательностей символов (обычно битов), которые могут быть использованы для создания контрольной суммы. Используемое кодовое пространство зависит от способа вычисления контрольной суммы и требуемой степени защиты данных. Например, одним из наиболее распространенных кодовых пространств является CRC (Cyclic Redundancy Check), который используется для обнаружения ошибок в бинарных данных.

Существует несколько *методов обнаружения ошибок* с использованием контрольных сумм. Один из таких методов - *одиночная проверка*. При этом методе контрольная сумма вычисляется только один раз, после чего данные передаются по сети. Если при получении данных контрольная сумма не совпадает с исходной, то происходит повторная передача данных.

Другой метод - *многократная проверка*. При этом методе контрольная сумма вычисляется несколько раз в течение передачи данных. Это позволяет обнаруживать ошибки в более коротких интервалах времени и уменьшить количество повторных передач.

Механизм контроля правильности передачи данных может быть построен на основе различных протоколов передачи данных, таких как UDP (User Datagram Protocol) или TCP (Transmission Control Protocol).

Протокол UDP не гарантирует доставку и целостность передаваемых данных, что позволяет уменьшить нагрузку на сеть. Однако, при использовании UDP могут возникнуть ошибки в передаче данных, например, данные могут быть повреждены или потеряны. Для обнаружения таких ошибок в протоколе UDP используется контрольная сумма.

Протокол TCP гарантирует доставку и целостность передаваемых данных, следит за порядком их доставки, а также обеспечивает повторную передачу потерянных пакетов данных. Контроль правильности передачи данных в протоколе TCP реализуется с помощью ACK-пакетов, которые отправляются для подтверждения получения пакетов данных и определения недостающих пакетов. Также в протоколе TCP используется проверка целостности и аутентичности данных с помощью специального поля в заголовке пакета - контрольной суммы.

ACK-пакет (сокр. от acknowledgment packet) - это пакет данных, который используется для подтверждения получения другого пакета данных в сетевой связи между устройствами. Обычно ACK-пакет отправляется после получения пакета данных от другого устройства, чтобы сообщить об успешной передаче или получении данных. Он содержит информацию о номере подтверждаемого пакета и может также содержать дополнительные данные, такие как временные метки или контрольные суммы для обеспечения целостности данных. ACK-пакеты широко используются в протоколах TCP/IP (Transmission Control Protocol/Internet Protocol), которые обеспечивают надежную доставку данных в Интернете.

Аутентичность данных (data authenticity) - это свойство информации, которое гарантирует, что данные были созданы или изменены только допустимым и авторизованным источником, а не поддельным или недобросовестным. Другими словами, аутентичность данных означает, что данные являются подлинными и соответствуют тому, что заявлено.

Другие способы построения механизма контроля правильности передачи данных могут включать в себя использование дополнительных битов для обнаружения ошибок или применение кодовых слов, которые позволяют исправлять ошибки в передаваемых данных.

Дополнительные биты для обнаружения ошибок могут быть добавлены к передаваемым данным с использованием различных алгоритмов. Один из таких алгоритмов - это циклический избыточный код (CRC), который является широко используемым механизмом контроля правильности передачи данных.

Алгоритм CRC работает следующим образом:

- Исходные данные рассматриваются как последовательность битов.
- Формируется блочный код, состоящий из исходных данных и дополнительных контрольных битов. Для создания контрольных битов используется операция деления многочлена.
- Полученный блочный код передается по каналу связи.
- Получатель вычисляет контрольные биты на основе принятого блочного кода, используя тот же самый многочлен. Если полученная контрольная сумма совпадает с переданной контрольной суммой, то данные считаются доставленными без ошибок.

Многочлен, который используется для генерации контрольных битов, называется *генераторным многочленом*.

При использовании CRC для каждого блока данных создается уникальная проверочная сумма, которая содержит информацию о последовательностях битов и другой дополнительной информации. Получатель данных вычисляет проверочную сумму, основываясь на принятых данных, и сравнивает ее со значением, полученным вместе с переданными данными. Если значения не совпадают, то это указывает на возможное изменение данных в процессе передачи, и получатель может запросить повторную передачу.

Кодовые слова, которые позволяют исправлять ошибки в передаваемых данных, являются другим способом обеспечения надежности при передаче данных. Например, код Хемминга использует специальные кодовые слова, которые позволяют исправлять одну или несколько ошибок в передаваемых данных. Это достигается путем добавления дополнительных битов информации (кодовых слов) к передаваемым данным.

Код Хемминга - это метод обнаружения и исправления ошибок в передаваемых данных. В коде Хемминга используются кодовые слова, которые состоят из информационных битов и проверочных битов, добавленных для обнаружения и исправления ошибок.

Каждый проверочный бит определяется как побитовая сумма некоторых информационных битов. Количество информационных битов в каждом блоке данных зависит от количества проверочных битов: чем больше проверочных битов, тем меньше информационных битов.

При передаче данных кодируются с помощью кодовых слов. Если произошла ошибка в передаче, то приемник вычисляет проверочные биты и сравнивает их со значениями, полученными вместе с информационными битами. Если значение проверочного бита не соответствует вычисленному значению, то произошла ошибка и приемник может использовать проверочную информацию для определения места ошибки.

Если возможно исправить одну ошибку, то приемник может изменить единственный бит, который был ошибочным. Если ошибок было несколько, то приемник может определить биты, которые содержат ошибки, но не может точно установить, какие же значения этих битов должны быть.

По мере приема данных получатель вычисляет контрольные суммы и/или проверяет кодовые слова, чтобы обнаружить ошибки. Если были обнаружены ошибки, то получатель может попросить отправителя повторно передать данные или исправить ошибки входных данных, используя специальные алгоритмы.

Эффективность контролирующего механизма зависит от его избыточности и обнаруживающей способности.

Избыточность контролирующего механизма означает, что он должен быть способен обнаруживать как можно больше ошибок, чтобы минимизировать количество повторных передач.

Обнаруживающая способность контролирующего механизма определяет его способность обнаруживать различные типы ошибок. Чем выше избыточность и обнаруживающая способность, тем более эффективным будет механизм контроля правильности передачи данных.

28. Контроль по паритету, матричный код, код Хэмминга. Исправляющая способность. Недостатки паритетных кодов.

Контроль по паритету – это метод обнаружения ошибок в передаче данных, основанный на проверке четности битового потока. Для этого добавляется дополнительный бит, называемый битом четности или битом проверки

четности (Parity Check Bit), который определяет, должно ли количество единиц в передаваемом сообщении быть четным или нечетным. Например, для контроля по четности с четным числом единиц добавляется "0", а для контроля по четности с нечетным числом единиц добавляется "1".

Матричный код – это метод кодирования информации, основанный на использовании матриц. В матричных кодах каждое сообщение представлено в виде вектора из битовых символов, а для его кодирования используется матрица.

Существует несколько различных типов матричных кодов, но одним из наиболее распространенных является *блочный матричный код*. В блочном матричном коде информационное сообщение разбивается на блоки фиксированной длины, которые затем кодируются отдельно.

Для создания *матрицы кодирования* используются *генерирующие матрицы*, которые позволяют преобразовать информационные данные в *кодовые слова*. Кодовые слова обычно имеют большую длину, чем исходное сообщение, что позволяет исправлять ошибки при передаче данных.

При *декодировании матричного кода*, полученное кодовое слово умножается *на проверочную матрицу*, которая помогает определить наличие ошибок и их местонахождение. Если была обнаружена ошибка, то декодер может использовать синдром, который вычисляется на основе проверочной матрицы, чтобы определить, какой бит был изменен при передаче данных, и исправить его.

Матричные коды используются в различных областях, включая телекоммуникации, компьютерные сети, цифровое телевидение и др. Они позволяют эффективно использовать доступную пропускную способность канала связи и обеспечивают надежную передачу данных.

Код Хэмминга – это тип блочного кодирования, который используется для исправления одиночных ошибок в передаваемых данных. Он был разработан Ричардом Хэммингом в 1950-х годах.

Код Хэмминга расширяет информационный блок данных, добавляя дополнительные биты контроля ошибок. Каждый бит контроля ошибок используется для проверки определенной комбинации битов данных. Это позволяет обнаруживать и исправлять одиночные ошибки в передаваемых данных. Например, если в сообщении произошла ошибка в одном из битов данных, проверочные биты могут использоваться для определения, какой бит был изменен, и исправить его.

В коде Хэмминга используется матрица проверки четности, которая определяет, какие биты данных должны быть проверены на наличие ошибок. Если при передаче данных происходит ошибка, то полученный код будет несоответствующим одному из возможных кодов, которые можно сформировать с помощью матрицы проверки четности. Таким образом, принимающая сторона может определить, что произошла ошибка, и исправить ее.

Общая формула для построения матрицы проверки четности имеет вид:

$H = [PT \mid I],$

где P - матрица перестановок (порядок битов данных), T - транспонированная матрица паритетных чеков (какие комбинации битов проверять на наличие ошибок), I - единичная матрица размерности k x k (k - количество бит данных). Каждый столбец транспонированной матрицы соответствует определенному биту данных, а каждая строка - определенному биту контроля ошибок.

Исправляющая способность – это количество ошибок, которое может быть обнаружено и исправлено с помощью данного кода. Например, код Хэмминга сможет исправить одиночные ошибки в блоках данных, но не сможет корректировать множественные ошибки.

Недостатки паритетных кодов – это то, что они не могут обнаруживать ошибки, если число ошибок в передаваемом сообщении больше одной. Также паритетные коды не могут исправлять ошибки, только обнаруживать их. Кроме того, добавление дополнительного бита четности увеличивает объем передаваемых данных на 1 бит на каждые 7 бит информации, что неэффективно при передаче больших объемов данных.

29. Циклические избыточные проверки (CRC) - описание, принцип действия, показатели эффективности.

Циклические избыточные проверки (CRC) – это метод обнаружения ошибок в передаваемых данных, который используется во многих протоколах связи и хранения данных.

При использовании CRC данные представляются в виде битового потока, к которому добавляется некоторое количество контрольных битов, вычисленных с помощью алгоритма циклического кодирования. Контрольные биты являются функцией данных и рассчитываются на основе полиномиальной формулы, которая выбирается заранее.

При приеме данных полученный битовый поток проверяется на соответствие этой же полиномиальной формуле. Если результат не соответствует ожидаемому значению, то произошла ошибка в передаче данных.

Особенностью метода CRC является его способность обнаруживать не только одиночные ошибки, но и групповые ошибки, при условии что их количество не превышает размер контрольной суммы (т.е. число контрольных бит). Кроме того, метод CRC отличается достаточно высокой скоростью обработки данных и простотой реализации.

Недостатком метода CRC является то, что он не позволяет исправлять ошибки, а только обнаруживать их. Кроме того, возможно появление ложных срабатываний в случае использования не подходящей полиномиальной формулы или недостаточного количества контрольных бит.

Принцип действия CRC заключается в добавлении к передаваемому сообщению контрольной суммы, которая рассчитывается на основании содержимого сообщения. Получатель сообщения также вычисляет контрольную сумму и сравнивает ее со значением, полученным от отправителя. Если значения не совпадают, то это означает, что произошла ошибка при передаче данных.

Контрольная сумма вычисляется путем деления передаваемого сообщения на заданный полином и взятия остатка от этого деления. Этот остаток является контрольной суммой, которая добавляется к сообщению и передается вместе с ним. Получатель сообщения выполняет тот же самый расчет контрольной суммы и сравнивает ее со значением, полученным от отправителя.

Показатели эффективности CRC зависят от выбора полинома, используемого для расчета контрольной суммы. Некоторые полиномы имеют более высокую степень защиты от ошибок, чем другие. Важно выбирать полиномы с высокой степенью защиты от ошибок, чтобы обеспечить надежность передачи данных.

Также следует учитывать, что использование CRC приводит к увеличению размера передаваемых данных из-за добавления контрольной суммы. Однако эта дополнительная нагрузка на систему обычно оправдана за счет повышенной надежности передачи данных.

30. Восстановление ошибочно переданных данных: FEC и ARQ. Описание протокола простого ARQ (stop and wait): подтверждение, восстановление от ошибок, таймаут, предотвращение дублей. Эффективность простого ARQ.

При передаче данных по каналу связи могут возникать ошибки, из-за которых данные могут быть искажены или не доходить до получателя вовсе. Для исправления таких ошибок используют два метода: прямое кодирование с помощью передачи дополнительной информации (Forward Error Correction, FEC) и обратная связь с автоматическим повтором запросов (Automatic Repeat reQuest, ARQ).

FEC - это метод, при котором данные дополняются специальным кодом, который позволяет восстановить ошибки без необходимости отправки дополнительных запросов на повтор передачи. Однако данный метод требует заранее известного количества ошибок, которые могут возникнуть в процессе передачи данных.

ARQ - это метод, при котором получатель отправляет подтверждение (ACK) о правильном приеме данных и запрос на повтор передачи (NAK) в случае обнаружения ошибок. Существуют различные реализации протокола ARQ, одной из которых является простой протокол "stop and wait".

Протокол "stop and wait" работает следующим образом:

- Отправитель отправляет блок данных.
- Получатель принимает блок данных и проверяет его на наличие ошибок. Если блок содержит ошибки, то получатель отправляет запрос на повтор передачи.
- Отправитель повторно передает те же данные.
- Получатель повторно принимает данные и отправляет подтверждение о правильном приеме.
- Отправитель переходит к отправке следующего блока данных.

Протокол ARQ "stop and wait" - это простой протокол обратной связи с автоматическим повтором запросов, который используется для восстановления ошибочно переданных данных. Основные механизмы:

- *Подтверждение (ACK)* - получатель отправляет подтверждение о правильном приеме данных. Данное сообщение позволяет отправителю знать, что данные были успешно доставлены.
- *Восстановление от ошибок* - если получатель обнаруживает ошибки в переданных данных, он отправляет запрос на повтор передачи (NAK). Отправитель получает этот запрос и повторно отправляет данные.
- *Таймаут* - отправитель устанавливает таймер, который запускается после отправки блока данных. Если за определенное время не будет получено подтверждение о правильном приеме, то отправитель повторно отправляет данные.
- *Предотвращение дублей* - каждый блок данных содержит уникальный номер, который позволяет получателю проверить, были ли уже получены данные с таким же номером. Если данные уже были получены, то получатель игнорирует их.

В целом, протокол ARQ "stop and wait" является достаточно простым, но эффективным способом обеспечения надежности передачи данных. Однако он имеет свои ограничения и может быть неэффективным при высокой вероятности возникновения ошибок или при больших задержках в передаче данных. Для более эффективной обработки ошибок существуют более сложные протоколы ARQ, такие как "go-back-n" и "selective-repeat".

Эффективность простого ARQ зависит от нескольких факторов:

- *Вероятность возникновения ошибок* - чем меньше вероятность возникновения ошибок в передаваемых данных, тем более эффективен будет протокол ARQ "stop and wait". Если же вероятность ошибок высока, то скорость передачи данных может быть существенно замедлена из-за повторных запросов на повторную передачу.
- *Время ожидания подтверждений* - время ожидания подтверждений может быть значительно большим в условиях низкой пропускной способности канала связи. Это также может привести к увеличению времени передачи данных и снижению эффективности протокола ARQ.
- *Размер блока данных* - если размер блока данных слишком велик, то повторная передача целого блока может занять слишком много времени. В этом случае протокол ARQ "stop and wait" может быть неэффективным и более сложные протоколы ARQ должны использоваться вместо него.
- *Задержки в сети* - задержки в сети могут привести к тому, что получатель не успевает отправить подтверждение вовремя. В таком случае отправитель повторно передает данные, что может занять слишком много времени и увеличить время передачи данных.

В целом, простой протокол ARQ "stop and wait" может быть эффективным в условиях низкой вероятности ошибок и невысоких задержек в передаче данных. Однако для более сложных условий, таких как высокая вероятность ошибок или большие задержки в сети, могут потребоваться более сложные протоколы ARQ с механизмами обработки ошибок, адаптивной скоростью передачи данных и многими другими функциями.

31. Описание протокола ARQ со скользящим окном (Go-Back-N): понятие окна, процедура нормальной передачи, примитив NAK, правила продвижения окна, процедура повторной передачи.

Протокол ARQ (Automatic Repeat Request) со скользящим окном – это протокол обеспечения надежной доставки пакетов между двумя узлами сети. Он использует механизм повторной передачи для обеспечения корректной доставки пакетов. Протокол ARQ со скользящим окном имеет две основные процедуры: нормальную передачу и повторную передачу.

Окно - это диапазон последовательных номеров пакетов, которые может отправить отправитель, не дожидаясь подтверждения от получателя. Размер окна определяется параметрами протокола и обычно является константой.

Процедура нормальной передачи:

- Отправитель посылает пакеты от N до N+W-1, где N - номер первого пакета в окне, а W - размер окна.
- Получатель получает пакеты и отправляет подтверждение (ACK) с номером следующего ожидаемого пакета.
- Отправитель продвигает окно на количество успешно доставленных пакетов, т.е. если получатель подтвердил доставку пакета с номером N+1, то отправитель продвигает окно на один пакет и начинает передачу пакета с номером N+W.

Примитив NAK используется, когда получатель обнаруживает ошибку в принятом пакете. Получатель посылает NAK с номером наибольшего непринятого пакета. Это заставляет отправителя повторно передать все пакеты, начиная с номера, указанного в примитиве NAK.

Правила продвижения окна:

- Если получатель отправляет ACK, равный номеру ожидаемого пакета, то отправитель продвигает окно на один пакет.
- Если получатель отправляет ACK, меньший номера ожидаемого пакета, то это может быть дубликат подтверждения и отправитель проигнорирует его.
- Если получатель отправляет NAK, то отправитель сбросит окно до номера, указанного в NAK.

Процедура повторной передачи:

- Если отправитель не получает подтверждение доставки пакета в течение определенного времени, он считает, что произошел сбой и запускает процедуру повторной передачи. Он повторно передает все пакеты, находящиеся в текущем окне, начиная с первого пакета. Получатель должен корректно обработать повторно переданные пакеты и отправить подтверждения, как при нормальной передаче.
- Если отправитель не получает подтверждения после определенного количества повторных передач, он считает, что соединение недоступно и прерывает передачу.

Таким образом, протокол ARQ со скользящим окном гарантирует доставку пакетов в правильном порядке и без потерь при условии, что размер окна выбран правильно и задержки в сети не слишком большие.

32. Эффективность Go-Back-N ARQ. Выбор оптимального размера окна. Метод повышения эффективности путем выборочного повтора.

Go-Back-N ARQ (Automatic Repeat Request) - это протокол управления ошибками, который используется для обеспечения надежной доставки данных в сетях передачи пакетов. Он работает по принципу отправки нескольких пакетов подряд без ожидания подтверждения от приемника, а затем ожидания подтверждения и повторной отправки всех пакетов, начиная с последнего успешно доставленного.

Эффективность Go-Back-N ARQ зависит от многих факторов, таких как задержка в сети, размер окна, скорость передачи данных и вероятность возникновения ошибок. Оптимальный размер окна выбирается на основе этих факторов, чтобы минимизировать потери пакетов и избежать перегрузки сети. Существует несколько *методов для выбора оптимального размера окна*, один из которых - это использование формулы:

*W = sqrt(2 * RTT * C / L)*

где W - размер окна, RTT - время задержки в сети, C - скорость передачи данных, L - длина пакета данных.

Другой метод повышения эффективности Go-Back-N ARQ - это *выборочный повтор*. В этом случае, если приемник обнаруживает ошибку в каком-то пакете, он не отправляет подтверждение для всех последующих пакетов до тех пор, пока отправитель не повторит только те пакеты, которые были потеряны или повреждены. Это сокращает количество лишних повторных передач и уменьшает задержку в сети.

Однако, следует отметить, что использование выборочного повтора может привести к увеличению задержки для приемника, так как ему нужно ждать повторной передачи определенных пакетов. Поэтому, выбор между использованием этого метода или стандартного Go-Back-N ARQ должен зависеть от конкретных условий сети и требований к надежности и скорости передачи данных.

33. Функции управления звеном передачи данных: управление потоком (Flow control) и управление установкой соединения (Dial-UP).

Управление потоком (Flow control) в терминологии передачи данных означает процесс регулирования скорости передачи данных между устройствами, чтобы избежать переполнения буфера приемника. Это особенно важно в случаях, когда скорость передачи данных на приемнике меньше, чем на передатчике. Для управления потоком используются различные методы, включая *программное управление потоком и аппаратное управление потоком.*

Программное управление потоком предполагает отправку специальных сигнальных символов (контрольных символов), которые информируют устройство-приемник о доступности буфера для получения новых данных.

Аппаратное управление потоком, с другой стороны, предусматривает использование дополнительной аппаратуры, например, линий *RTS/CTS*, которые сигнализируют об уровне занятости буфера приемника.

Управление установкой соединения (Dial-up) - это процесс подключения удаленного компьютера к локальной сети или Интернету через модем. В Windows, например, установка соединения Dial-up осуществляется через встроенный в ОС наборщик номера телефона, который доступен в меню "Пуск" -> "Настройка" -> "Сеть и Интернет" -> "Центр управления сетями и общим доступом" -> "Настройка нового подключения".

После запуска этого инструмента пользователь должен выбрать тип подключения (в данном случае - Dial-up), указать номер телефона провайдера, имя пользователя и пароль, а также другие необходимые параметры, такие как настройки протокола TCP/IP и DNS.

После завершения настройки подключения пользователь может запустить его, нажав на соответствующую кнопку. Если все настройки указаны правильно, то ОС должна установить соединение с провайдером и пользователь сможет

начать использовать Интернет.

Кроме того, многие модемы для Dial-up соединений имеют свое собственное программное обеспечение, которое обычно можно установить на компьютер. Это ПО может предоставлять дополнительные функции, такие как отображение текущей скорости передачи данных, статистику использования подключения и т.д.

После установки соединения между устройствами начинается передача данных через соединение Dial-up. При этом также используются методы управления потоком, чтобы избежать переполнения буфера приемника. По завершении передачи данных соединение разрывается, и устройство-отправитель снова становится доступным для нового соединения.

Управление потоком (Flow control) - это процесс контроля скорости передачи данных между устройствами. Один из способов программного управления потоком - это использование особого символа Xoff для временной приостановки передачи данных, когда буфер приемника заполнен до определенного уровня. При *получении символа Xoff* устройство-отправитель прекращает отправку данных и ожидает появления *символа Xon*, сигнализирующего о доступности буфера приемника.

Другой метод программного *управления потоком - это использование протокола RTS/CTS* (Request to Send/Clear to Send). *Устройство-приемник отправляет сигнал RTS* устройству-отправителю, когда его буфер достигает определенного уровня.

Устройство-отправитель в ответ отправляет сигнал CTS, если он готов начать передачу данных, или же игнорирует запрос, если еще не может начать передачу.

Аппаратное управление потоком реализуется с помощью линий RTS/CTS и DTR/DSR (Data Terminal Ready/Data Set Ready), которые дополнительно контролируют состояние связи между устройствами.

DTR и DSR - это аббревиатуры, используемые в информационных технологиях для обозначения двух линий связи между устройствами.

- *DTR (Data Terminal Ready)* - это сигнальная линия, которую использует оборудование для указания терминальному оборудованию о том, что оно готово к передаче данных.
- *DSR (Data Set Ready)* - это сигнальная линия, которую использует модем или другое устройство для указания подключенному компьютеру о том, что оно готово к передаче данных.

Таким образом, DTR и DSR используются для обмена сигналами между двумя устройствами, чтобы они могли определить, готовы ли они к передаче и приему данных.

Алгоритм управления установкой соединения (Dial-up) представляет собой процесс подключения удаленного компьютера к локальной сети или Интернету через модем. При установке соединения используются следующие шаги:

- Устройство-отправитель набирает номер телефона удаленного устройства и устанавливает соединение через телефонную линию.
- Когда удаленное устройство отвечает на вызов, устройство-отправитель передает ему информацию о типе своего модема, скорости передачи данных и других параметрах.
- Удаленное устройство проверяет правильность полученных параметров, и если все настроено правильно, соединение устанавливается.
- После установки соединения начинается передача данных между устройствами, при которой также используются методы управления потоком.
- По завершении передачи данных соединение разрывается, и устройство-отправитель становится доступным для нового соединения.

Таким образом, функции управления звеном передачи данных играют важную роль в обеспечении эффективной передачи данных между устройствами, в том числе и при использовании соединения Dial-up.

34. Управление доступом к среде передачи данных: описание сути проблемы и условий, при которых она проявляется. Понятие коллизии. Режим управления звеном Master/Slave и его свойства.

Управление доступом к среде передачи данных - это процесс регулирования доступа различных устройств к общей среде передачи данных. Эта проблема возникает при работе в сетевой среде, где несколько устройств пытаются передавать данные одновременно. Если доступ к среде не будет регулироваться, то возможны коллизии - случаи, когда два или более устройства начинают передачу одновременно и данные перепутываются.

Коллизия - это конфликтный случай, который возникает, когда два или более устройства пытаются передавать данные в одном и том же временном интервале. В результате, данные от этих устройств могут столкнуться на линии связи и образовать ошибки в передаче.

Для решения проблемы управления доступом к среде передачи данных был разработан режим управления звеном *Master/Slave*. В этом режиме одно из устройств (Master) контролирует доступ к среде передачи данных и определяет, какое устройство должно передавать данные в данный момент. Остальные устройства (Slave) получают команды от Master и передают данные только по указанию Master.

Свойства режима управления звеном Master/Slave:

- Организация сети по принципу мастер-раб.
- Равномерное использование ресурсов сети.
- Уменьшение количества коллизий при передаче данных.
- Возможность контроля доступа к среде передачи данных.
- Снижение времени задержки на передачу данных.

В целом, режим управления звеном Master/Slave позволяет оптимизировать процесс передачи данных в сети, повысить его эффективность и уменьшить количество ошибок в передаче.

Конкретным примером может быть использование режима Master/Slave в сети Ethernet. В этом случае, коммутатор работает в режиме Master, а все подключенные к нему устройства - в режиме Slave. Коммутатор контролирует передачу данных между устройствами и управляет потоком данных, чтобы избежать коллизий и обеспечить максимальную производительность сети.

В режиме Master/Slave, мастер имеет полный контроль над рабочими устройствами, которые выполняют только то, что им поручает мастер. Например, в сети Ethernet, мастер коммутатор может определять порты, на которые должны быть направлены данные, контролировать поток данных и устанавливать приоритеты передачи данных.

Этот режим также может использоваться в других типах сетей, таких как CAN (Controller Area Network), где управляющее устройство (мастер) управляет передачей данных между другими устройствами в сети (рабочие).

35. Управление доступом к среде в режиме маркерной шины. Процедура нормального обмена данными и процедура реконфигурации логического кольца. Преимущества и недостатки маркерной шины.

Управление доступом к среде в режиме маркерной шины (Token Bus) - это еще один способ координировать доступ к общей среде передачи данных. В этом режиме каждый узел сети имеет свой собственный адрес, а данные передаются в виде пакетов, которые перемещаются по логическому кольцу от одного узла к другому.

Процедура нормального обмена данными в режиме маркерной шины выглядит следующим образом:

- Маркер передается по кольцу от узла к узлу. Каждый узел может отправлять пакеты только тогда, когда маркер находится у него.
- Узел, который хочет передать данные, добавляет пакет в кольцо после маркера и передает его дальше по кольцу.
- Пакет движется по кольцу до тех пор, пока не достигнет адресата.
- Адресат получает пакет, подтверждает его получение и удаляет из кольца.

Маркер маркерной шины - это специальное устройство, представленное в виде байта, которое перемещается по кольцевой топологии сети Token Ring. Маркер может быть реализован как *аппаратно, так и программно*.

- *Аппаратный маркер* обычно реализуется в виде устройства, генерирующего электрический импульс-маркер, который передается по кольцу от одного устройства к другому. Он состоит из набора логических элементов, которые обеспечивают его перемещение по кольцу и контроль доступа к кольцу.
- *Программный маркер* может быть реализован на программном уровне, используя определенное программное обеспечение для контроля доступа к кольцевой топологии сети. В этом случае, маркер представляет собой логическую единицу, которая переносится между устройствами в виде блока данных.

Независимо от того, как реализован маркер маркерной шины, его функция заключается в определении порядка передачи данных в сети Token Ring и контролировании доступа к кольцевой топологии.

Процедура реконфигурации логического кольца в режиме маркерной шины используется в случае, если узел выходит из сети или новый узел присоединяется к сети. Эта процедура состоит из следующих шагов:

- Узел, который должен выйти из сети, отправляет специальный пакет на удаление своего адреса из кольца.
- Когда узел, который должен выйти из сети, получает маркер, он отправляет сообщение о том, что он готов удалить свой адрес из кольца.
- Когда все узлы подтверждают, что они готовы к удалению адреса узла, маркер передается другому узлу в кольце.
- Новый узел захватывает маркер и добавляет свой адрес в кольцо.

Преимущества режима маркерной шины:

- Организация сети по принципу логического кольца.
- Предотвращение коллизий при передаче данных.
- Высокая степень отказоустойчивости - если один узел выходит из сети, общение продолжается между оставшимися узлами.
- Возможность работы с большим количеством узлов.

Недостатки режима маркерной шины:

- Низкая скорость передачи данных по сравнению с другими методами доступа.
- Необходимость тщательной настройки параметров сети.
- Возможность возникновения задержек в передаче данных из-за необходимости ожидания маркера.

- Трудность обнаружения и устранения ошибок в случае их возникновения.

В режиме маркерной шины для поиска ошибок используется специальный инструмент - *анализатор маркерной шины*. Он подключается к маркерной шине и позволяет отслеживать передаваемые данные, а также проверять наличие ошибок в этих данных.

Анализатор маркерной шины может отслеживать различные типы ошибок, например, ошибки четности, ошибки битов при передаче данных и другие ошибки, связанные с нарушением целостности информации.

Кроме того, анализатор маркерной шины может использоваться для сбора статистики о процессе передачи данных на маркерной шине, что позволяет выявлять возможные проблемы в работе системы и проводить ее диагностику.

В методе передачи данных по маркерной шине возможны различные проблемы, которые могут затруднять поиск ошибок с помощью анализатора маркерной шины.

Некоторые из этих проблем могут включать:

- Сложность настройки анализатора маркерной шины: чтобы корректно работать, анализатор маркерной шины должен быть правильно настроен и сконфигурирован для работы с конкретной маркерной шиной и используемыми на ней устройствами. Это может потребовать дополнительного времени и усилий со стороны инженеров.

- Возможность задержек при передаче данных: если данные на маркерной шине передаются слишком медленно или с задержкой, это может затруднить обнаружение ошибок, особенно если они происходят не на всех передаваемых данных.

- Несовместимость между устройствами: разные устройства могут использовать разные протоколы и форматы данных для передачи информации по маркерной шине. Это может привести к тому, что анализатор маркерной шины не может корректно распознать данные, передаваемые между ними, что затруднит поиск ошибок.

- Необходимость дополнительного оборудования: для работы с маркерной шиной может потребоваться специализированное оборудование, такое как анализатор маркерной шины или логический анализатор. При его отсутствии или неумении им пользоваться, поиск ошибок может быть затруднен.

В целом, поиск ошибок в методе передачи данных по маркерной шине может быть сложным и требует хорошего знания протоколов и форматов передаваемых данных. Однако, при правильной настройке и использовании специальных инструментов, можно достичь высокой точности в поиске ошибок.

36. Управление доступом к среде в режиме соперничества. Усовершенствованный метод доступа Ethernet МДПН/ОК (CSMA/CD): детектор несущей, детектор коллизий, регламент доступа станции к среде. Механизм возникновения коллизии в CSMA/CD и факторы, влияющие на её вероятность.

Управление доступом к среде в режиме соперничества используется для управления доступом к общей среде передачи данных, которая может быть использована несколькими станциями одновременно. Одним из методов управления доступом является Ethernet МДПН/ОК (CSMA/CD), который используется в локальных вычислительных сетях.

Ethernet МДПН/ОК (CSMA/CD) расшифровывается как "Множественный доступ с прослушиванием несущей/обнаружением столкновения" (Carrier Sense Multiple Access with Collision Detection). Это технология передачи данных по кабельным линиям, которая используется для создания локальных сетей (LAN). Она позволяет нескольким устройствам подключаться к сети одновременно и обмениваться данными. В процессе передачи данных каждое устройство слушает линию на наличие других передач, и в случае обнаружения конфликта прерывает свою передачу.

CSMA/CD - это протокол управления доступом в сетях Ethernet, который основывается на трех компонентах: детекторе несущей, детекторе коллизий и регламенте доступа станции к среде.

- *Детектор несущей* отслеживает наличие сигнала на среде передачи данных. Если сигнал обнаружен, то станция ожидает окончания передачи данных другой станции и только после этого начинает передачу своих данных.

Детектор несущей - это устройство в сетевых технологиях, которое служит для определения наличия несущей частоты на кабельной линии. Он используется для контроля за передачей данных и гарантирует, что только одно устройство находится в активном режиме передачи в каждый момент времени. Если детектор несущей обнаруживает, что линия занята, то устройство откладывает свою передачу до тех пор, пока линия не будет свободна.

Несущая частота - это высокочастотный сигнал, который используется для передачи информации в радиосвязи или кабельных сетях. Он представляет собой сигнал, который не содержит непосредственно передаваемую информацию, а служит только для передачи этой информации на носительной волне.

Носительная волна - это высокочастотный сигнал, который используется для передачи информации в радиосвязи и кабельных сетях. Он представляет собой электромагнитную волну, которая не несет информацию напрямую, но служит для переноса модулирующего сигнала.

Для передачи данных через кабельную линию, модулирующий сигнал (например, данные) накладывается на несущую частоту, чтобы создать модулированный сигнал, который может быть передан через среду передачи. Приемник на другом конце линии извлекает (демодулирует) оригинальный сигнал из модулированного сигнала, вычитая несущую частоту.

- *Детектор коллизий* следит за тем, что происходит на среде передачи данных, и определяет, если две или более станций пытаются передать данные одновременно. В этом случае происходит коллизия.

- *Регламент доступа* станции к среде определяет, как станция получает доступ к среде передачи данных. Станции получают доступ к среде только тогда, когда она свободна. Если станция не обнаруживает наличия сигнала на среде передачи данных, то она может начать передачу своих данных.

Коллизия в CSMA/CD возникает, когда две или более станции начинают передачу данных одновременно на общей среде передачи данных. Это происходит потому, что станции не могут одновременно обнаружить наличие других передач на среде передачи данных, и поэтому они могут начинать свою передачу данных, что приводит к коллизии.

Механизм возникновения коллизии в CSMA/CD можно объяснить следующим образом:

- Станция A начинает передачу данных.
- Станция B также начинает передачу данных практически одновременно с станцией A, но не успевает обнаружить наличие сигнала на среде передачи данных, который был создан станцией A.
- Оба сигнала доходят до всех станций в сети, и все станции обнаруживают наличие коллизии.
- Станции отбрасывают свои пакеты и ждут случайное время перед повторной передачей.

Факторы, которые влияют на вероятность возникновения коллизии в CSMA/CD, включают в себя:

- *Высокая загрузка среды передачи* данных - если много станций пытаются передавать данные одновременно, то вероятность коллизии возрастает.
- *Большое количество станций, которые пытаются передавать данные одновременно* - чем больше станций, тем выше вероятность коллизии.
- *Большое расстояние между станциями* - если расстояние между станциями большое, то время задержки при передаче данных увеличивается, что приводит к более высокой вероятности коллизии.
- *Задержки в передаче данных, вызванные проблемами сетевого оборудования* - если сетевое оборудование работает неправильно или есть другие проблемы, то это может привести к задержке при передаче данных и увеличить вероятность коллизии.
- *Использование старых или некачественных кабелей для передачи данных* - плохие кабели могут создавать шум на среде передачи данных, что может привести к повышению вероятности коллизии.
- *Низкий уровень шума на среде передачи данных* - если уровень шума на среде передачи данных слишком низкий, то станции могут определить наличие свободной среды передачи данных неправильно, что может привести к коллизии.

Уровень шума в среде передачи данных - это мера того, насколько сильно сигналы, передаваемые по каналу связи, искажаются различными помехами. Он выражается в децибелах и показывает отношение мощности сигнала к мощности шума.

В целом, чтобы снизить вероятность возникновения коллизии в CSMA/CD, необходимо правильно настроить сеть и использовать высококачественное сетевое оборудование.

37. Семейство протоколов HDLC - место в протокольном стеке, назначение. Виды и формат кадров. Режимы обмена данными. Реализация ARQ в HDLC. Методы управления потоком.

HDLC (High-Level Data Link Control) является семейством протоколов передачи данных, используемых в сетях связи. HDLC является стандартом ISO 3309 и является расширением протокола SDLC (Synchronous Data Link Control), который был разработан для использования на каналах связи, работающих по синхронному режиму.

HDLC находится на канальном уровне в протокольном стеке OSI модели. Его основным назначением является обеспечение надежной передачи данных между компьютерами через сеть связи. Протокол HDLC может быть использован для передачи данных между точками одной линии связи (точка-точка) или между группами устройств (многопользовательский режим).

Формат кадра HDLC состоит из заголовка, информационного поля, контрольной последовательности и окончательной последовательности.

- Заголовок содержит информацию о типе кадра, адресе отправителя и получателя.
- Информационное поле содержит данные, которые нужно передать.
- Контрольная последовательность - это последовательность бит, используемая для проверки ошибок в данных.
- Окончательная последовательность используется для обозначения конца кадра.

HDLC поддерживает два режима передачи данных:

- *Асинхронный режим (Asynchronous Balanced Mode - ABM)* - это режим передачи данных, в котором каждая сторона может выступать как инициатор или получатель. Каждый кадр имеет свой уникальный номер, который используется для определения порядка кадров и проверки наличия ошибок.
- *Режим передачи данных с ограниченным доступом (Restricted Unbalanced Mode - RUM)* - это режим передачи данных, в котором одна сторона является инициатором, а другая сторона - получателем. В этом режиме передачи данных используется механизмы управления потоком и контроля ошибок.

ARQ (Automatic Repeat Request) представляет собой метод проверки и повторной передачи (retransmission) данных в случае возникновения ошибок при передаче. HDLC поддерживает ARQ, позволяя получателю отправить запрос на повторную передачу блока данных, который был принят с ошибками. Если отправитель не получает подтверждения о доставке кадра, то он повторно передает этот кадр.

Управление потоком - это процесс контроля объема данных, передаваемых от инициатора к получателю. HDLC поддерживает два метода управления потоком:

- *Метод управления потоком "Stop and Wait"* - это метод, в котором передатчик отправляет один кадр и останавливается на некоторое время, ожидая подтверждения о получении данных от получателя.

- *Метод управления потоком "Sliding Window"* - это метод, в котором передатчик может отправить несколько кадров до того, как он должен остановиться и ожидать подтверждения от получателя. Получатель отправляет подтверждения о получении кадров, и передатчик продолжает отправлять следующий блок данных, пока не достигнет максимального размера окна (Window Size).

HDLC определяет три типа кадров: информационный кадр (I-кадр), подтверждающий кадр (ACK-кадр) и отрицательный подтверждающий кадр (NACK-кадр).

- *Информационный кадр (I-кадр)* содержит данные, которые передаются от отправителя к получателю. Каждый I-кадр имеет свой уникальный номер, который используется для проверки наличия ошибок в данных и распознавания порядка кадров приемником.

- *Подтверждающий кадр (ACK-кадр)* отправляется получателем, чтобы подтвердить успешную передачу одного или нескольких I-кадров. ACK-кадры не содержат данных и имеют номер последнего успешно принятого I-кадра.

- *Отрицательный подтверждающий кадр (NACK-кадр)* отправляется получателем, если один или несколько I-кадров были приняты с ошибками. NACK-кадры сообщают отправителю, какие кадры нужно повторно передать, и имеют номер последнего успешно принятого I-кадра.

Реализация ARQ в HDLC основана на использовании подтверждений и повторной передачи кадров. Когда отправитель передает I-кадр, он ожидает подтверждение от получателя. Если получатель успешно принимает кадр, то он отправляет ACK-кадр с номером последнего успешно принятого I-кадра. Если получатель не может успешно принять I-кадр, то он отправляет NACK-кадр с номером последнего успешно принятого кадра и запросом на повторную передачу неверно принятых данных.

Когда отправитель получает NACK-кадр, он повторно передает только те кадры, которые были указаны в NACK-кадре. Отправитель повторно передает каждый кадр до тех пор, пока не получит подтверждение от получателя или не будет достигнут лимит попыток повторной передачи.

Например, если вы используете протокол TCP/IP для передачи данных, то количество попыток повторной передачи может быть ограничено настройками таймаута, которые определяют, сколько времени должно пройти до повторной попытки отправки данных. Если качество сетевого соединения низкое, то это может привести к большому количеству ошибок при передаче данных, что требует большего количества попыток повторной передачи.

38. Протокол (протокольное семейство) PPP - назначение, разновидности, место в протокольном стеке. Формат кадров PPP и способы инкапсуляции. Внутренняя структура протоколов PPP: управляющие протоколы xCP, протоколы аутентификации xAP и сетевые протоколы. Процедура установки соединения PPP.

Протокол PPP (Point-to-Point Protocol) - это протокол сетевого уровня, который используется для установки и поддержания соединений между двумя сетевыми устройствами через последовательный канал связи. PPP является одним из самых распространенных протоколов для обеспечения логической связи между компьютером и Интернет-сервером.

Протокольное семейство PPP включает несколько разновидностей протоколов, таких как:

- PAP (Password Authentication Protocol) - протокол аутентификации по паролю.

- CHAP (Challenge Handshake Authentication Protocol) - протокол аутентификации с использованием вызова и отклика.

- EAP (Extensible Authentication Protocol) - расширяемый протокол аутентификации, который позволяет использовать различные методы аутентификации, такие как система идентификации по сертификатам или биометрические данные.

- IPCP (Internet Protocol Control Protocol) - протокол управления протоколом интернет-протокола (IP), используемый для настройки адресов IP на соединении.

- LCP (Link Control Protocol) - протокол управления каналом связи, который проверяет работоспособность линии связи и настраивает параметры соединения.
- NCP (Network Control Protocol) - протокол управления сетью, который определяет тип сети и параметры, такие как MTU (максимальный размер передаваемого пакета).

- SLIP (Serial Line Internet Protocol) - простой протокол интернет-соединения, который используется для передачи данных между компьютерами через последовательный порт.

- MPPC (Microsoft Point-to-Point Compression) - протокол сжатия данных, разработанный Microsoft для использования с PPP.

- MPPE (Microsoft Point-to-Point Encryption) - протокол шифрования данных, который также был разработан Microsoft для использования с PPP.

- BACP (Bandwidth Allocation Control Protocol) - протокол управления шириной канала связи, который позволяет динамически изменять параметры соединения на основе текущей загрузки сети.

- CCP (Compression Control Protocol) - протокол управления сжатием данных, который позволяет выбирать наиболее эффективный алгоритм сжатия для передачи данных по соединению.

- ECP (Encryption Control Protocol) - протокол управления шифрованием данных, который позволяет выбирать наиболее эффективный алгоритм шифрования для защиты данных при передаче по соединению.

- CCPMP (Compression Control Protocol for the MPPE) - расширение протокола CCP для управления сжатием данных при использовании протокола MPPE.

- ML-PPP (Multilink Point-to-Point Protocol) - протокол, позволяющий объединять несколько каналов связи в одно логическое соединение PPP для повышения пропускной способности.

- PPPoE (PPP over Ethernet) - протокол, использующий PPP для установления соединения по Ethernet, часто используется для доступа к Интернету через DSL-модемы.

DSL-модем (Digital Subscriber Line modem) — это устройство, предназначенное для передачи данных по линии цифровой абонентской линии (DSL). DSL-модемы используются для подключения к Интернету в домашних и офисных сетях, использующих технологию DSL.

DSL-модемы обеспечивают преобразование сигналов между цифровой и аналоговой формой, что позволяет использовать одну линию для передачи голосовых и данных сигналов одновременно. Обычно DSL-модемы устанавливаются провайдерами интернет-услуг и подключаются к компьютерам или маршрутизаторам через Ethernet-порты.

- PPPoA (PPP over ATM) - протокол, использующий PPP для установления соединения по ATM (Asynchronous Transfer Mode), также используется для доступа к Интернету через DSL-модемы.

Соединение по ATM (Asynchronous Transfer Mode) - это метод передачи данных, который используется для сетей высокой скорости. Он позволяет обеспечивать быстрый и эффективный обмен данными между устройствами в компьютерных сетях.

ATM-соединения представляют из себя логические каналы связи между устройствами, которые обеспечивают передачу данных с гарантированным качеством обслуживания. Это означает, что при использовании ATM-соединений можно гарантировать определенный уровень пропускной способности и задержки, что является очень важным для многих приложений, таких как видеоконференции, онлайн-игры и другие.

ATM-соединения работают по принципу передачи данных в виде небольших "клеток" фиксированного размера, которые передаются между устройствами в сети. Каждая клетка содержит информацию о типе данных, адресе назначения и другие параметры, необходимые для правильной передачи данных. ATM-соединения могут быть настроены для передачи данных различных типов, включая голос, видео и данные.

- RADIUS (Remote Authentication Dial-In User Service) - протокол аутентификации и управления доступом пользователей, который может быть использован с PPP для проверки логина и пароля пользователя.

- L2TP (Layer Two Tunneling Protocol) - протокол, который использует PPP в качестве подслоя управления соединением для создания туннелей между двумя удаленными сетями через Интернет.

Туннель в компьютерных сетях - это механизм, который позволяет передавать данные между устройствами через другую сеть, как будто они находятся в одной локальной сети. Туннели используются для обеспечения безопасности и конфиденциальности данных, а также для связи между удаленными сетевыми устройствами.

Туннель создается путем упаковки данных из одного протокола в другой. Например, IP-трафик может быть упакован в другой протокол, такой как SSL или SSH, и передаваться через интернет для обеспечения безопасной связи между двумя сетевыми устройствами.

Туннели также могут использоваться для переадресации трафика через несколько устройств на пути от отправителя к получателю. Например, VPN-сервер может создавать туннель между клиентским компьютером и целевой сетью, чтобы обеспечить безопасный доступ к ресурсам в этой сети.

Туннели могут быть настроены на различных уровнях сетевой стеки, включая уровень приложения, транспортный уровень и сетевой уровень. Они также могут использоваться для соединения различных типов сетей, включая локальные сети (LAN), глобальные сети (WAN) и Интернет.

- SSCP (Synchronous Serial Control Protocol) - протокол управления подключением на синхронных последовательных линиях связи.

- EAP-TLS (Extensible Authentication Protocol-Transport Layer Security) - расширяемый протокол аутентификации, который использует TLS-шифрование для обеспечения безопасности при передаче данных об аутентификации.

TLS (Transport Layer Security) - это протокол безопасности, который используется для шифрования данных при передаче через компьютерные сети. TLS является преемником протокола SSL и обеспечивает безопасное соединение между клиентским и серверным устройствами.

TLS-шифрование позволяет защитить данные от несанкционированного доступа, перехвата и изменения. При использовании TLS-шифрования данные передаются между клиентом и сервером в зашифрованном виде, что обеспечивает конфиденциальность информации и защиту от подмены данных.

TLS использует симметричное и асимметричное шифрование для обеспечения безопасности данных. Когда клиент устанавливает связь с сервером, он отправляет запрос на установление защищенного канала связи. Затем сервер отправляет сертификат, который содержит открытый ключ, используемый для установления защищенного канала связи. Клиент использует открытый ключ сервера для шифрования сессионного ключа, который затем используется для шифрования данных. Это позволяет обеспечить безопасную передачу данных между клиентом и сервером.

TLS-шифрование используется, например, в HTTPS-соединениях для обеспечения безопасного доступа к сайтам в Интернете. Оно также широко используется в других приложениях, включая электронную почту, чаты и телефонию через Интернет.

- PPP-MP (PPP Multilink Protocol) - протокол, позволяющий объединять несколько каналов связи в одно логическое соединение PPP для повышения пропускной способности.

PPP находится на сетевом уровне OSI-модели и используется для создания логической связи между точками. В протокольном стеке TCP/IP PPP находится на уровне 2, под IP-протоколом, но выше уровня физической передачи данных.

Формат кадров PPP имеет следующую структуру:

- Байт начала кадра (0x7E)

- Поле адреса (обычно 0xFF)

- Поле управления (обычно 0x03)

- Поле протокола (например, 0x0021 для CHAP)

- Поле данных (передаваемые данные)

- CRC-16 (циклический избыточный код) для проверки целостности кадра
- Байт конца кадра (0x7E)

Существует несколько способов инкапсуляции PPP-фреймов, таких как PPPoE и PPPoA.

- *PPPoE (Point-to-Point Protocol over Ethernet)* - это метод инкапсуляции PPP кадров в кадры Ethernet, который позволяет передавать данные между двумя устройствами через Ethernet-сеть. Он является стандартной технологией широкополосного доступа в Интернет и используется преимущественно в домашних сетях, а также в корпоративных сетях, где требуется высокая скорость передачи данных.

Широкополосный доступ - это способ получения скоростного подключения к интернету, который позволяет передавать большое количество данных за короткое время. Более простыми словами, *это быстрый интернет*.

- *PPPoA (Point-to-Point Protocol over ATM)* - это метод инкапсуляции PPP кадров в кадры ATM, который позволяет передавать данные между двумя устройствами через сеть ATM. Он используется преимущественно в выделенных линиях и общедоступных телефонных сетях.
- HDLC-инкапсуляция (High-level Data Link Control) - наиболее распространенный метод инкапсуляции в PPP. Данные упаковываются в кадры и добавляются заголовки и контрольные суммы для обеспечения целостности данных.
- PAP-инкапсуляция (Password Authentication Protocol) - используется для аутентификации пользователей при подключении к удаленной системе. Клиент отправляет имя пользователя и пароль серверу, который проверяет их на правильность.
- CHAP-инкапсуляция (Challenge Handshake Authentication Protocol) - более безопасный метод аутентификации, который использует хэш-функции для защиты пароля пользователя. Клиент отправляет свой идентификатор и случайное число серверу, который использует эти данные для генерации хэш-кода. Клиент затем выполняет проверку полученного хэша.
- EAP-инкапсуляция (Extensible Authentication Protocol) - гибкий метод аутентификации, который позволяет использовать различные типы аутентификации, такие как пароль, сертификаты и токены.
- BACP-инкапсуляция (Bandwidth Allocation Control Protocol) - используется для динамического выделения пропускной способности канала связи.
- CCP-инкапсуляция (Compression Control Protocol) - позволяет сжимать данные, которые передаются через канал связи.
- IPCP-инкапсуляция (Internet Protocol Control Protocol) - используется для установления IP-адресов на каждой стороне соединения.

Внутренняя структура протоколов PPP включает три типа протоколов:

- управляющие протоколы (Control Protocols, xCP), которые отвечают за установление, поддержание и разрыв логического соединения;
- протоколы аутентификации (Authentication Protocols, xAP), которые используются для проверки подлинности пользователей или устройств.
- сетевые протоколы, такие как IP, ICMP и другие, которые передают данные между точками соединения.

Процедура установки соединения PPP включает следующие шаги:

- Установка физического соединения между двумя устройствами.
- Отправка инициализационных кадров связи с использованием LCP (Link Control Protocol). Этот протокол определяет параметры соединения, такие как скорость передачи данных, размер кадра и другие.
- Аутентификация пользователя или устройства (если необходимо) с помощью протоколов аутентификации xAP.
- Установка сетевого соединения с использованием протоколов NCP (Network Control Protocol).
- Обмен данными между устройствами через логическое соединение PPP.

В целом, протокол PPP отлично себя зарекомендовал в качестве стандарта для установки и поддержания соединений между двумя устройствами через последовательный канал связи.

39. Управляющие протоколы PPP: назначение (согласование параметров), виды и структуры управляющих пакетов. Параметры (options) и процедура их согласования.

Протокол PPP (Point-to-Point Protocol) широко используется для установления соединения между двумя сетевыми устройствами, такими как маршрутизаторы или модемы. Один из ключевых аспектов работы протокола PPP - это процесс согласования параметров между устройствами, который осуществляется через передачу управляющих пакетов.

Назначение управляющих пакетов PPP заключается в обмене информацией о параметрах соединения и согласовании их между устройствами. Это позволяет обеспечить правильную работу протокола, а также настроить различные параметры соединения в соответствии с требованиями конкретной сети.

Существует несколько видов управляющих пакетов PPP, которые можно разделить на три категории:

- *Пакеты управления связью (Link Control Protocol, LCP)* - отвечают за установление и поддержание соединения между устройствами.

- *Пакеты управления сетью (Network Control Protocol, NCP)* - отвечают за настройку параметров сети, таких как адресация, протоколирование и другие.
- *Пакеты данных (Data Packets)* - представляют собой данные, которые передаются между устройствами.

Структура управляющих пакетов PPP включает в себя заголовок и тело.

Заголовок содержит информацию о:

- типе пакета (LCP, NCP или Data)
- длине пакета
- Код (Code): 8 бит, определяет тип управляющего пакета. Существует несколько типов кодов, таких как Configure-Request, Configure-Ack, Terminate-Request и т.д.

Код (Code) в структуре управляющих пакетов PPP - это 8-битное поле, которое определяет тип управляющего пакета. Код может принимать различные значения в зависимости от назначения пакета.

Некоторые из наиболее распространенных кодов управляющих пакетов PPP:

- Configure-Request (0x01): используется для запроса конфигурации соединения.
- Configure-Ack (0x02): используется для подтверждения приема и согласия на предложенную конфигурацию соединения.
- Configure-Nak (0x03): используется для указания тех параметров конфигурации, которые не могут быть приняты.
- Configure-Reject (0x04): используется для указания тех параметров конфигурации, которые необходимо отклонить.
- Terminate-Request (0x05): используется для запроса завершения соединения.
- Terminate-Ack (0x06): используется для подтверждения завершения соединения.
- Code-Reject (0x07): используется для отклонения неверной последовательности кодов.

Код является обязательным полем управляющих пакетов PPP и играет важную роль в установлении и контроле соединения между двумя узлами.

- Идентификатор протокола (Protocol Identifier): определяет сетевой протокол, который используется поверх PPP. Например, значение 0x0021 указывает на протокол IP (Internet Protocol).
- Биты флага (Flags): используются для обозначения состояния соединения. Например, бит "S" (Start) устанавливается для обозначения начала соединения, а бит "E" (End) устанавливается для обозначения завершения соединения.
- Последовательный номер (Sequence Number): используется для отслеживания последовательности управляющих пакетов PPP в рамках сеанса связи.
- Подтверждение (Acknowledgement Number): используется для подтверждения приема предыдущих управляющих пакетов PPP.
- Опции управления потоком (Flow Control Options): используются для контроля размера окна передачи и скорости передачи данных между двумя узлами.

Тело пакета содержит собственно данные, которые передаются между устройствами. В случае управляющих пакетов, тело может содержать информацию, связанную с установлением и контролем соединения между двумя узлами. Например, в управляющем пакете Config-Request тело может содержать параметры конфигурации, запрашиваемые отправителем у получателя.

Протокол PPP предоставляет механизм для передачи различных типов данных между устройствами, включая управляющие пакеты и данные пользовательского трафика. Данные пользовательского трафика передаются в виде пакетов данных без каких-либо дополнительных полей заголовка и тела.

PPP определяет набор *параметров (options)*, которые могут быть настроены для управления процессом установки и поддержания соединения между двумя узлами. Некоторые из этих параметров могут быть обязательными, другие - необязательными.

Примеры параметров, которые можно настроить в PPP:

- Authentication Protocol (протокол аутентификации). Этот параметр определяет метод аутентификации, который будет использоваться при установке соединения. PPP поддерживает различные методы аутентификации, такие как PAP (Password Authentication Protocol) и CHAP (Challenge Handshake Authentication Protocol).
- Network Control Protocol (протокол управления сетью). Этот параметр определяет протокол, который будет использоваться для управления сетью, например, для определения IP-адресов и других параметров сети. Существует ряд стандартных протоколов управления сетью, таких как IPCP (IP Control Protocol) и IPv6CP (IPv6 Control Protocol).

- Compression Protocol (протокол сжатия). PPP может использовать специальные протоколы сжатия, чтобы уменьшить объем передаваемых данных. Некоторые из этих протоколов включают PRC (PPP Compression Protocol) и MPPC (Microsoft Point-to-Point Compression).

Процедура согласования параметров в PPP состоит из нескольких этапов:

- Установление линии связи. Прежде чем начать процедуру согласования параметров, необходимо установить физическую линию связи между двумя узлами PPP. Для этого используются специальные протоколы, такие как LCP (Link Control Protocol), которые обеспечивают проверку качества соединения и другие функции.

- Отправка запросов на параметры. После установления линии связи один узел PPP начинает отправлять запросы на установку конкретных параметров другому узлу PPP. Запросы могут содержать информацию о различных параметрах, таких как протокол аутентификации или протокол управления сетью.

- Обработка запросов и ответов. Получив запрос на установку параметров, другой узел PPP обрабатывает его и отправляет обратный ответ с указанием того, был ли запрошенный параметр успешно установлен. Если параметр не может быть установлен, то в ответе указывается соответствующий код ошибки.

- Согласование параметров. Если все запрошенные параметры были успешно установлены и целостность соединения подтверждена, то процедура согласования параметров завершается успешно. Если же какой-то из параметров не может быть установлен, то процесс может быть повторен или соединение может быть разорвано.

- Поддержание соединения. После успешного согласования параметров узлы PPP начинают обмениваться данными и мониторят качество связи. Если качество связи падает, то может потребоваться повторное согласование параметров или пересогласование на более низком уровне протокола (например, LCP).

Важно отметить, что процедура согласования параметров в PPP является довольно гибкой и может быть настроена для работы с различными типами сетей и устройств. Все параметры PPP определяются в стандарте RFC 1661 и других соответствующих документах, которые содержат подробные описания каждого параметра и его возможных значений.

40. Транспортный уровень ЭМВОС: назначение и выполняемые функции. Режимы транспортного сервиса и протоколы, их реализующие (UDP и TCP).

Транспортный уровень ЭМВОС (Электронной Международной Вычислительной Офисной Сети) - это один из уровней модели OSI (Open Systems Interconnection), который отвечает за передачу данных между приложениями на разных узлах сети. Назначение транспортного уровня заключается в обеспечении надежности и целостности передаваемых данных, а также контроле потока информации.
Основными функциями транспортного уровня являются:
- Разбитие данных на пакеты: для передачи больших объемов информации по сети, данные разбиваются на меньшие блоки (пакеты), которые передаются по отдельности.

- Обеспечение надежности передачи: транспортный уровень гарантирует, что данные будут доставлены в нужном порядке и без потерь.

- Управление потоком данных: чтобы сеть не перегружалась, транспортный уровень контролирует скорость передачи данных и учитывает возможные задержки.

- Определение адресата: транспортный уровень определяет, к какому приложению должны быть доставлены данные.

Для реализации этих функций транспортный уровень использует различные протоколы. Наиболее распространенными являются протоколы *UDP (User Datagram Protocol) и TCP (Transmission Control Protocol).*

Протокол UDP является простым и быстрым способом передачи данных без гарантий доставки или контроля потока. Он часто используется для передачи видео, аудио, игровых приложений и других приложений, которые не требуют высокой надежности.

Протокол TCP обеспечивает более высокую надежность передачи данных, так как он контролирует поток данных и подтверждает получение каждого пакета. Также он гарантирует, что данные будут доставлены в нужном порядке. Протокол TCP используется для большинства приложений, которые требуют высокой надежности, например, для сайтов, электронной почты и файловых передач.
Режимы транспортного сервиса определяют тип соединения между узлами сети:

- *Соединение на основе установления: этот режим используется в протоколе TCP*. Передача данных начинается только после установления соединения между узлами сети, и данные передаются в определенном порядке.

- *Режим без установления соединения: этот режим используется в протоколе UDP*. Передача данных начинается сразу после отправки пакета, и данные могут быть доставлены в любом порядке.

41. Идентификация SAP внутри одного узла на примере TCP/UDP. Номер порта. Явное и автоматическое назначение номера порта на SAP.

SAP (System Application and Products), как и многие другие приложения, использует номера портов TCP/UDP для идентификации процессов на узле.

Номер порта - это 16-битный целочисленный идентификатор, который указывается в заголовке каждого сетевого пакета на уровне транспорта.

Для SAP, номер порта по умолчанию для TCP - 32 00 (hex) или 12800 (decimal) и для UDP - 33 00 (hex) или 13000 (decimal). Однако SAP также предоставляет возможность явного назначения номера порта для каждого процесса, который должен использовать протокол SAP.

Явное назначение номера порта для SAP может быть выполнено следующим образом:

- Определите номер порта, который будет использоваться для процесса SAP
- В файле конфигурации SAP (saprfc.ini) добавьте параметр "gwhost" и установите его значение в IP-адрес вашей системы.
- Добавьте параметр "gw/reg_port" и установите его значение равным выбранному номеру порта.
- Сохраните изменения в файле конфигурации.

Автоматическое назначение номера порта для SAP может быть выполнено с помощью следующих шагов:

- Запустите административную консоль SAP GUI.
- В меню выберите "Сервисы администратора системы" и выберите "Управление сервером".
- Перейдите на вкладку "Настройки сети" и выберите "Автоматический выбор порта".
- Установите параметры для автоматического назначения номера порта, если это необходимо.
- Сохраните изменения и перезапустите процессы SAP.

Таким образом, идентификация SAP внутри одного узла на примере TCP/UDP осуществляется посредством номера порта, который можно явно или автоматически назначить для каждого процесса SAP.

42. Протокол UDP: назначение, формат сегмента. Сценарий использование UDP в программе.

UDP (User Datagram Protocol) - это протокол транспортного уровня, который обеспечивает передачу данных без гарантии доставки и подтверждения получения. UDP используется для быстрой передачи данных в сетях, где небольшие задержки имеют решающее значение.

Формат сегмента UDP состоит из следующих полей:

- Поле порта источника (source port) - 16 бит
- Поле порта назначения (destination port) - 16 бит
- Поле длины (length) - 16 бит
- Поле контрольной суммы (checksum) - 16 бит

Поля порта источника и порта назначения указывают на номера портов отправителя и получателя соответственно.

Поле длины указывает на размер всего сегмента UDP в байтах, включая поля заголовка и данные.

Поле контрольной суммы используется для обнаружения ошибок при передаче данных.

Сценарии использования UDP в программе могут быть различными:

- *Реализация игровых приложений* - UDP используется для быстрой передачи данных о состоянии игры между клиентами и сервером без задержек, которые могут возникнуть при использовании TCP.
- *Реализация приложений для потоковой передачи данных* - UDP используется для передачи потокового видео и аудио, где быстрая передача данных имеет приоритет перед надежностью доставки.
- *Реализация систем управления сетевыми устройствами* - UDP используется для отправки команд на удаленные устройства без гарантии доставки и подтверждения получения.
- *Реализации протоколов VoIP (Voice over IP)* - UDP используется для передачи голосовых данных в режиме реального времени, где задержки могут повлиять на качество связи.

В общем, UDP используется там, где быстродействие более важно, чем надежность передачи данных. Также его широко используют для реализации многих сетевых протоколов, в том числе DNS, DHCP, SNMP и других.

43. Протокол TCP: назначение, структура передаваемых данных и способ двойной асинхронной буферизации. Сценарий использования TCP в программе. Механизм клонирования socket-а в сервере.

Протокол TCP (Transmission Control Protocol) - это протокол транспортного уровня, который обеспечивает надежную передачу данных между приложениями. Он использует методы управления потоком и контроля ошибок, чтобы гарантировать доставку данных без повторной передачи или потерь.

Структура передаваемых данных в TCP состоит из заголовка и полезной нагрузки:

Заголовок TCP содержит следующие поля:

- Порт источника - 16 бит
- Порт назначения - 16 бит
- Номер последовательности - 32 бита
- Номер подтверждения - 32 бита
- Длина заголовка - 4 бита
- Флаги - 6 бит
- Размер окна - 16 бит
- Контрольная сумма - 16 бит
- Указатель на данные - 16 бит (опционально)

Полезная нагрузка содержит данные, которые нужно передать.

Способ двойной асинхронной буферизации в TCP позволяет оптимизировать процесс передачи данных. Он основан на использовании двух буферов: один буфер для чтения и другой для записи. Когда данные передаются по сети, они сначала помещаются в буфер записи, а затем извлекаются и отправляются по сети. Это позволяет приложению продолжать работу без ожидания окончания передачи данных.

Сценарии использования TCP в программе могут быть различными:

- *Реализация веб-серверов* - TCP используется для передачи данных между клиентами и сервером во время запроса и ответа на HTTP-запросы.
- *Реализация электронной почты* - TCP используется для передачи электронных писем между почтовыми серверами и почтовыми клиентами.
- *Реализация систем управления базами данных* - TCP используется для передачи запросов и ответов между клиентами и серверами баз данных.
- *Реализация приложений для мгновенного обмена сообщениями* - TCP используется для передачи сообщений между пользователями в режиме реального времени.

Механизм клонирования socket-а в сервере позволяет создавать несколько одинаковых соединений на один порт сервера. Это позволяет обрабатывать несколько запросов от одного или нескольких клиентов одновременно. Когда сервер получает запрос на создание нового соединения, он создает новый socket, который является копией существующего socket-а. Это позволяет серверу обрабатывать несколько соединений параллельно без привязки к конкретному клиенту.

44. Протокол TCP: формат сегмента, описание полей. Процедуры установки и закрытия соединения
TCP. Общее понятие об автомате состояний TCP модуля.

Протокол TCP (Transmission Control Protocol) – это надежный протокол транспортного уровня, который обеспечивает передачу данных между приложениями на разных компьютерах в сетях TCP/IP.

Формат сегмента TCP:

- Поле порта отправителя - 16 бит
- Поле порта получателя - 16 бит
- Номер последовательности - 32 бита
- Номер подтверждения - 32 бита
- Длина заголовка - 4 бита
- Зарезервировано - 3 бита
- Флаги - 9 битов
- Размер окна - 16 бит
- Контрольная сумма - 16 бит

- Указатель Urgent - это приоритетный индикатор, который указывает на то, что сообщение или задача имеют высокий уровень важности и требуют немедленного внимания. - 16 бит (опционально)
- Опции - переменной длины, могут отсутствовать

Описание полей:

- Порт отправителя и порт получателя: определяют источник и назначение сообщения.
- Номер последовательности: определяет количество переданных байт в потоке данных.
- Номер подтверждения: указывает на следующий ожидаемый байт в потоке данных.
- Длина заголовка: указывает размер заголовка в 32-битных словах.
- Флаги: используются для управления сеансом связи.
- Размер окна: задает количество данных, которые можно передать без подтверждения от получателя.
- Контрольная сумма: используется для обнаружения ошибок в переданных данных.
- Указатель Urgent: указывает на последний байт, который имеет высокий приоритет.
- Опции: используются для определения различных параметров сеанса связи.

Процедуры установки и закрытия соединения TCP:

Установка соединения:

- Клиент отправляет пакет со значением SYN=1 и номером последовательности X.
- Сервер отправляет пакет со значением SYN=1 и номером последовательности Y, а также ACK=X+1 - подтверждение получения пакета от клиента и запрос на установку соединения.
- Клиент отправляет пакет со значением ACK=Y+1 - подтверждение получения пакета от сервера и завершение процедуры установки соединения.

Закрытие соединения:

- Клиент отправляет пакет со значением FIN=1 - запрос на закрытие соединения.
- Сервер отправляет пакет со значением ACK=1 - подтверждение получения запроса от клиента.
- Сервер отправляет пакет со значением FIN=1 - запрос на закрытие соединения с своей стороны.
- Клиент отправляет пакет со значением ACK=1 - подтверждение получения запроса от сервера и завершение процедуры закрытия соединения.

Автомат состояний TCP модуля:

TCP-соединение может находиться в одном из шести состояний:

- LISTEN - ожидание установления соединения;
- SYN-SENT - передача запроса на установку соединения;
- SYN-RECEIVED - ожидание подтверждения запроса на установку соединения;
- ESTABLISHED - соединение установлено;
- FIN-WAIT-1 - ожидание запроса на закрытие соединения или подтверждения о закрытии соединения;
- CLOSED - соединение закрыто.

Автомат состояний TCP модуля позволяет контролировать состояние соединения и выполнять соответствующие операции в зависимости от текущего состояния. При установке соединения TCP-модуль переходит из состояния LISTEN в состояние SYN-SENT, затем в состояние SYN-RECEIVED и далее - в состояние ESTABLISHED, когда соединение установлено.

При закрытии соединения TCP-модуль может находиться в состоянии FIN-WAIT-1, ожидая запроса на закрытие соединения или подтверждения о закрытии соединения. Затем он переходит в состояние FIN-WAIT-2, когда ожидается подтверждение закрытия соединения от удаленного хоста, и в состояние TIME-WAIT, когда производится задержка для того, чтобы убедиться в доставке всех пакетов. Наконец, TCP-модуль переходит в состояние CLOSED, когда соединение закрыто полностью.

Таким образом, протокол TCP обеспечивает надежную передачу данных между приложениями на разных компьютерах в сетях TCP/IP, а автомат состояний TCP-модуля контролирует состояние соединения и обеспечивает корректное выполнение процедур установки и закрытия соединения.

-- Флаги:

- NS (ECN-nonce) — Устойчивый механизм сигнализации насыщения с помощью ECN-nonce

- CWR (Congestion Window Reduced) («Окно перегрузки уменьшено») - флаг установлен отправителем, чтобы указать, что получен пакет с установленным флагом ECE

- ECE (ECN-Echo) — указывает, что данный узел способен на ECN (явное уведомление перегрузки) и для указания отправителю о перегрузках в сети

- URG (Urgent pointer). Когда узел отправляет сегмент с URG флагом, то узел-получатель принимает его на отдельном канале.

- ACK (Acknowledgement) — поле «Номер подтверждения» задействовано

- PSH (Push function) - инструктирует получателя протолкнуть данные, накопившиеся в приёмном буфере, в приложение пользователя. Когда узел отправляет информацию, TCP сохраняет её в буфере и не передает её сразу другому узлу, ожидая, захочет ли узел-отправитель передать ещё. Такая же схема работает и у узла-получателя. Когда он получает информацию, TCP сохраняет её в буфере, чтобы не тревожить приложение из-за каждого байта полученной информации. Если узел отправляет сегмент с PSH флагом, это значит, что он отправил все, что было нужно.

- RST (Reset the connection) — оборвать соединения, очистить буфер

- SYN (Synchronize sequence numbers) — синхронизация номеров последовательности

- FIN (final бит) — флаг, будучи установлен, указывает на завершение соединения