Сети доставки контента. adsl, fttb

ADSL — это технология высокоскоростного доступа в интернет по обычной медной телефонной паре (телефонной линии). Ключевая особенность — асимметрия: скорость передачи данных **от сети к пользователю** (download) значительно выше, чем скорость от пользователя в сеть (upload).

FTTB (оптика до здания) / FTTH (оптика до квартиры) — это технология доступа в интернет, при которой **оптическое волокно** прокладывается до границы жилого дома (в подвал/чердак, где стоит коммутатор) или непосредственно в квартиру абонента. От этой точки до конечного устройства (компьютера, роутера) сигнал передаётся по обычной медной витой паре (Ethernet) или по тому же волокну. Обеспечивает симметричный канал с высокой скоростью и низкой задержкой.

VPN

VPN — это технология, которая создает защищенное, зашифрованное логическое ("виртуальное") соединение **поверх другой сети** (чаще всего поверх публичной и небезопасной, такой как Интернет). Она эмулирует прямое защищенное подключение между устройствами (или между устройством и корпоративной сетью), как если бы они были соединены напрямую или находились в одной локальной сети. Основные компоненты: туннелирование данных, их шифрование и аутентификация сторон. Работает на сетевом и транспортном уровнях.

- **Что происходит:**
    1. Ваши данные (запрос к сайту, сообщение) упаковываются в **зашифрованный "конверт"**.
    2. Этот "конверт" отправляется на **сервер VPN** (это вход в туннель).
    3. По пути в открытом интернете все видят только, что вы связались с сервером VPN, но **не могут прочитать, что внутри конверта**.
    4. Сервер VPN распаковывает конверт и отправляет ваш запрос дальше к целевому сайту (например, в Google) **от своего имени**.

Зачем это нужно? Три главные причины:

1. **Безопасность в чужих сетях** (в кафе, аэропорту): Ваш трафик шифруется, и хакер в той же сети не сможет перехватить ваши пароли.

2. **Конфиденциальность (анонимность):** Для внешних сайтов и вашего провайдера **запрос исходит от сервера VPN**, а не от вас. Они видят IP-адрес VPN-сервера, а не ваш настоящий. Это может помочь обойти географические блокировки.
3. **Доступ к "закрытым" сетям:** Сотрудник из дома может через VPN подключиться к внутренней сети офиса, как будто его рабочий компьютер прямо в ней находится (получить доступ к файловым серверам, базам данных).

Структура стандартов ieee

Представьте, что **IEEE — это "Министерство сетевого строительства"**, которое пишет **"Свод правил" (стандарты)** для всех, кто делает сетевое оборудование и ПО. Чтобы не создавать одну гигантскую и запутанную книгу правил, Министерство разбило её на **отдельные "главы" или "проекты"**, за каждый из которых отвечает своя команда экспертов.

Стандарты IEEE 802 **не изобретают одну технологию**. Они **описывают, как разные технологии должны себя вести, чтобы оборудование от разных производителей (Cisco, HP, TP-Link) могло работать вместе.** Благодаря этому вы можете купить любой Wi-Fi роутер и любой ноутбук, и они **гарантированно** поймут друг друга.

Отличие коммутатора (Switch) от концентратора (Hub)

- **Концентратор (Hub)** — устройство **физического уровня (1-й уровень модели OSI)**. Он принимает электрический сигнал (биты) на один порт и **безусловно ретранслирует (повторяет) его на все остальные активные порты.** Hub реализует топологию "общая шина" в форм-факторе "звезда". Не анализирует кадры, не знает MAC-адресов. Всё подключенное к нему устройство находится в **едином домене коллизий (collision domain)**.
- **Коммутатор (Switch)** — устройство **канального уровня (2-й уровень модели OSI)**. Он **анализирует входящие кадры Ethernet (фреймы)**, считывает MAC-адрес источника (чтобы запомнить, с какого порта пришёл этот адрес) и **MAC-адрес назначения**. На основе внутренней таблицы коммутации (MAC-адрес -> номер порта) он направляет кадр **только на тот порт, к которому подключен получатель**, или на

все порты (флуд), если адрес назначения ещё не известен. Каждый порт свитча — это **отдельный домен коллизий** (при полно-дуплексном режиме коллизий вообще не возникает).

Протокол CSMA/CD

Множественный доступ с прослушиванием несущей и обнаружением коллизии

1. **"Послушай, прежде чем говорить" (Carrier Sense):** Прежде чем что-то сказать, ты **обязан прислушаться**. Если в рупоре тишина — можно начинать.
2. **"Любой может начать говорить в тишину" (Multiple Access):** Все в комнате равны. Кто первый начал в момент тишины, тот и вещает.
3. **"Говори и слушай одновременно" (Collision Detection):** Пока ты говоришь, ты **продолжаешь прислушиваться** к рупору. Вдруг окажется, что другой человек в другом конце комнаты тоже не услышал тебя и начал говорить одновременно с тобой. Ваши голоса смешаются в неразборчивый гул — это **коллизия**.
4. **"Если столкнулись — замолчи, предупреди всех и попробуй позже" (Jam Signal & Backoff):** Как только ты понял, что произошла коллизия (услышал не свой голос), ты:
    - **Сразу кричи "Эй, все молчать!" (jam-signal)**, чтобы и другие поняли, что была коллизия.
    - **Замолкаешь.**
    - **Ждёшь случайное время** (например, 2 секунды, потом 4, потом 8... но не бесконечно). Это чтобы вы и ваш "соперник" не начали говорить снова одновременно.
    - **Повторяешь попытку с пункта 1.**

Модель osi

Представляет из себя путь, который данные проходят при отправке/получении

Декапсуляция (получаем данные)

-физический уровень достает биты из полученных ему сигналов/импульсов

-канальный уровень смотрит на начало и конец битов, подсчитывает контрольную сумму (src). Если все верно, достает кадр и сверяет mac-адрес с mac-адресом получателя. Если все хорошо достает внутренний ip-пакет и передает выше

-сетевой уровень принимает ip-пакет, далее сверяет его с ip-адресом получателя. Если все хорошо, достает блок(tcp\udp) и передает его выше

-транспортный уровень принимает блок, смотрит port по которому нужно передать данные с этого блока (если блок tcp то данные нужно проверить и доставить без потерь)

-сеансовый уровень контролирует процесс передачи данных получателю, следит, и если нужно, реанимирует передачу

-уровень представления декодирует/распоковывает данные (по сути делает так, что бы приложение могло понять что ему пришло на вход)

-прикладной уровень непосредственно показывает полученные данные пользователю

Инкапсуляция (отправляем данные)

- Прикладной уровень. Пользовательское приложение формирует данные для отправки

- Уровень представления. Получает данные от прикладного уровня. Преобразует (кодирует) их для безопасной и эффективной передачи: шифрует (например, по TLS), сжимает, конвертирует в нужную кодировку.

- Сеансовый уровень. Устанавливает, управляет или поддерживает сеанс связи с удалённым хостом. Добавляет информацию для синхронизации диалога (контрольные точки).

- Транспортный уровень. Получает поток данных от верхних уровней. Дробит их на сегменты (для TCP) или датаграммы (для UDP). Добавляет транспортный заголовок, в котором указываются:

    - Порты отправителя и получателя (чтобы ОС на другом конце знала, какому приложению отдать данные).

- Для TCP: Номера последовательности, флаги управления, окно — всё для надёжной доставки.

На выходе: TCP-сегмент или UDP-датаграмма.

- Сетевой уровень. Получает сегмент/датаграмму от транспортного уровня. Добавляет IP-заголовок, создавая IP-пакет. В заголовке главное:

- IP-адреса отправителя и получателя (чтобы маршрутизаторы знали, куда вести пакет).
- TTL (время жизни), протокол (TCP=6, UDP=17) и др.

На выходе: IP-пакет (Packet).

- Канальный уровень. Получает IP-пакет от сетевого уровня. Добавляет заголовок и "хвост" (трейлер), создавая кадр (Frame). В заголовке кадра главное:

- MAC-адреса отправителя (свой) и получателя (обычно MAC-адрес следующего "прыжка" — шлюза или конечного узла).
- Тип инкапсулированного протокола (чаще всего IPv4 или IPv6).

В трейлер добавляет контрольную сумму (FCS/CRC) для проверки целостности на стороне получателя.

На выходе: Кадр (Frame).

- Физический уровень. Получает кадр (последовательность бит) от канального уровня. Кодирует эти биты в сигналы, соответствующие физической среде: электрические импульсы (для витой пары), модулированный свет (для оптики), радиоволны (для Wi-Fi). Передаёт эти сигналы в среду передачи (кабель, воздух).

Протоколы на уровнях оси:

- Фиизческий: провода, радиоволны

- Канальный:

- LLC - Его основная задача — обеспечить **единый интерфейс** для различных протоколов сетевого уровня (IP, IPX) к разным технологиям нижнего подуровня MAC (Ethernet, Token Ring, Wi-Fi). Он обеспечивает управление потоком данных и обнаружение ошибок на стыке между уровнями
- PPP - Протокол для **прямого соединения** двух узлов (например, компьютер ↔ провайдер по модему). Используется для установления связи, аутентификации
- ARP - Работает на стыке канального и сетевого уровней. Его задача — найти **MAC-адрес** устройства, зная его **IP-адрес** в локальной сети. Без ARP пакеты не смогли бы уйти с компьютера.

- Сетевой:

- IP - протокол сетевого уровня, обеспечивающий адресацию и маршрутизацию пакетов.
- ICMP - это вспомогательный протокол сетевого уровня, используемый для отправки служебных и диагностических сообщений об ошибках и исключительных ситуациях в IP-сети.
- OSPF – это протокол динамической маршрутизации состояния канала, в котором маршрутизаторы обмениваются информацией о состоянии своих соединений, строят полную карту топологии сети и с помощью алгоритма Дейкстры выбирают кратчайший путь на основе стоимости канала, обеспечивая быструю сходимость и масштабируемость за счёт иерархического деления сети на области
- EIGRP – это протокол маршрутизации, разработанный Cisco, который использует алгоритм DUAL для быстрого расчёта маршрутов, обменивается только изменениями маршрутов с соседями и выбирает оптимальный путь на основе совокупных характеристик канала, таких как пропускная способность и задержка, обеспечивая высокую скорость сходимости и эффективность передачи данных
- RIP – это дистанционно-векторный протокол динамической маршрутизации, при котором маршрутизаторы периодически (раз в 30 секунд) обмениваются своими таблицами маршрутов. Каждый роутер знает только расстояние (хопы) и направление (соседа, через которого идти) до сети. Не знает полной топологии
- BGP – протокол динамической маршрутизации, который Обеспечивает маршрутизацию между разными автономными системами **(AS)**, т.е. между разными интернет-провайдерами или огромными компаниями.

Рассылает не метрику, а путь (AS_PATH) — список AS, через которые нужно пройти, чтобы достичь сети. Выбор пути основан на политиках и правилах, а не только на "скорости".

Это протоколы, с помощью которых **маршрутизаторы общаются друг с другом**, чтобы автоматически строить **таблицы маршрутизации** и узнавать лучшие пути до сетей назначения

- Траспортный:

- TCP – это надёжный, ориентированный на соединение протокол. Он гарантирует доставку данных без ошибок, в правильном порядке, без потерь и дубликатов. Для этого использует: установление соединения (3-ное рукопожатие), подтверждение приёма (квитанция), тайм-ауты и повторные передачи, управление потоком (скользящее окно) и контроль перегрузки
- UDP - это ненадёжный, не требующий установления соединения транспортный протокол. Он предоставляет минимальные услуги поверх IP: мультиплексирование/демультиплексирование по портам и проверку целостности данных. Не гарантирует доставку, порядок следования или защиту от дублирования пакетов

- Сеансовый:

- Представления:

- Прикладной:

Прозрачный мост

Прозрачный мост - устройство канального уровня, которое соединяет нексолько устройств внутри локальной сети. Допустим у нас есть сеть где содержится 3 пк и 1 свитч. По сути наш прозрачный мост это и есть наш свитч. Наш мост получает кадр, и смотрит по своей таблице куда и кому отправить этот кадр (смотрит mac-адрес и port). Если у нашего моста нет адреса получателя, происходит флуд, после которого он запишет себе адрес получателя

Прозрачный мост — это **устройство канального уровня (Level 2)**, которое соединяет несколько сегментов локальной сети (LAN) и **изучает MAC-адреса**, чтобы принимать решение о пересылке кадров. Он строит таблицу

соответствия MAC-адресов и портов, на которых эти адреса были обнаружены. Кадр пересылается **только на тот порт, где находится адресат**, или флудится (отправляется на все порты), если адрес неизвестен. "Прозрачный" означает, что хостам в сети не требуется специальная настройка для работы с ним

Алгоритм скользящего окна:

Это алгоритм, используемый в протоколе TCP для управления потоком данных. Отправителю разрешается передавать несколько сегментов подряд, не дожидаясь подтверждения (ACK) на каждый. Количество байт, которые можно отправить без подтверждения, называется размером окна. Окно «скользит» вправо по мере получения подтверждений от получателя. Размер окна динамически меняется: получатель указывает свой размер окна приёма (rwnd) в каждом ACK, а отправитель вычисляет окно перегрузки (cwnd) для контроля загрузки сети. Фактический размер окна — минимум из (rwnd, cwnd).

Структура СО: размер окна, скорость передачи, номер пакета, номер принятого пакета, первый неподтвержденный пакет, номер следующего пакета для отправки

NAT:

Nat – технология, которая представляет из себя переводчик адресов. Nat – делиться на :

Static NAT (Статический): Где каждому локальному адресу ставится строго 1 внешний адрес

Dynamic NAT (Динамический): Есть Пул внешних адресов который nat динамически выделяет внутренним хостам на время сессии.

PAT: Для всех внутренних хостов 1 внешний адрес. Для того что бы рахличать хосты к адресу приписываются номера портов

NAT позволяет представить внешнему миру внутреннюю структуру IP-адресации предприятия иначе, чем она на самом деле выглядит

VLAN:

Технология, которая позволяет физическую сеть разделить на логические подсети.

Зачем это нужно?

1. **Безопасность:** Устройства из разных VLAN не могут общаться напрямую на канальном уровне (L2). Для обмена данными им нужен маршрутизатор (L3), где можно настроить фильтры (ACL).
2. **Уменьшение широковещательных доменов:** Широковещательный трафик (например, ARP) ограничивается рамками одного VLAN и не "зашумляет" всю физическую сеть.
3. **Гибкость:** Устройства можно объединять в сеть по **функциональному признаку** (например, все IP-камеры в VLAN "Видеонаблюдение"), а не по физическому расположению. Чтобы переместить устройство в другой VLAN, достаточно изменить настройку порта коммутатора, а не перекладывать кабели.
4. **Управление трафиком**

Тройное рукопожатие:

Процесс установления соединения перед началом передачи данных по TCP. Состоит из трёх шагов:

1. **SYN:** Клиент отправляет сегмент с флагом **SYN=1** и случайным начальным номером последовательности (**Seq=X**).
2. **SYN-ACK:** Сервер отвечает сегментом с флагами **SYN=1, ACK=1**. Указывает свой начальный номер последовательности (**Seq=Y**) и номер подтверждения **Ack=X+1**.
3. **ACK:** Клиент отправляет сегмент с флагом **ACK=1**, где **Ack=Y+1**. Начинается передача данных.

2. Доступным языком:
Это как **вежливое начало телефонного разговора**:

1. **Клиент:** "Алло, это Васенька. Можем поговорить?" (**SYN**)

2. **Сервер:** "Да, Васенька, я вас слышу! Это Петрович. Говорите!" (**SYN-ACK**)
3. **Клиент:** "Отлично, Петрович, слушаю!" (**ACK**)

Теперь соединение установлено, можно обсуждать дела (передавать данные).

Четверное рукопожатие:

Корректное завершение TCP-соединения, когда обе стороны закончили передачу. Состоит из четырёх (иногда трёх) шагов:

1. **FIN:** Сторона, завершающая соединение, отправляет сегмент с флагом **FIN=1**.
2. **ACK:** Вторая сторона подтверждает получение FIN сегментом **ACK**.
3. **FIN:** Когда вторая сторона тоже готова закрыться, она отправляет свой **FIN**.
4. **ACK:** Первая сторона отправляет финальное **ACK**. После тайм-аута соединение закрывается.

2. Доступным языком:
Вежливое окончание разговора:

1. **Вася:** "Петрович, у меня всё, спасибо!" (**FIN**)
2. **Петрович:** "Понял, Васенька!" (**ACK**) *Петрович может ещё что-то дослать.*
3. **Петрович:** "Мне тоже всё, до связи!" (**FIN**)
4. **Вася:** "Договорились, всего доброго!" (**ACK**)

Разъединение завершено.

ACL:

ACL — это упорядоченный набор правил (записей), которые используются сетевым устройством (маршрутизатором) для фильтрации трафика (разрешения или запрета прохождения пакетов) на основе заданных

критериев. ACL применяется к интерфейсам устройства и проверяет каждый проходящий пакет. Работает по принципу «первое совпадение».

Стандартный ACL – выполняет фильтрацию пакетов исключительно на основе адреса отправителя

Расширенный ACL - выполняет фильтрацию пакетов на основе расширенного набора критериев

Мультиплексирование и демультиплексирование на транспортном уровне:

Мультиплексирование — это технология объединения нескольких независимых потоков данных в один общий физический канал для передачи по сети с использованием общих идентификаторов (портов, IP-адресов). **Демультиплексирование** — это обратный процесс разделения общего потока данных на отдельные потоки и доставки каждому целевому приложению.

Мультиплексирование (отправка) — это когда **много программ** пихают свои данные в **один общий провод**, подписывая их своими «номерками» (портами).

Демультиплексирование (получение) — это когда данные из **общего провода** разносятся по **разным программам** по этим «номеркам»

Специальные адреса

Unicast (Одноадресная)

- **Назначение:** Один отправитель → один получатель**.**

. Multicast (Групповая)

- **Назначение:** Один отправитель → группа устройств, подписанных на определённый адрес

Broadcast (Широковещательная)

- **Назначение:** Один отправитель → все устройства в сегменте сети.

Специальные (зарезервированные) IP-адреса

0.0.0.0 - адрес по умолчанию

127.0.0.1 - Loopback

255.255.255.255 – Локальный broadcast

x.255.255.255 – broadcast (вроде)

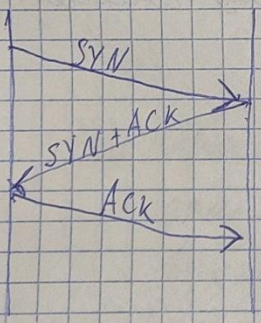
данные	прикладной	HTTP(S), DNS, FTP, SMTP
данные	представления	TLS/SSL, JPeG, MPEG
данные	сеансовый	RAP, RPC
блоки	транспортный	TCP, UDP
пакеты	сетевой	IP, ICMP, OSPF, EIGRP, BGP, RIP
кадры	канальный	L2C, PPP, ARP
биты	физический	~~IEEE, ADSL, FTTB(H)~~ ~~USB~~ провода, радиоволны

Прозрачный мост

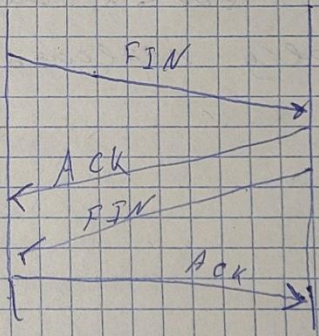
Пк1 ——— Switch ——— Пк2 Физически

Пк1 ——————————— Пк2 Логически

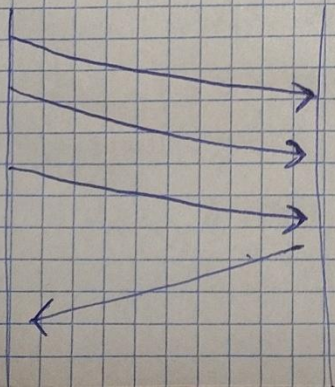
3-ое Рукопожатие

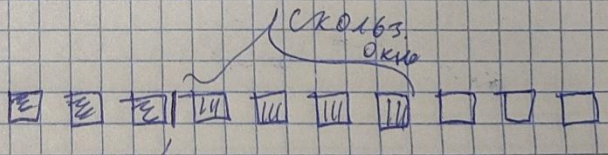


4-ое Рукопожатие



Скользящее окно





подтвердил

Получатель говорит: я получил еще 3 ~~пока~~ мое окно = 4