Модель OSI, содержит уровни и каждый отдельно взятый уровень взаимодействует ТОЛЬКО либо с верхним либо с нижним. Взаимодействие отдельно взятого уровня через один вверх или вниз невозможно, это даёт возможность при внесении изменений в отдельно взятый уровень изменять только связанные с ним вышестоящий и нижестоящий уровни, а не изменять всю модель.

протоколы - это стандарты, которые определяют каким образом, будут обмениваться данными различные программы.

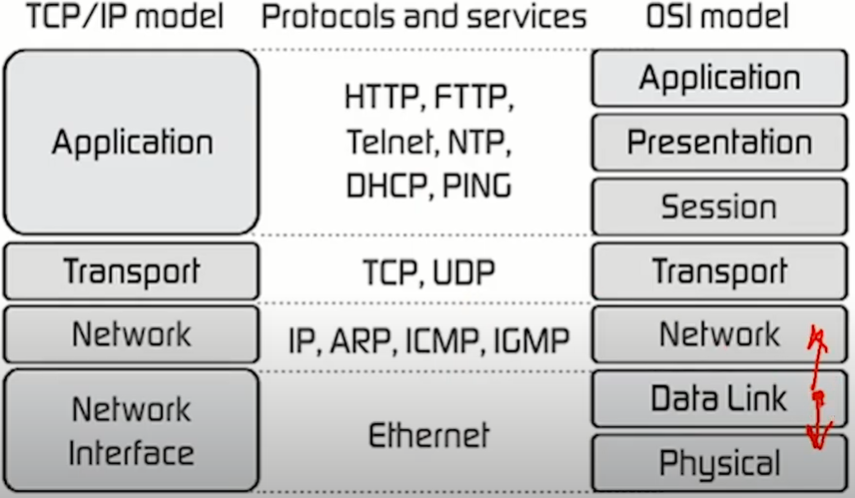
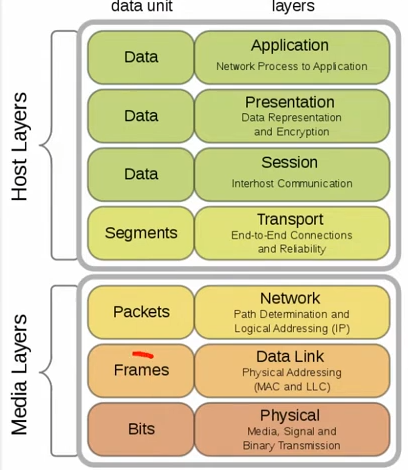
Если устройство имеет IP адресс - то это хост.

На самом нижнем уровне модели OSI данные представляют собой физические объекты — ток, свет или радиоволны. Они передаются по проводам или с помощью беспроводных сигналов.

Самый известный протокол на физическом уровне — Ethernet. Он описывает, как сигналы кодируются и передаются по проводам. Кроме него, есть ещо протоколы: Bluetooth, Wi-Fi и ИК-порт, которые также содержат инструкции для передачи данных.

*Процесс, когда данные передаются с верхних уровней на нижние называется* ***инкапсуляцией*** *данных, а когда наоборот, наверх, с первого, физического к седьмому, то этот процесс называется* ***декапсуляцией*** *данных*

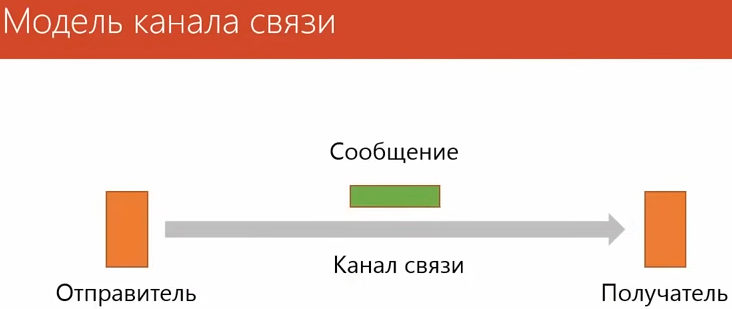
\*Паша\*

1) Физический уровень

Задача - передача потока данных (поток бит), представить биты информации в виде сигналов.

Если коротко то данный уровень передает поток данных в виде нераспозноваемых потоков бит и как он это делает неважно.



Есть: канал связи, отправитель, получатель, передаваемое сообщение.

Канал связи это: - Телефонный кабель (устарел)

* Коаксиальный кабель
* Витая пара
* Оптический кабель (оптоволокно, передаёт свет)
* Радиоволны (WIFI)

Характеристики: а)-пропускная способность (bit / sec)

б)-задержка (сколько времени идёт сообщение).

-совместно эти две характеристики формируют скорость работы канала связи.

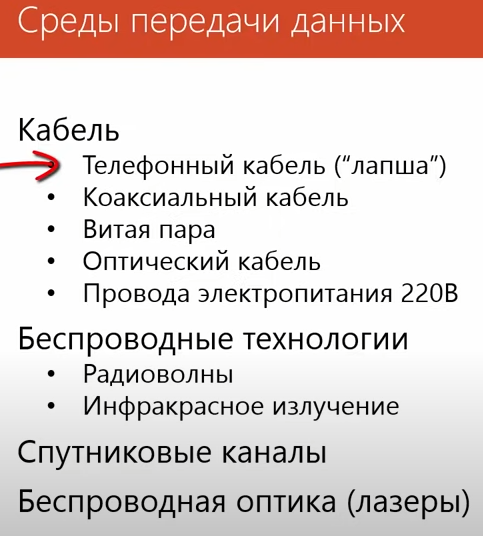
в)-количество ошибок.

Типы каналов связи бывают:

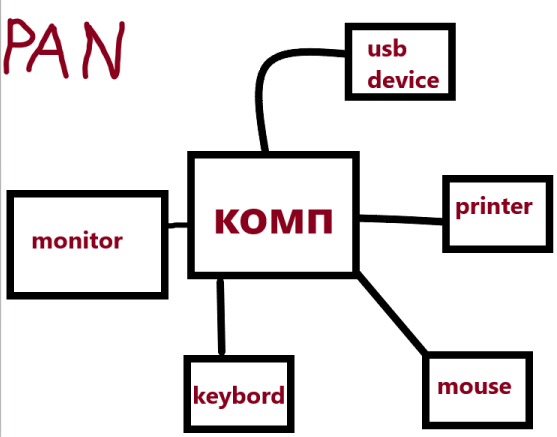
1)-Симплексный (данные можно передавать только в одну сторону)

2)-Дуплексный, можно передавать данные в обе стороны одновременно

3)-Полудуплексный, можно передавать данные в обе стороны но по очереди.



Данный уровень как мне кажется реализован в PAN (personal area network) соединение по кабелю



2) Канальный уровень. Уровень SWITCH когда соединяется несколько локальных компов/хостов.

Задача - А)- передача сообщений по каналам связи (такие сообщения называются кадрами / frame)

Б)- определяется где в потоке бит начало сообщения и где конец.

В)- коррекция ошибок.

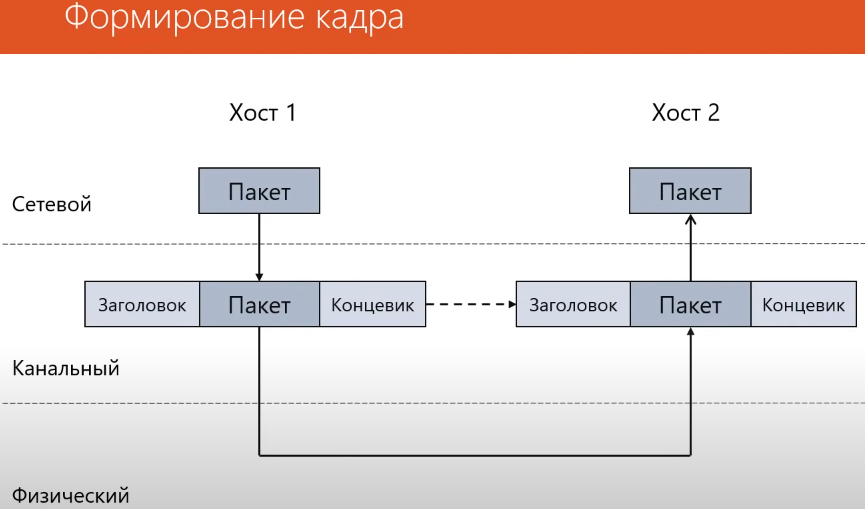
Когда получили распостронения разделяемые каналы связи были введены подуравни, т.е. когда к одной и той же среде передачи данных подключено несколько устройств, модель пришлось изменить. Канальный уровень был разделён на два уровня: 1)-Подуровень управления логическим каналом Logical Link Control (LLC), отвечает за передачу данных (создание кадров, обработка ошибок) 2)-Подуровень управления доступом к среде Media Access Control (MAC)

Для некоторых типов каналов с множественным доступам (несколько устройств используют один канал для передачи данных, например 2 компа подключены через switch по кабелю), данный уровень решает доп. задачи:

1)Адресация-если на канале несколько устройств то надо определить какому устройству предназначено текущее сообщение.

2)Согласованный доступ к каналу, если все устройства начнут передавать данные одновременно то сообщение исказится

А) как происходит передача и формирование “кадров”

Хост 1 это компьютер который отправляет сообщение, хост 2 это компьютер принимающий сообщение. На сетевом уровне отправителя формируется сообщение (пакет), далее на канальном уровне добавляется заголовок и концевик, данное сообщение называется уже “кадр” Каждый кадр/фрейм состоит из служебной информации — например, адреса отправителя и адреса получателя и передается по физическому уровню получателю. Получатель (хост 2) также получив кадр через физический уровень передаёт его на канальный, где считываются заголовок и концевик и отправляет далее на более высокий уровень.

Б) как в потоке бит указать где начинаются и где заканчиваются отдельные кадры.

Для этого существует несколько методов:

1)-В начале кадра указать длинну етого кадра, например здесь 6 обозначает количество бит для данного кадра, далее следующий кадр имеет 8 бит… 

на практике такой метод не применяется, так - как если была допущена ошибка то все последующие кадры(а именно их длинна будет искажена )

2)-Вставка в начала и конец кадра специальных последовательностей байтов и бит, недостаток в том что в тексте могут встречаться одинаковые последовательности таких-же бит и байт и тогда опять-же возникнет ошибка (будет не понятно где начало,где конец сообщения).

3)-Вставка бит, применяется в более современных протоколах (HDLC, PPP), здесь перед началом и концом каждого кадра добавляется последовательность из битов “01111110”

В) Обнаружение и исправление ошибок на канальном уровне могут быть следующих видов:

1) Просто обнаружение ошибок ,если кадр повреждён он просто отбрасывается, никаких попыток восстановить данные не производится.

2) Исправление ошибок в поврежденных кадрах, связанно именно с исправлением…

3) Повторная отправка, данный метод используется вместе с просто обноружением ошибки. Когда отправитель передал данные он запускает таймер во время которого ожидает ответ от получателя, если получатель получил сообщение корректно то он высылает отправителю ответ “что всё пришло, можно отправлять следующую часть”, отправитель получив ответ отправляет следущую часть. Если получатель не получил корректно сообщение то он просто не отправляет ответ, что для отправителя служит сигналом о том , что надо выслать то же самое сообщение повторно, так-как оно не было получено. Есть два варианта метода повторной отправки: 1)Остановка и ожидание, отправитель отправляет кадр, останавливается и ожидает ответа, следующий кадр передаётся **только после получения подтверждения** от получателя, этот способ используется в WiFi. 2)Скользящее окно - отправитель высылает несколько кадров не дожидаясь ответа от получателя, количество кадров - называется размером окна, получатель отвечает только на последнее сообщение в окне

Технологии канального уровня: Ethernet, WiFi.

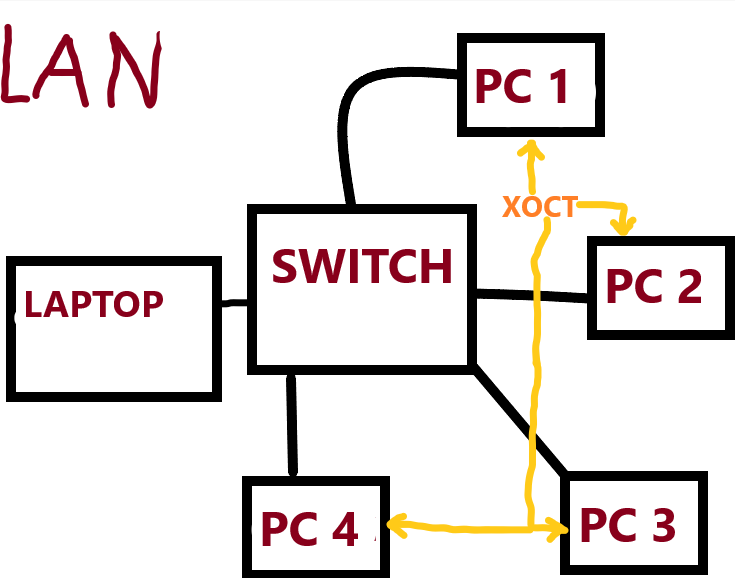
В локальных сетях канальный уровень разделяется на два подуровня:1) уровень управления логическим каналом (logical link control, LLC) нужен для взаимодействия с верхним уровнем. 2) уровень доступа к среде (media access layer, MAC)

на канальном уровне адресация узлов осуществлялась при помощи физического МАС-адреса сетевого устройства У любого устройства, способного подключиться к глобальной сети, есть специальный идентификатор, который называется MAC-адрес. Зачастую он нужен для настройки сетевого подключения. MAC-адрес является одной из основных характеристик сетевого адаптера, MAC (Media Access Control) адрес или физический адрес используется для уникальной идентификации устройств в локальной сети. Он записывается на заводе-производителе в постоянную (энергонезависимую) память устройства, например сетевой карты компьютера или маршрутизатора, когда на заводе собирают ноутбук или смартфон, ему сразу же присваивают определённый MAC-адрес, который потом уже никак нельзя поменять. MAC-адрес настольных ПК зашит в сетевую карту, поэтому его можно изменить, только заменив эту самую карту. Можно сказать,что MAC это физическии адрес устройства. как выглядит MAC адрес



Канальный уровень — уровень [сетевой модели OSI](https://www.oslogic.ru/knowledge/245/modeli-osi-i-tcp-ip/), предназначенный для обмена данными между узлами находящимся в том же сегменте локальной сети, путем передачи специальных блоков данных, которые называются кадрами (frame). В процессе формирования кадров данные снабжаются служебной информацией (заголовком), необходимой для корректной доставки получателю, и, в соответствии с правилами доступа к среде передачи, отправляются на [физический уровень](https://www.oslogic.ru/knowledge/720/fizicheskij-uroven-modeli-osi/). Кадры канального уровня не пересекают границ сетевого сегмента. Межсетевая маршрутизация и глобальная адресация это функция более высокого уровня, что позволяет протоколам канального уровня сосредоточится на локальной доставке и адресации.

Наверно реализацией является LAN-(local area network), также реализована кабелями, протокол PPP (point to point protocol) здесь вводится понятие хоста - какойто вычислительной единице способной принимать и отправлять какие-то запросы/ответы



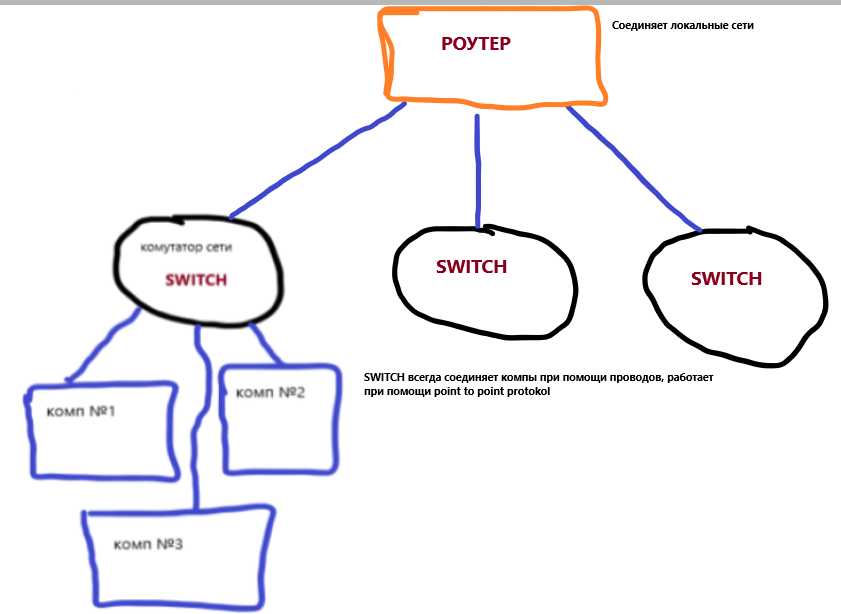
3) Сетевой уровень. Уровень роутера.

Здесь появляется понятие — маршрутизация и «IP-адреса».. Для этой задачи были созданы устройства третьего уровня — маршрутизаторы (их еще называют роутерами). Маршрутизаторы получают MAC-адрес от коммутаторов с предыдущего уровня (switch) и занимаются построением маршрута от одного устройства к другому (от одного роутера к другому) с учетом всех потенциальных неполадок в сети.

Данные на сетевом уровне представляются в виде пакетов. Такие пакеты похожи на фреймы из канального уровня, но используют другие адреса получателя и отправителя — IP-адреса. Чтобы получить IP-адрес обоих устройств (отправителя и получателя), используется протокол [ARP](https://ru.wikipedia.org/wiki/ARP) (Address Resolution Protocol). Он умеет конвертировать MAC- в IP-адрес и наоборот.

Здесь осуществляется маршрутизация трафика. Когда пользователь, к примеру, желает перейти на сайт и вводит его адрес, отправляется DNS-запрос. Ответом на него будет IP-адрес, который подставляется в пакет данных. Устройствами здесь являются роутер или маршрутизатор.

На данном уровне речь идёт об объединении неких локальных сетей в более глобальную



4) Транспортный уровень



Все семь уровней модели OSI можно условно разделить на две группы:

* Media layers (уровни среды),
* Host layers (уровни хоста) или Application layer

Уровни группы Media Layers (L1, L2, L3) занимаются передачей информации (по кабелю или беспроводной сети), используются сетевыми устройствами, такими как коммутаторы, здесь данные передаются от одного устройства до другога маршрутизаторы и т.п. Уровни группы Host Layers (L4, L5, L6, L7) используются непосредственно на устройствах, будь то стационарные компьютеры или мобильные устройства.

главной задачей является транспортировка пакетов/данных между **ПРОЦЕССАМИ** (как я понимаю приложениями) устройств сети. Естественно, при транспортировке возможны потери, но некоторые типы данных более чувствительны к потерям, чем другие. Например, если в тексте потеряются гласные, то будет сложно понять смысл, а если из видеопотока пропадет пара кадров, то это практически никак не скажется на конечном пользователе. Поэтому при передаче данных, наиболее чувствительных к потерям на транспортном уровне, используется протокол TCP, контролирующий целостность доставленной информации.

При передаче по протоколу TCP данные делятся на сегменты. Сегмент — это часть пакета. Когда приходит пакет данных, который превышает пропускную способность сети, пакет делится на сегменты допустимого размера. Сегментация пакетов также требуется в ненадежных сетях, когда существует большая вероятность того, что большой пакет будет потерян. При передаче данных по протоколу UDP пакеты данных делятся уже на датаграммы. Главное отличие датаграмм — в автономности. Каждая датаграмма содержит все необходимые заголовки, чтобы дойти до конечного адресата, поэтому они не зависят от сети, могут доставляться разными маршрутами и в разном порядке.

Важной особенностью уровня является прямое взаимодействие транспортного уровня одного компьютера с транспортным уровнем другого компьютера, т.е. без задействования Media layers., другими словами это сквозное соединение - в котором между двумя взаимодействующими хостами(компами) может находится большое количество сетевых устройств (из Media layers) но они не влияют на работу транспортного уровня.

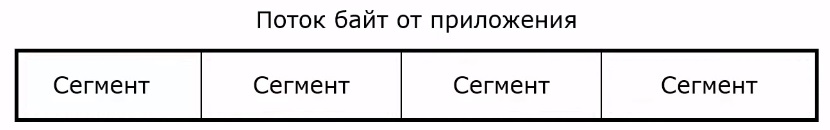
WEB- работает на TCP / IP

Два главных протокола здесь — TCP и UDP. Они как раз и отвечают за то, как именно будут передаваться данные.

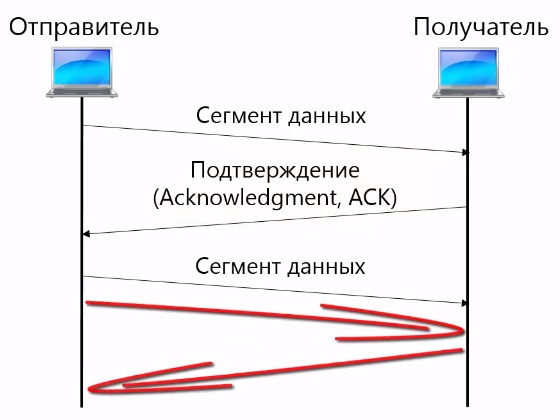
TCP (Transmission Control Protocol) — это протокол, который гарантирует доставку данных в корректном виде. Он жёстко следит за каждым битом информации, но работает гораздо медленнее UDP.

TCP обеспечивает как гарантию доставки данных, так и гарантию сохранения порядка следования сообщений.

От приложения, протокол TCP получает поток байт, который может быть очень большим. Например, вы можете скачивать из интернета файл, который составляет несколько мегабайт или несколько гигабайт. Данные файлы приходят на транспортный уровень в виде одного большого потока байт.



В протоколе TCP поток байт делится на отдельные части, которые называются сегменты. Каждый сегмент отправляется отдельно получателю. Получатель со своей стороны, принимает сегменты, собирает их в один большой поток байт и отправляет этот поток байт приложению. Для того чтобы обеспечить гарантию доставки данных, TCP использует подтверждение получения сообщения. Рассмотрим, как это работает. Отправитель пересылает по сети некоторый сегмент данных, получатель принимает сегмент и посылает отправителю подтверждение, сокращенно ACK от английского Acknowledgment, которая говорит о том что сегмент данных получен. Затем отправляется следующий сегмент данных, снова подтверждение и так далее.



Что происходит, если произошла ошибка при передаче данных? Сегмент данных потерян в сети, он не доходит до получателя, получатель не отправляет подтверждение сообщения. Отправитель при отправке сегмента устанавливает таймер, который задает время ожидания подтверждения, если в течении этого времени подтверждение не пришло, таймер срабатывает и тот же самый сегмент отправляются повторно.

UDP (User Datagram Protocol). Он уже не проверяет цельность битов, его задача — как можно быстрее передать данные с одного устройства на другое.

Для адресации (от одного приложения к другому приложению) на данном уровне используются порты(порт это просто число от 1- 65535). Для примера в Google происходит самостоятельное добавление порта 80. Опаньки - оказывается протокол HTTP работает на порте 80!

Пример сетевого взаимодействия с использованием IP адресов и портов:

К примеру у нас есть сервер на котором работает веб сервер на порту № 80, И есть клиент который хочет подключится к этому вебсерверу. Клиент открывает браузер, операционная система автоматически назначает ему порт 50298, браузер выполняет соединение с вебсервером, запрашивает вебстраницу , вебсервер отправляет ему эту страницу. К примеру клиент открыл ещо один браузер на том же компе и хочет зайти на туже вебстраницу с другого браузера, операционная система опять же назначила браузеру номер порта 50302, браузер запрашивает страницу у вебсервера. Вебсервер видит не только IP адрес хоста (компа), который в обоих случаях одинаковый, но и номер порта, который в обоих случаях разный, поэтому когда приходит ответ то он отправляется именно в тот браузер из которого был зделан запрос, а не в другой. **СОКЕТ** - это грубо говоря связка ip адреса хоста и номера порта, по другому это называют интерфейсом

5) Сеансовый уровень

обеспечивает поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизацией задач, определением права на передачу данных и поддержанием сеанса в периоды неактивности приложений.

Протоколы сеансового уровня:

ADSP ([**AppleTalk Data Stream Protocol**](https://ru.wikipedia.org/wiki/AppleTalk)),

ASP ([**AppleTalk Session Protocol**](https://ru.wikipedia.org/wiki/AppleTalk)),

H.245 ([**Call Control Protocol for Multimedia Communication**](https://ru.wikipedia.org/wiki/H.245)),

ISO-SP (OSI Session Layer Protocol (X.225, ISO 8327)),

iSNS ([**Internet Storage Name Service**](https://ru.wikipedia.org/wiki/Internet_Storage_Name_Service)),

L2F ([**Layer 2 Forwarding Protocol**](https://ru.wikipedia.org/wiki/Layer_2_Forwarding_Protocol)),

L2TP ([**Layer 2 Tunneling Protocol**](https://ru.wikipedia.org/wiki/L2TP)),

NetBIOS ([**Network Basic Input Output System**](https://ru.wikipedia.org/wiki/NetBIOS)),

PAP ([**Password Authentication Protocol**](https://ru.wikipedia.org/wiki/Password_Authentication_Protocol)),

PPTP ([**Point-to-Point Tunneling Protocol**](https://ru.wikipedia.org/wiki/PPTP)),

RPC ([**Remote Procedure Call Protocol**](https://ru.wikipedia.org/wiki/Remote_procedure_call)),

RTCP ([**Real-time Transport Control Protocol**](https://ru.wikipedia.org/wiki/RTCP)),

SMPP ([**Short Message Peer-to-Peer**](https://ru.wikipedia.org/wiki/SMPP)),

ZIP ([**Zone Information Protocol**](https://ru.wikipedia.org/wiki/Zone_Information_Protocol)),

SDP ([**Sockets Direct Protocol**](https://ru.wikipedia.org/wiki/Sockets_Direct_Protocol))..

Этот уровень позволяет пользователям на разных машинах устанавливать между собой активные сеансы связи. Он отвечает за установление, поддержку, синхронизацию и завершение сеансов между приложениями конечного пользователя.

The main functions of the session layer are as follows −

* It works as a dialog controller. It allows the systems to communicate in either half-duplex or full-duplex mode of communication.
* It is responsible for token management. Through this, it prevents the two users to simultaneously attempt the same critical operation.
* It synchronizes communication. It adds synchronization points or checkpoints in data streams for long communications. This ensures that data streams up to the checkpoints are successfully received and acknowledged. In case of any failures, only the streams after the checkpoints have to be re-transmitted.

token management -Это технология что-то типа если токен находится у одного user-а то только он может совершать какие-то действия, другой в это время ожидает своей очереди.

The session layer provides the mechanism for the opening, closing, and managing of sessions between applications.

Данный уровень используются для управления и отслеживания соединений между сетевыми узлами. Клиентское приложение используется для доступа к серверному приложению — это пример, который можно использовать для описания использования сеансов. Сетевой клиент, такой как Microsoft Outlook, должен будет взаимодействовать с сервером Microsoft Exchange для доступа к почтовому ящику пользователя.

Чтобы пользователь мог загружать новые сообщения из почтового ящика, необходимо установить соединение, чтобы можно было начать процесс загрузки. Во время связи с сервером почтовых ящиков вы можете инициировать различные другие соединения с другими серверами, используя различные клиентские приложения. Именно Сеансовый уровень отслеживает, какие пакеты и данные относятся к какой операции (или процессу, программе), которую вы инициировали, и отслеживает, какие из этих пакетов принадлежат какому вызвавшему их приложению.

6) Уровень представления (presentation)

**Представительский уровень или уровень представления данных (presentation layer) – он преобразует данные в соответствующий формат. На примере понять проще: те картинки (все изображения) которые вы видите на экране, передаются при пересылке файла в виде маленьких порций единиц и ноликов (битов). Так вот, когда Вы отправляете своему другу фотографию по электронной почте, протокол Прикладного уровня SMTP отправляет фотографию на нижний уровень, т.е. на уровень Представления. Где Ваша фотка преобразуется в удобный вид данных для более низких уровней, например в биты (единицы и нолики).**

**Именно таким же образом, когда Ваш друг начнет получать Ваше фото, ему оно будет поступать в виде все тех же единиц и нулей, и именно уровень Представления преобразует биты в полноценное фото, например JPEG.**

Этот уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Запросы приложений, полученные с уровня приложений, он преобразует в формат для передачи по сети, а полученные из сети данные преобразует в формат, понятный приложениям. На этом уровне может осуществляться сжатие/распаковка или кодирование/декодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

Другой функцией, выполняемой на уровне представлений, является шифрование данных, которое применяется в тех случаях, когда необходимо защитить передаваемую информацию от приема несанкционированными получателями. Чтобы решить эту задачу, процессы и коды, находящиеся на уровне представлений, должны выполнить преобразование данных. На этом уровне существуют и другие подпрограммы, которые сжимают тексты и преобразовывают графические изображения в битовые потоки, так что они могут передаваться по сети.

7)Прикладной уровень:

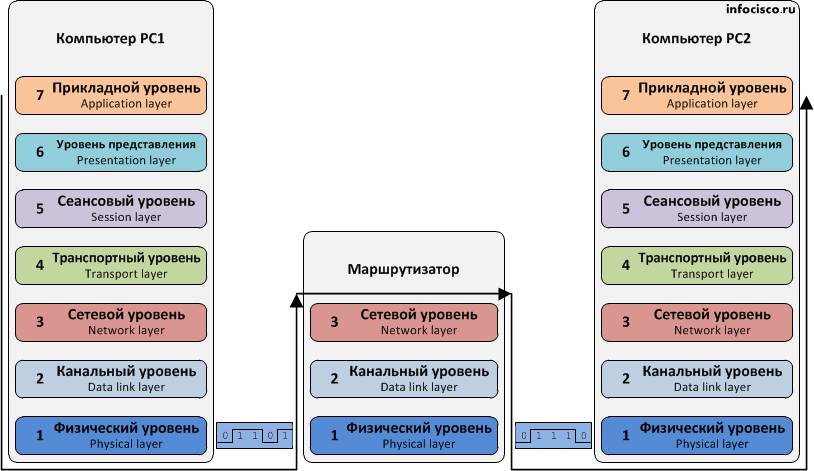
Он осуществляет связь пользовательских приложений с сетью. Эти приложения нам всем знакомы: просмотр веб-страниц (HTTP), передача и приём почты (SMTP, POP3), приём и получение файлов (FTP, TFTP), удаленный доступ (Telnet) и т.д.

# **Инкапсуляция**

Инкапсуляция – это процесс передачи данных с верхнего уровня приложений вниз (по стеку протоколов) к физическому уровню, чтобы быть переданными по сетевой физической среде (витая пара, оптическое волокно, Wi-Fi, и др.). Причём на каждом уровне различные протоколы добавляют к передающимся данным свою информацию.

Напомню, что сетевая модель OSI состоит из 7 уровней (уровень приложений, уровень представления, сеансовый, транспортный, сетевой, канальный и физический). Все сетевые устройства работают согласно модели OSI, только некоторые используют все 7 уровней, а другие меньше. Это позволяет обрабатывать поступающие данные в несколько раз быстрее.

Например, Ваш компьютер использует все 7 уровней, маршрутизатор – 3 нижних уровня, коммутатор – только 2 нижних уровня.



На рисунке Вы видите взаимодействие двух компьютеров, между которыми находится маршрутизатор. Компьютерами PC1 и PC2 могут быть как домашние компьютеры, так и сервера. Маршрутизатор, как и говорилось выше, работает только на трех уровнях модели, их (трех уровней) достаточно, чтобы проложить маршрут в любой сети.

## **Инкапсуляция и декапсуляция**

Проще будет разобрать эти процессы инкапсуляции и декапсуляции на примере. Допустим, Вы захотели посмотреть какую-то веб-страничку, ввели в адресную строк браузера адрес сайта и нажали кнопку Enter. После этого браузер должен отправить запрос на сервер (на котором хранится эта веб-страничка), с целью получения данных. Вот как раз на этом этапе, введённый Вами адрес сайта является данными, которые должны передаться на сервер в виде запроса.

Данные PDU 7 уровня модели OSI

Эти данные опускаются с уровня приложений, на уровень представления данных.

На этом уровне Ваш компьютер преобразует строку введенного текста (адреса) в формат удобный для передачи далее на нижний уровень.

Данные PDU 6 уровня модели OSI

Далее данные (уже не текст) поступают на сеансовый уровень, но на нём (в данном случае) нам нет необходимости использовать протоколы (этого уровня), и поэтому данные передаются далее.

Данные PDU 5 уровня модели OSI

Транспортный уровень получает данные и определяет, что дальше они должны быть переданы используя протокол TCP. Перед передачей транспортный уровень разбивает данные на кусочки данных и добавляет к каждому кусочку заголовок, в котором содержится информация о логических портах компьютеров (с какого данные были посланы (например 1223) и для какого предназначаются (в данном случае 80)). На транспортном уровне эти кусочки данных с заголовком называются сегментами. Сегменты передаются дальше вниз к сетевому уровню.

Данные PDU 4 уровня модели OSI

Сетевой уровень, получая каждый сегмент, разделяет его на еще более маленькие части и к каждой части добавляет свой заголовок. В заголовке сетевого уровня указываются логические сетевые адреса отправителя (Ваш компьютер) и получателя (Сервер).

Логические сетевые адреса – это всем известные IP-адреса, еще наверное непонятно что обозначают цифры и точки в них, но вскоре, этот пробел в знаниях заполнит соответствующая информация ;)

Эти маленькие кусочки данных уже с несколькими заголовками (на верхних уровнях тоже добавляются специфичные заголовки) на сетевом уровне называются пакетами, которые в свою очередь передаются на канальный уровень.

Данные PDU 3 уровня модели OSI

На канальном уровне пакеты разделяются на еще более маленькие кусочки данных, и к ним помимо опять добавляемого заголовка, только уже канального уровня, добавляется еще и трейлер. На этом уровне в заголовках содержатся физические адреса устройств – передающего и для кого они предназначаются, а в трейлере находится вычисленная контрольная сумма, некий код (информация), который используется для определения целостности данных.

Физические адреса устройств – это MAC-адреса.

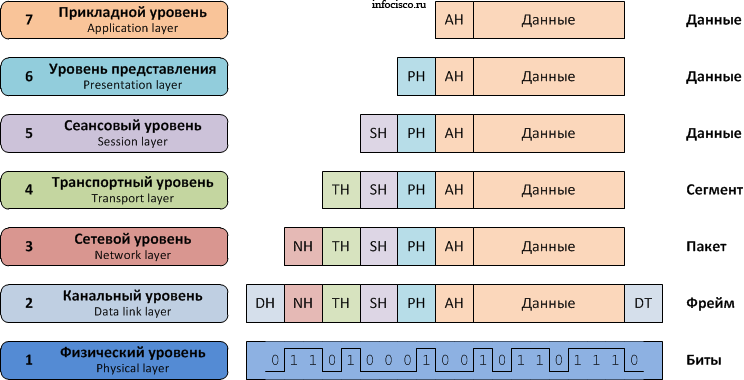
Эти очень маленькие кусочки данных именуются кадрами или фреймами (одно и тоже). Далее кадры передаются на физический уровень.

Данные PDU 2 уровня модели OSI

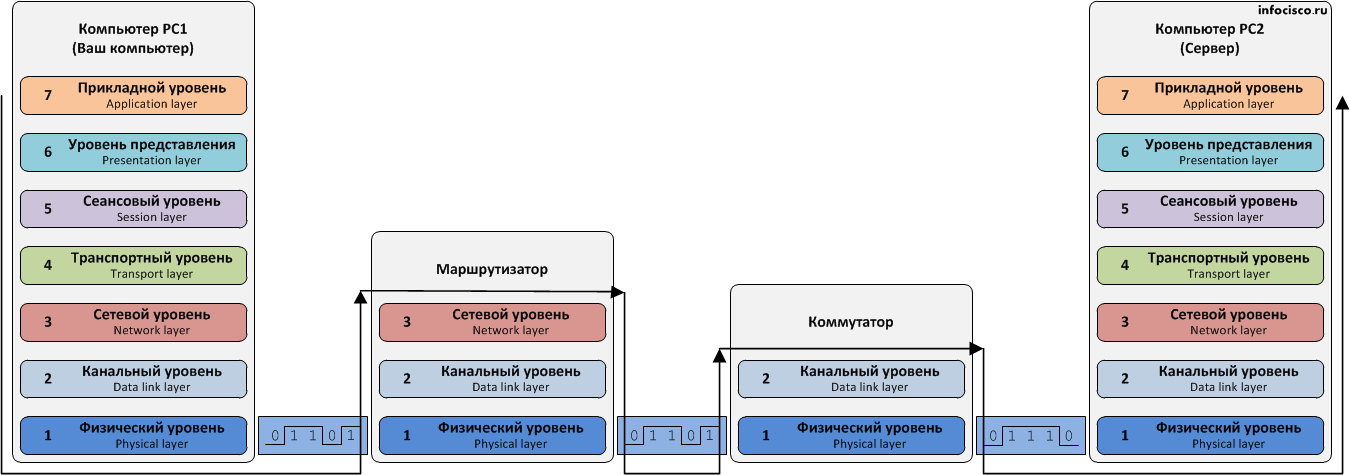
На физический уровень кадры передаются уже в виде сигналов битов и следуют через другие сетевые устройства в пункт назначения.

Данные PDU 1 уровня модели OSI

Весь процесс преобразования данных (с верхнего уровня) в сигналы (на нижний уровень) называется инкапсуляцией. Посмотрите на рисунок ниже, там представлена общая схема инкапсулирования с верхнего уровня на нижний:



Далее сигналы, проходя через несколько сетевых устройств (в нашем случае это маршрутизатор и коммутатор), доходят до получателя, в данном случае до сервера



Сетевая карта сервера принимает биты (на физическом уровне) и преобразует их в кадры (для канального уровня). Канальный уровень в обратной последовательности должен преобразовать кадры в пакеты (для сетевого уровня), только перед преобразованием уровень сначала смотрит на МАС-адрес (физический адрес) получателя, он должен совпадать с MAC-адресом сетевой карты, иначе кадр будет уничтожен. Затем канальный уровень (в случае совпадения MAC-адреса) высчитывает сумму полученных данных и сравнивает полученное значение со значением трейлера. Напомню, что значение трейлера высчитывалось на Вашем компьютере, а теперь оно, после передачи по проводам, сравнивается с полученным значением на сервере и если они совпадают, кадр преобразуется в пакет. Если проверочный код целостности данных рознится – кадр незамедлительно уничтожается.

На сетевом уровне происходит проверка логического адреса (IP-адреса), в случае успешной проверки пакет преобразуется в сегмент, попадая на транспортный уровень.

На транспортном уровне проверяется информация из заголовка, что это за сегмент, какой используется протокол, для какого логического порта предназначается и т.п. Протокол использовался TCP, поэтому назад на Ваш компьютер посылается уведомление о прибытии сегмента. Как говорилось выше (когда данные упаковывали в сегмент) в том случае использовался 80 порт назначения. Т.к. на веб-сервере как раз открыт этот порт, данные передаются дальше на верхний уровень.

На верхних уровнях запрос (введенный адрес сайта) обрабатывается веб-сервером (проверяется, доступна-ли запрашиваемая веб-страничка).

Этот процесс преобразования сигналов из провода в данные называется процессом **декапсуляции**.

После того, как страница будет найдена на сервере, она (текст, изображения, музыка) преобразуется в цифровой код, удобный для инкапсулирования. Большой объём данных делится на части и поступает ниже на уровень – транспортный. Там кусочек данных преобразуется в сегмент, только порт назначения теперь будет тот, с которого вы посылали (вспоминайте, 1223). Сегмент преобразуется в пакет, в заголовке которого содержится IP-адрес вашего компьютера и переходит ниже. На канальном уровне пакет в свою очередь преобразуется в кадры и добавляется заголовок и трейлер. В заголовок помещается МАС-адрес назначения (в данном случае это будет адрес шлюза), а в трейлер проверочный код на целостность данных. Далее сетевая карта посылает кадры в виде сигналов по кабелю по направлению к Вашему компьютеру.

**PDU -** Вам обязательно надо запомнить, что те кусочки данных (вместе с заголовками), которые переходят с уровня на уровень (с добавлением заголовков или наоборот) называются **Protocol Data Unit или PDU**. Если перевести литературно на русский язык, то получается **фрагмент данных на каждом уровне модели**

**Самая крутая серия статей про сети - http://infocisco.ru/articles.html**