**Уровни OSI. Назначение уровней.**

**7 уровней:**

* прикладной – предоставляет услуги, которые приложения используют для взаимодействия с сетью. Примеры включают протоколы передачи почты (SMTP), протоколы передачи файлов (TFTP) и протоколы удалённого рабочего стола (RDP);
* представления – преобразует данные в общий формат для передачи по сети; Примеры включают прикладной программный интерфейс (API), протоколы передачи файлов (FTP) и протоколы передачи гипертекста (HTTP)
* сеансовый уровень – поддерживает взаимодействие между удалёнными процессами. Сеансовый уровень - управляет установлением и разрывом соединения между двумя хостами. Пример – токен, который поддерживает уровень соединения на контрольных точках (авторизация).
* транспортный уровень – управляет передачей данных по сети, обеспечивает подтверждение передачи; гарантирует, что данные доставляются без ошибок и в правильном порядке. Примеры включают TCP (Transmission Control Protocol) и UDP (User Datagram Protocol)
* сетевой уровень – маршрутизация, управление потоками данных, адресация сообщений для доставки; преобразование логических сетевых адресов и имен в соответствующие им физические; отвечает за определение пути данных через различные сети и подсети, чтобы достичь пункта назначения. Пример – IP,
* канальный -контроль логической связи (llc): формирование кадров; контроль доступа к среде (mac): управление доступом к среде. Пример – Ethernet, Token Ring, ATM
* физический – обеспечивает битовые протоколы передачи информации. Пример – оптоволокно, коаксиал, витая пара.

**Сетевые сервисы:**

Сервис печати – есть устройство, которым можно управлять удалённо через сетевые устройства. Протоколы – PAP, IPP

Сервис СУБД (Протокол – SQL):

Файл-серверная архитектура: в этой модели данные хранятся на файловом сервере, и клиенты обращаются к серверу для получения доступа к данным. Файл-серверные СУБД, такие как Microsoft Access и FileMaker, являются популярными решениями для небольших приложений.

Серверная архитектура: здесь СУБД обрабатывает запросы клиентов, храня данные на сервере. SQL Server, Oracle и MySQL являются примерами серверных СУБД. Эти системы обеспечивают лучшую производительность и масштабируемость по сравнению с файл-серверными решениями.

Трехзвенная архитектура (трехуровневая архитектура): в этой архитектуре между клиентом и сервером базы данных вводится дополнительный уровень - сервер приложений или middleware. Это позволяет отделить уровень доступа к данным от бизнес-логики и пользовательского интерфейса, что улучшает масштабируемость и гибкость системы. Примерами трехуровневых систем являются Java EE и .NET.

Сервис сообщений: посыл сообщений удалённо – почта, голосовая почта, видеоконференции, телефония, http, чаты

Файловый сервис: управление, синхронизация файлов, распределение доступа, резервирование. Всё осуществляется через сеть, не надо применять дискеты. Протокол - ftp

Сервис приложений: основная задача – выдача вычислительных ресурсов Протоколы (http, ftp).

IaaS (инфраструктура как сервис) - это модель предоставления облачных вычислений, которая позволяет пользователям арендовать вычислительные ресурсы, такие как серверы, хранилища и сетевые компоненты. IaaS позволяет организациям снизить затраты на инфраструктуру и улучшить гибкость, поскольку они могут масштабировать свои ресурсы по мере необходимости.

SaaS (программное обеспечение как сервис) - это модель развертывания программного обеспечения, при которой приложение предоставляется как услуга через Интернет. SaaS упрощает развертывание и управление программным обеспечением, поскольку пользователи платят за использование приложения и не нуждаются в его установке или обслуживании.

PaaS (платформа как сервис) - это облачная платформа, которая предоставляет инструменты и инфраструктуру для разработки, тестирования и развертывания приложений. PaaS облегчает создание, тестирование и развертывание приложений, избавляя разработчиков от необходимости управлять базовой инфраструктурой.

Виртуализация на уровне ядра - это технология, которая позволяет запускать несколько операционных систем на одном физическом сервере. Эта технология позволяет использовать сервер более эффективно, так как несколько виртуальных машин могут работать на одном сервере, используя его ресурсы.

**Источники стандартов в телекоммуникациях, сетях и интернете, способы принятия стандартов.**

Разные источники стандартизации: международные (ISO, ITU), национальные (IEEE, ANSI), спец.комитеты и объединения (ATM, Fast Internet), передовые фирмы (Apple, IBM PC)

Стадии разработки стандартов ИСО:

Стадия 1: Стадия предложения

Стадия 2: Подготовительная стадия

Стадия 3: Стадия комитета

Стадия 4: Стадия вопросов

Стадия 5: Стадия одобрения

Стадия 6: Стадия публикации.

Примеры функций в модели OSI:

**Как на практике работает сетевая модель OSI**

В начале статьи мы задались вопросом: а как передаются сообщения в Telegram? Настало время на него ответить — и показать весь процесс передачи данных по модели OSI.

Мы хотим отправить сообщение нашему другу. Печатаем текст и нажимает кнопку «Отправить», а дальше перемещаемся внутрь компьютера.

**Прикладной уровень.**Приложение Telegram работает на прикладном уровне модели OSI. Когда мы печатаем текст сообщения и нажимаем кнопку «Отправить», эти данные передаются на сервер мессенджера, а оттуда — нашему другу.

Весь процесс проходит через API разных библиотек — например, для HTTP-запросов. Интерфейсы позволяют без лишних проблем обмениваться данными и не погружаться в то, как они представлены на низком уровне. Всё, что нужно знать, — это какую функцию вызвать и какие переменные туда передать.

**Уровень представления.**Здесь данные должны преобразоваться в унифицированный формат, чтобы их можно было передавать на разные устройства и операционные системы. Например, если мы отправляем сообщение c Windows на macOS, данные должны быть в читаемом для компьютеров Apple виде. Такая же ситуация и с другими устройствами.

Раз мы собираемся передать данные на другой компьютер, их нужно перевести в бинарный формат. После этого начнётся сам процесс передачи по сети.

**Сеансовый уровень.**Чтобы данные успешно передались сначала на сервер Telegram, а затем к нашему другу, приложению нужно установить соединение, или сеанс. Он обеспечивает синхронизацию между устройствами и восстанавливает связь, если она прервалась.

Благодаря сеансам вы можете видеть, что собеседник что-то печатает или отправляет вам картинки или видео. Но главная задача этого соединения — обеспечить стабильное соединение для передачи данных.

**Транспортный уровень.**Когда соединение установлено и данные унифицированы, пора передавать их. Этим занимается транспортный уровень.

Здесь данные разбиваются на сегменты и к ним добавляется дополнительная информация — например, номер порта и контрольные суммы. Всё это нужно, чтобы данные дошли до пользователя в целостности.

**Сетевой уровень.**Теперь данным нужно найти маршрут к устройству нашего друга, а затем отправить их по нему. Поэтому данные упаковываются в пакеты и к ним добавляются IP-адреса.

Чтобы получить IP-адрес устройств, которым нужно отправить пакеты, маршрутизаторы (устройства сетевого уровня) обращаются к ARP. Этот протокол быстро найдёт адрес получателя и отдаст его нам.

**Канальный уровень.**Здесь данные передаются от одного MAC-адреса к другому. Изначальный текст делится на фреймы — с заголовками и контрольными суммами для проверки целостности данных.

**Физический уровень.**И на самом нижнем уровне данные в виде электрических сигналов передаются по проводам, кабелям или по радиоволнам. Тут только одна задача — как можно быстрее откликаться на сигналы свыше.

После прохождения всех уровней модели OSI сообщение успешно доставляется на устройство нашего друга. Правда, в реальности это занимает всего миллисекунды.