Вопросы для защиты 2ой лабораторной работы

Программирование сетевых приложений

1. Перечислить все ***сетевые утилиты***.

| Наименование  утилиты | Назначение утилит |
| --- | --- |
| ping | Проверка соединения с одним или более хостами в сети |
| tracert | Определение маршрута до пункта назначения |
| route | Просмотр и модификация таблицы маршрутизации. Таблица маршрутизации — это набор правил, которые определяют, куда отправлять сетевые пакеты. |
| arp | Просмотр и модификация ARP-таблицы (сопоставление IP-адресов и MAC-адресов ) |
| ipconfig | Просмотр текущей конфигурации сети TCP/IP |
| nslookup | Диагностика DNS-серверов |
| hostname | Просмотр имени хоста |
| netstat | Просмотр статистики текущих сетевых TCP/IP-соединений |
| nbtstat | Просмотр статистики текущих сетевых NBT-соединений |
| net | Управление сетью |

2. Перечислить все ***уровни модели OSI/ISO***. Описать назначение каждого уровня.

**Физический уровень** — это самый нижний из семи уровней, отвечающий за передачу необработанных данных в двоичном виде между устройствами по физической среде. Он определяет свойства среды передачи данных (коаксиальный кабель, витая пара, оптоволоконный канал и т.п.) и способы ее соединения с сетевыми адаптерами:технические характеристики кабелей (сопротивление, емкость, изоляция и т.д.), перечень допустимых разъемов, способы обработки сигнала и т.п.

**Канальный уровень**. На канальном уровне модели рассматривается два подуровня: подуровень управления доступом к среде передачи данных и подуровень управления логическим каналом. Управление доступом к среде передачи данных MAC (media access control) определяет методы совместного использования сетевыми адаптерами среды передачи данных. Подуровень управления логической связью LLC (logical link control): осуществляет:

* управление передачей данных;
* обеспечивает проверку и правильность передачи информации по соединению

Основное назначение процедур канального уровня подготовить блок данных (обычно называемый **кадром**) для следующего сетевого уровня.

Здесь следует отметить два момента: 1) начиная с подуровня управления логической связью и выше протоколы никак не зависят от среды передачи данных; 2) для **организации локальной сети** достаточно только физического и канального уровней, но такая сеть не будет масштабируемой (не сможет расширяться), т.к. имеет ограниченные возможности адресации и не имеет функций маршрутизации.

**Сетевой уровень**. Сетевой уровень определяет методы адресации и маршрутизации компьютеров в сети. В отличие от канального, уровня сетевой уровень определяет **единый метод адресации для всех компьютеров в сети** не зависимого от способа передачи данных. На этом уровне определяются способы соединения компьютерных сетей. Результатом процедур сетевого уровня является **пакет**, который обрабатывается процедурами транспортного уровня.

**Транспортный уровень**. Основным назначением процедур транспортного уровня является подготовка и доставка пакетов данных между конечными точками без ошибок и в правильной последовательности. Процедуры транспортного уровня формируют файлы для сеансового уровня из пакетов, полученных от сетевого уровня.

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**. Датаграмма (datagram) — это тоже часть пакета, но ее нельзя путать с сегментом.

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. При потере датаграмм или сегментов получаются «битые» куски данных, которые не получится корректно обработать.

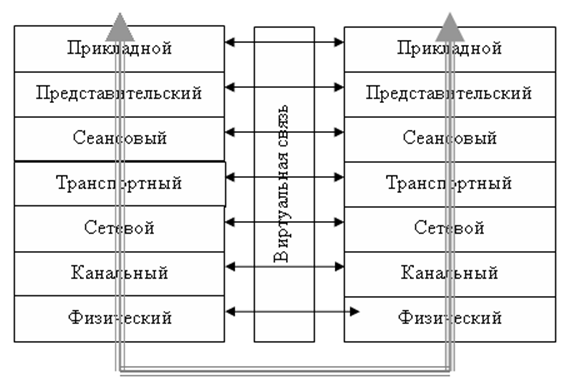
**Сеансовый уровень**. Сеансовый уровень определяет способы установки, поддержки и разрыва соединений (называемых сеансами) двух приложений, работающих в сети.

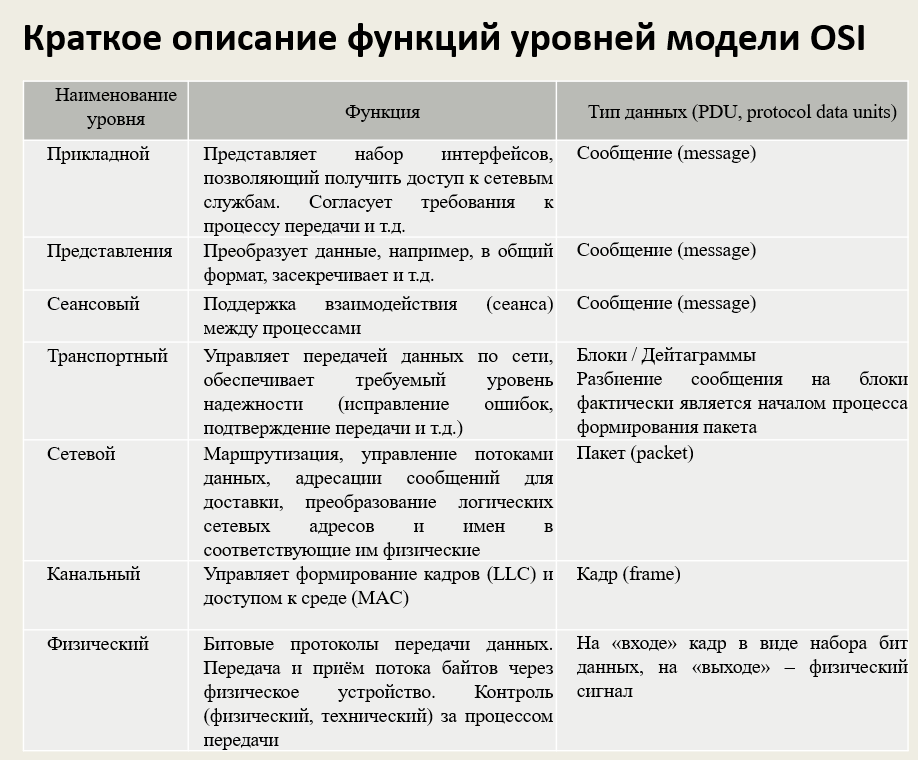
Следует отметить, что сеансовый уровень - это точка взаимодействия программ и компьютерной сети.

**Представительский уровень**. На представительский уровне определяется формат данных, используемых приложениями. Процедуры этого уровня описывают способы шифрования, сжатия и преобразования наборов символов данных.

**Прикладной уровень**. Основное назначения уровня: определить способы взаимодействия пользователей с системой (определить интерфейс). Что, куда и как отправится.

Данные проходят от прикладного уровня одной системы до прикладного уровня другой через все нижние уровни системы. Причем по мере своего движения от отправителя к получателю (из одной системы в другую) на каждом уровне данные подвергаются необходимому преобразованию (в соответствии с протоколами модели): при движении от прикладного уровня к физическому данные преобразовываются в формат, позволяющий передать данные по физическому каналу; при движении от физического уровня до прикладного происходит обратное преобразование данных.





из лк миронова

3. Поясните понятие ***сетевой протокол***.

Согласованный набор протоколов разных уровней, достаточный для организации межсетевого взаимодействия, называется **стеком протоколов**.

В общем случае можно выделить **три укрупненных уровня протоколов**, характерных в той или иной степени для любых стеков:

**−сетевые**;

−транспортные;

−прикладные.

**Сетевые протоколы** предоставляют следующие услуги: адресацию и маршрутизацию информации, проверку на наличие ошибок, запрос повторной передачи и установление правил взаимодействия в конкретной сетевой среде.

* DDP (Datagram Delivery Protocol – Протокол доставки дейтаграмм). Протокол передачи данных Apple, используемый в Apple Talk.
* IP (Internet Protocol – Протокол Internet). Протокол стека TCP/IP, обеспечивающий адресную информацию и информацию о маршрутизации.
* IPX (Internetwork Packet eXchange – Межсетевой обмен пакетами) в NWLink. Протокол Novel NetWare, используемый для маршрутизации и направления пакетов.
* NetBEUI (NetBIOS Extended User Interface – расширенный пользовательский интерфейс базовой сетевой системы ввода вывода). Разработанный совместно IBM и Microsoft, этот протокол обеспечивает транспортные услуги для NetBIOS.

4. Указать где в OSI/ISO проходит ***граница между аппаратным и программным обеспечением***.

На канальном уровне между её подуровнями. На канальном уровне модели рассматривается два подуровня: подуровень управления доступом к среде передачи данных и подуровень управления логическим каналом. Управление доступом к среде передачи данных MAC (media access control) определяет методы совместного использования сетевыми адаптерами среды передачи данных. Подуровень управления логической связью LLC (logical link control): осуществляет:

* управление передачей данных;
* обеспечивает проверку и правильность передачи информации по соединению

5. Определить понятие ***CSMA/CD***.

**Множественный доступ с прослушиванием несущей и разрешением коллизий** (Carrier Sense Multiple Access with Collision Detection – CSMA/CD)

1. Если рабочая станция хочет воспользоваться сетью для передачи данных, она сначала должна **проверить состояние канала**: начинать передачу станция может, **если канал свободен**.
2. В процессе передачи станция **продолжает прослушивание сети** для обнаружения возможных конфликтов.
3. Если возникает конфликт из-за того, что два узла попытаются занять канал, то обнаружившая конфликт интерфейсная плата, выдает в сеть специальный сигнал, и обе станции **одновременно прекращают передачу**.
4. Принимающая станция отбрасывает частично принятое сообщение, а все рабочие станции, желающие передать сообщение, в течение некоторого, случайно выбранного промежутка времени выжидают, прежде чем начать заново передачу сообщение.

Отметим, что все сетевые интерфейсные платы запрограммированы на разные псевдослучайные промежутки времени.

Если конфликт возникнет во время повторной передачи сообщения, этот промежуток времени будет увеличен.

Данная процедура будет повторяться до тех пор, пока сеть не выйдет из состояния коллизий.

Ещё есть **Множественный доступ с прослушиванием несущей и предотвращением коллизий** (Carrier Sense Multiple Access with Collision Avoidance – CSMA/CA). В сущности, CSMA/CA отличается от CSMA/CD тем, что узлы сообщают о намерении передать данные по сети до фактической их передачи. Узлы постоянно «прослушивают» объявления других узлов и при обнаружении объявления отменяют передачу своих данных.

6. Как называется ***программное обеспечение реализующий подуровень LLC канального уровня***.

Спецификации IEEE делят канальный уровень на два подуровня:

-**LLC** (Logical Link Control) управление логическим каналом осуществляет логический контроль связи. Подуровень LLC обеспечивает обслуживание сетевого уровня и связан с передачей и приемом пользовательских сообщений.

-**MAC** (Media Assess Control) контроль доступа к среде. Подуровень MAC регулирует доступ к разделяемой физической среде (передача маркера, обнаружение коллизий или столкновений) и управляет доступом к каналу связи. Подуровень LLC находится выше подуровня МАC.

7. Что определяет ***спецификация NDIS***?

**Спецификация NDIS** (Network Driver Interface Specification) определяет интерфейс между сетевыми драйверами и операционной системой Windows. Она предоставляет стандартный способ для сетевых драйверов взаимодействовать с операционной системой и сетевым оборудованием.

8. Свойства ***ненадежных протоколов***. Примеры ненадежных и ***надежных*** протоколов

Ненадежные и надежные протоколы различаются по способу обработки передачи данных и обеспечению их целостности. Вот основные свойства каждого типа:

### **Ненадежные протоколы**

**Свойства:**

1. **Отсутствие подтверждений доставки**: Ненадежные протоколы не обеспечивают подтверждения получения данных. Они не гарантируют, что данные были успешно доставлены получателю.
2. **Отсутствие контроля ошибок**: Эти протоколы не включают механизмы для обнаружения и исправления ошибок, которые могут возникнуть во время передачи данных.
3. **Отсутствие контроля потока**: Ненадежные протоколы не управляют скоростью передачи данных, что может привести к перегрузке сети или потере данных.
4. **Нет гарантии порядка доставки**: Они не гарантируют, что пакеты данных будут доставлены в том порядке, в котором они были отправлены.

**Примеры ненадежных протоколов:**

* **UDP (User Datagram Protocol)**: Обеспечивает передачу данных без подтверждения их получения и не гарантирует порядок доставки. Используется в приложениях, где важна скорость передачи данных и можно компенсировать потерю данных на уровне приложения (например, в потоковом видео или VoIP).
* **IP (Internet Protocol)**: Является базовым протоколом для передачи данных в сети, который не гарантирует доставку пакетов. Он работает на сетевом уровне и используется в связке с протоколами, такими как TCP, для создания надежных соединений.

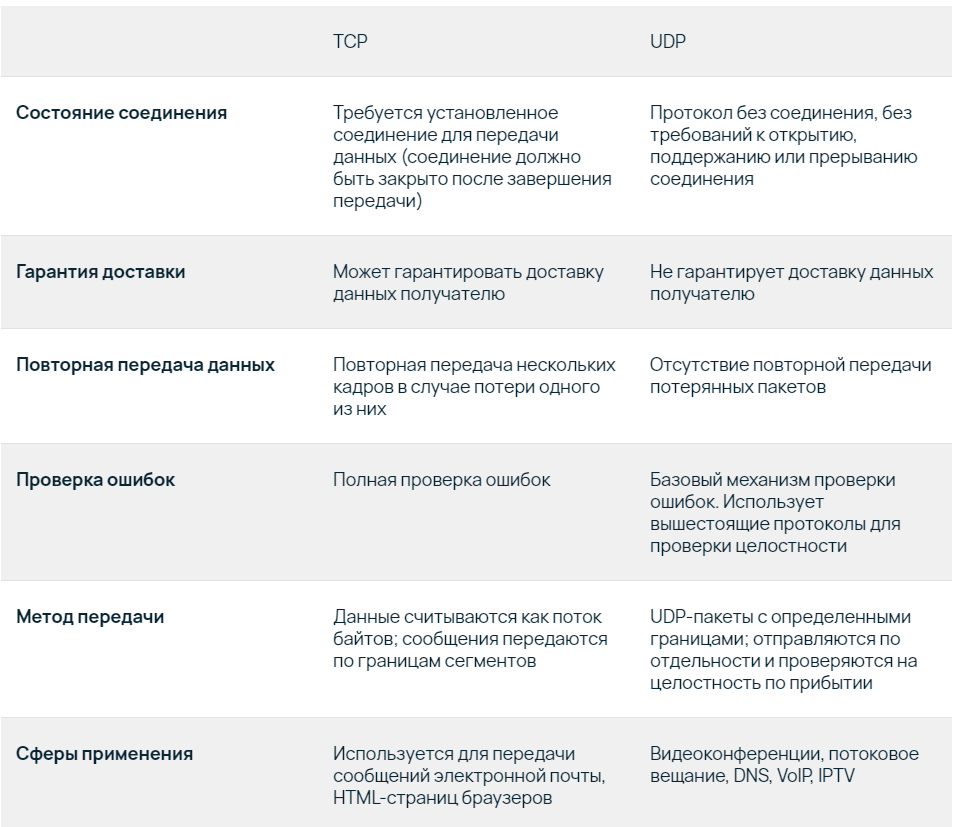
### **Надежные протоколы**

**Свойства:**

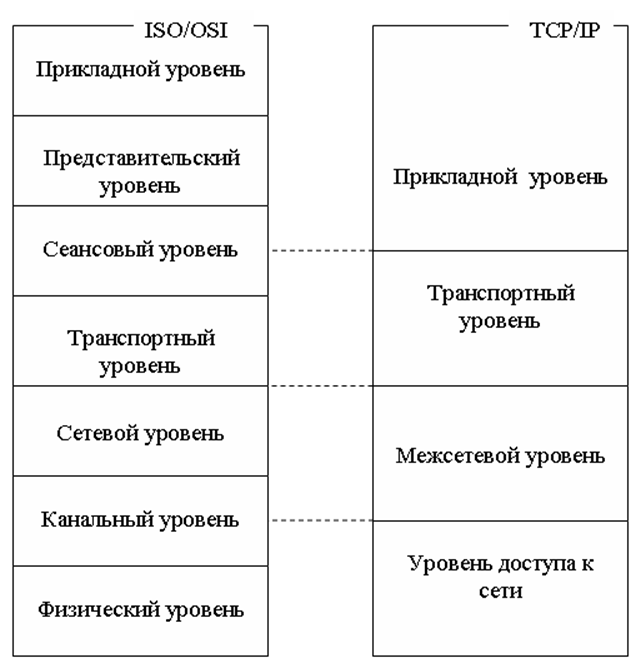
1. **Подтверждение доставки**: Надежные протоколы используют механизмы подтверждения получения данных, чтобы удостовериться, что данные были доставлены успешно.
2. **Контроль ошибок**: Включают функции для обнаружения и исправления ошибок, которые могут возникнуть в процессе передачи данных.
3. **Контроль потока**: Управляют скоростью передачи данных, чтобы предотвратить перегрузку сети и обеспечить эффективное использование ресурсов.
4. **Гарантия порядка доставки**: Гарантируют, что данные будут доставлены в том порядке, в котором они были отправлены.

**Примеры надежных протоколов:**

* **TCP (Transmission Control Protocol)**: Обеспечивает надежную передачу данных, используя подтверждения получения, контроль ошибок и управление потоком. Гарантирует, что данные будут доставлены в правильном порядке и без потерь. Используется для большинства приложений, требующих надежной передачи данных, таких как веб-браузеры и электронная почта.
* **FTP (File Transfer Protocol).** Протокол передачи файлов. Его использовали ещё в 1971 году — задолго до появления протокола IP. На текущий момент этим протоколом пользуются при удалённом доступе к хостингам. FTP является надёжным протоколом, поэтому гарантирует передачу данных. Этот протокол работает по принципу клиент-серверной архитектуры. Пользователь проходит аутентификацию (хотя в отдельных случаях может подключаться анонимно) и получает доступ к файловой системе сервера.



9. Перечислить все уровни ***модели TCP/IP***. Описать назначение каждого уровня. Привести примеры протоколов каждого уровня.



**Уровень доступа к сети** — на этом уровне данные передаются физически между устройствами через разнообразные сетевые системы. Этот слой также включает в себя методы управления доступом к физическим носителям данных и определяет правила физического адресации устройств в рамках локальной сети, используя такие методы, как MAC-адреса. Примеры протоколов канального уровня — [Ethernet](https://ru.wikipedia.org/wiki/Ethernet), [IEEE 802.11](https://ru.wikipedia.org/wiki/IEEE_802.11) [WLAN](https://ru.wikipedia.org/wiki/%D0%91%D0%B5%D1%81%D0%BF%D1%80%D0%BE%D0%B2%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_%D0%BB%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), [SLIP](https://ru.wikipedia.org/wiki/SLIP), [Token Ring](https://ru.wikipedia.org/wiki/Token_Ring), [ATM](https://ru.wikipedia.org/wiki/ATM) и [MPLS](https://ru.wikipedia.org/wiki/MPLS), Bluetooth.

**Межсетевой уровень** — занимается адресацией и маршрутизацией информации между сетями. Здесь ключевым является протокол IP (Internet Protocol), который назначает уникальные IP-адреса каждому устройству и обеспечивает доставку пакетов данных от отправителя к получателю через разнообразные сети. Также: ARP, ICMP.

**Транспортный уровень** — гарантирует достоверную передачу информации между устройствами в сети. Среди его протоколов можно выделить TCP (протокол управления передачей) и UDP (протокол пользовательских дейтаграмм). TCP гарантирует точность и исправление ошибок при передаче, тогда как UDP предлагает более быструю, но менее гарантированную доставку данных.

**Прикладной уровень** — поддержание сеанса связи между хостами, преобразование передаваемых данных, работа с конечным пользователем и сетью. Он включает в себя множество протоколов, используемых в различных сетевых приложениях, таких как HTTP/HTTPS для взаимодействия с веб-сайтами, FTP для передачи файлов, SMTP для отправки электронной почты и многие другие.

10. Поясните понятия ***хост***, ***адрес хоста***, ***имя хоста***.

Все устройства имеющие сетевой адаптер называются **хостами** (host).

**Адрес хоста** — это уникальный идентификатор устройства в сети, который используется для его адресации и передачи данных. Существуют несколько типов адресов хоста в зависимости от уровня сети.

**Имя хоста (hostname)** — это удобочитаемое имя, которое присваивается устройству для его идентификации в сети. В отличие от IP-адреса, имя хоста предназначено для упрощения взаимодействия между людьми и не является числовым.

11. Какая ***организация поддерживает сетевые*** протоколы Internet. Как называются ***документы, описывающие эти протоколы***.

Сетевые протоколы Интернета поддерживаются организацией под названием **IETF** (Internet Engineering Task Force). Это международное сообщество инженеров, разработчиков и исследователей, которое занимается разработкой и поддержкой стандартов, используемых в Интернете.

### **Основные функции IETF:**

* Разработка новых стандартов и протоколов для Интернета.
* Обновление и улучшение существующих протоколов.
* Обеспечение совместимости различных сетевых технологий и устройств.

### **Документы, описывающие эти протоколы, называются RFC (Request for Comments).**

**RFC** — это официальные документы, которые содержат описание стандартов и спецификаций интернет-протоколов, а также рекомендации по их использованию. Эти документы доступны для всех и публикуются в открытом доступе на сайте IETF.

12. Что такое ***МАС-адрес***? Структура ***Ethernet МАС-адреса***.

**MAC-адрес** — это уникальный номер любого устройства, имеющего выход в сеть. В отличие от динамических IP-адресов они статичны, что полезно для диагностики сетевых проблем.

Первые 3 октета (в порядке их передачи по сети; старшие 3 октета, если рассматривать их в традиционной бит-реверсной шестнадцатиричной записи MAC-адресов) содержат 24-битный уникальный идентификатор организации (OUI-Organization Unique Identifier), который производитель получает в IEEE. При этом используются только младшие 22 разряда (бита), 2 старшие имеют специальное назначение:



* первый бит указывает, для одиночного (0) или группового (1) адресата предназначен кадр;
* следующий бит указывает, является ли MAC-адрес глобально (0) или локально (1) администрируемым.

Следующие три октета выбираются изготовителем для каждого экземпляра устройства.

В дверь постучали два раза.  
- Наконец обычный человек, - догадался Штирлиц.  
- Не догадался, - догадалась четверть осьминога.

13. Как ***посмотреть MAC-адрес сетевой карты*** на компьютере?

1. Открыть командную строку [cmd]
2. Ввести команду [ipconfig/all]
3. После ввода команды, вы найдете информацию о физическом адресе (MAC) сетевой карты в Ethernet разделе.

14. Основное назначение ***межсетевого уровня***.

Основное назначение межсетевого уровня (уровня 3 в модели OSI, также называемого сетевым уровнем) *—* **обеспечение передачи данных между различными сетями и маршрутизация пакетов от отправителя к получателю через сложные сети, состоящие из множества узлов и маршрутов.**

Функции межсетевого уровня включают:

1. **Маршрутизация:** Определение оптимального пути для передачи пакетов данных между различными сетями на основе IP-адресов.
2. **Логическая адресация:** Назначение уникальных IP-адресов для устройств, что позволяет идентифицировать конечные устройства в сети.
3. **Фрагментация и сборка пакетов:** Разделение больших пакетов на меньшие фрагменты для передачи через сети с различными MTU (Maximum Transmission Unit) и их сборка на стороне получателя.
4. **Контроль ошибок:** Обнаружение ошибок, возникающих при передаче данных, и обеспечение их корректного восстановления.
5. **Перенаправление и пересылка:** Перенаправление трафика через промежуточные маршрутизаторы и другие устройства, которые принимают решения о пересылке пакетов дальше в зависимости от таблиц маршрутизации.

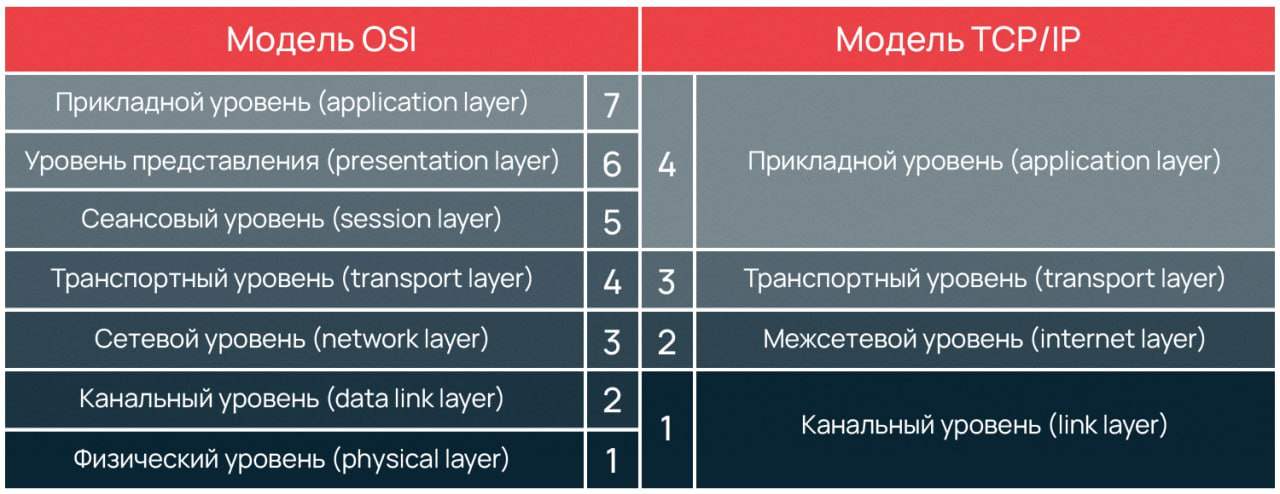
В модели TCP/IP **межсетевой уровень** (Network Layer) выполняет примерно те же функции, что и сетевой уровень в модели OSI, но с более фокусированным набором задач. В контексте TCP/IP, основная цель межсетевого уровня заключается в **маршрутизации и доставке пакетов данных** через различные сети с использованием протокола IP (Internet Protocol).

**Основные функции межсетевого уровня в модели TCP/IP:**

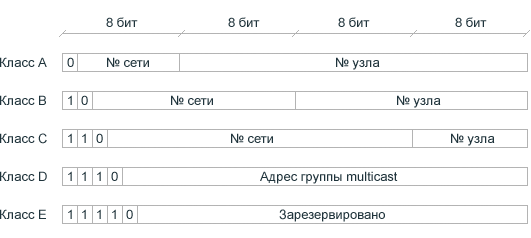
* **Маршрутизация пакетов**.
* **Адресация**.
* **Инкапсуляция и декapsulation**: Межсетевой уровень инкапсулирует данные из транспортного уровня (TCP или UDP) в IP-пакеты и передает их на канальный уровень для дальнейшей передачи по сети. При получении пакетов он выполняет обратный процесс — декapsulation.
* **Фрагментация и сборка пакетов**.
* **Обработка ошибок и TTL (Time to Live)**: Межсетевой уровень использует поле TTL в IP-заголовке, которое ограничивает время жизни пакета в сети. Это предотвращает бесконечное движение пакетов по сети в случае ошибок в маршрутизации.

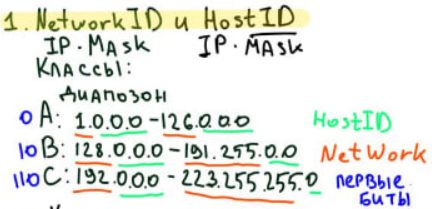
**Пример протоколов межсетевого уровня в TCP/IP:**

* **IP (Internet Protocol)** — основа межсетевого уровня, которая отвечает за адресацию и маршрутизацию пакетов.
* **ICMP (Internet Control Message Protocol)** — используется для диагностики сетевых проблем (например, команды ping и traceroute).



15. Структура ***IP-адреса***.





16. Типы ***IP-адресации***. ***Классы адресов Internet***.



1. **Частные (локальные) IP-адреса** (Private IP):  
   - Используются внутри локальных сетей для идентификации устройств. Эти адреса не маршрутизируются через интернет.  
   - Устройства с частными IP-адресами — это компьютеры, планшеты, смартфоны, устройства Bluetooth, смарт-телевизоры и принтеры.   
   - Диапазоны частных адресов (IPv4):

* 10.0.0.0 — 10.255.255.255
* 172.16.0.0 — 172.31.255.255
* 192.168.0.0 — 192.168.255.255

1. **Публичные (глобальные) IP-адреса** (Public IP):  
   - Это глобально уникальные адреса, которые могут быть использованы для идентификации устройств в интернете.  
   - Эти адреса выдаются интернет-провайдерами и могут быть видны за пределами локальной сети.
2. **Статические IP-адреса** (Static IP):  
   - Назначаются вручную и не изменяются. Обычно используются для серверов, маршрутизаторов и других устройств, которые требуют постоянного адреса.  
   - Статический IP-адрес гарантирует, что все веб-сайты и адреса электронной почты, связанные с определенным веб-сервером, всегда будут иметь согласованный IP-адрес, чтобы к ним можно было подключиться через Интернет.
3. **Динамические IP-адреса** (Dynamic IP):  
   - Назначаются автоматически с помощью DHCP-сервера и могут изменяться при каждом подключении устройства к сети. Этот тип используется для большинства пользователей интернета.
4. **IPv4 и IPv6 адреса**:  
   - **IPv4** (Internet Protocol version 4) — это 32-битный адрес (например, 192.168.0.1), который широко используется, но его диапазон ограничен.  
   - **IPv6** (Internet Protocol version 6) — это 128-битный адрес (например, 2001:0db8:85a3:0000:0000:8a2e:0370:7334), который был введен для расширения адресного пространства из-за исчерпания IPv4.
5. **Одноадресная адресация** (Unicast):  
   - Уникальный IP-адрес, используемый для передачи данных от одного источника к одному получателю.
6. **Многоадресная адресация** (Multicast):  
   - Используется для отправки данных от одного источника сразу нескольким получателям в пределах группы устройств.
7. **Широковещательная адресация** (Broadcast):  
   - Передача данных от одного устройства ко всем устройствам в сети.

**Основные классы IPv4:**

1. **Класс A**:
   * Диапазон: от 1.0.0.0 до 126.255.255.255
   * Первый бит: всегда 0
   * Количество сетей: 128 сетей (с 0 по 127, но 127 зарезервирован для loopback)
   * Количество хостов в сети: до 16,777,214 (2³² − 2)
   * Использование: для очень крупных сетей, например, для интернет-провайдеров и крупных организаций.
   * Пример: 10.0.0.1
2. **Класс B**:
   * Диапазон: от 128.0.0.0 до 191.255.255.255
   * Первые два бита: 10
   * Количество сетей: 16,384 сетей
   * Количество хостов в сети: до 65,534 (2¹⁶ − 2)
   * Использование: для средних и крупных сетей, таких как университеты или компании.
   * Пример: 172.16.0.1
3. **Класс C**:
   * Диапазон: от 192.0.0.0 до 223.255.255.255
   * Первые три бита: 110
   * Количество сетей: 2,097,152 сетей
   * Количество хостов в сети: до 254 (2⁸ − 2)
   * Использование: для небольших сетей, таких как локальные сети (LAN).
   * Пример: 192.168.0.1
4. **Класс D** (Многоадресный или Multicast):
   * Диапазон: от 224.0.0.0 до 239.255.255.255
   * Первые четыре бита: 1110
   * Использование: для многоадресной передачи данных. Этот класс не используется для адресации хостов, а применяется для отправки данных нескольким устройствам одновременно.
   * Пример: 224.0.0.1
5. **Класс E** (Зарезервирован):
   * Диапазон: от 240.0.0.0 до 255.255.255.255
   * Первые четыре бита: 1111
   * Использование: зарезервирован для экспериментальных целей и не применяется в обычной сети.

17. Поясните понятия ***публичный IP-адрес*** и ***частный IP-адрес***.

Термины «публичный» и «частный» относятся к сетевому местоположению — то есть частный IP-адрес используется внутри сети, а общедоступный — вне сети.

**Частные IP-адреса**

Каждое устройство, подключенное к домашней сети или частной сети, имеет частный IP-адрес. Частные IP-адреса не подключены к Интернету и используются только во внутренней сети. Устройства с частными IP-адресами — это компьютеры, планшеты, смартфоны, устройства Bluetooth, смарт-телевизоры и принтеры. С ростом популярности продуктов Интернета вещей использование частной IP-адресации, вероятно, будет продолжать расти.

**Публичные IP-адреса**

Интернет-провайдер назначает эти адреса, которые позволяют маршрутизатору взаимодействовать с Интернетом или внешней сетью. Общедоступные IP-адреса охватывают всю сеть, что означает, что несколько устройств, использующих одно и то же подключение к Интернету, также будут использовать один и тот же общедоступный IP-адрес.

18. Как посмотреть ***IP-адрес компьютера***.

1. **Через командную строку**:

ipconfig

В выводе команды найдите строку **"IPv4-адрес"** в разделе адаптера, используемого для подключения к сети (Ethernet или беспроводной адаптер).

1. **Через сетевые настройки**:
   * Нажмите на значок сети в трее, выберите **"Центр управления сетями и общим доступом"**.
   * Нажмите на название сети, к которой подключены, и выберите **"Сведения"**. Там будет указан ваш IPv4-адрес.

19. Как протестировать ***IP-соединение в локальной сети*** ?

**1. Команда ping:  
 ping** — это одна из наиболее простых и популярных команд для проверки доступности устройства в сети. Она отправляет ICMP-запросы (эхо-запросы) к другому устройству и ожидает ответа.

ping <IP-адрес>

**2. Команда tracert** Эта команда используется для отслеживания маршрута данных до удаленного узла. Она поможет выявить, на каком этапе теряется соединение в случае проблем.

tracert <IP-адрес>

**3. Команда arp:** Команда arp используется для просмотра таблицы ARP, которая отображает соответствие IP-адресов и MAC-адресов в локальной сети.

arp -a

Вы увидите список всех устройств в локальной сети с их IP и MAC-адресами, что поможет убедиться, что устройство присутствует в сети.

**4. Команда netstat:** С помощью команды netstat можно просматривать активные подключения, открытые порты и другие сетевые параметры. Это полезно для диагностики текущих сетевых подключений.

Для более подробной информации используйте флаги, например, для просмотра всех активных подключений:

netstat -a

20. Как получить ***перечень сетевых узлов*** между двумя хостами?

Чтобы получить перечень сетевых узлов (маршрутизаторов, шлюзов и других устройств) между двумя хостами, можно использовать команду tracert в Windows или traceroute в Linux и macOS. Эти команды отображают маршрут, по которому проходят пакеты от вашего устройства до целевого узла, перечисляя каждый промежуточный узел, через который проходит запрос.

**tracert <IP-адрес или доменное имя>**

21. Перечислите ***параметры настройки TCP/IP***.

При настройке TCP/IP (Transmission Control Protocol/Internet Protocol) для подключения устройства к сети, необходимо указать несколько ключевых параметров. Эти параметры обеспечивают корректную работу в сети и доступ к интернету. Вот основные параметры, которые нужно учитывать:

**1. IP-адрес (IP Address):**

**2. Маска подсети (Subnet Mask):** Определяет, какая часть IP-адреса относится к сети, а какая — к хосту.

**3. Шлюз по умолчанию (Default Gateway):** Устройство (обычно маршрутизатор), которое предоставляет доступ к внешним сетям, включая интернет.

**4. DNS-сервер (DNS Server):** Сервер, который преобразует доменные имена в IP-адреса.

**5. WINS-сервер (WINS Server) (редко используется):** Применяется в сетях Microsoft для разрешения имен NetBIOS. Обычно не требуется в современных сетях.

**6. DHCP (Dynamic Host Configuration Protocol):** Протокол автоматической настройки сетевых параметров, таких как IP-адрес, маска подсети, шлюз и DNS.

**7. MTU (Maximum Transmission Unit):** Максимальный размер пакета данных, который может быть передан по сети. Обычно это значение по умолчанию составляет 1500 байт, но его можно настроить для улучшения работы сети.

**8. Скорость и дуплекс (Speed and Duplex):** Настройка сетевого адаптера, определяющая скорость соединения (например, 100 Мбит/с или 1 Гбит/с) и режим дуплекса (полный или полудуплекс). Полный дуплекс позволяет одновременную передачу и прием данных.

**9. Альтернативная конфигурация IP (Alternate IP Configuration):** Эта настройка используется в случае, если устройство не получает IP-адрес через DHCP. Она позволяет указать резервные параметры для использования, если DHCP недоступен.

**10. MAC-адрес (Media Access Control Address):** Уникальный аппаратный адрес сетевого адаптера, используемый для физической идентификации устройства на уровне канала передачи данных. Хотя он не является частью конфигурации TCP/IP, его иногда можно вручную изменить (MAC spoofing).

**11. Резервирование IP-адреса:** Настройка DHCP-сервера, которая позволяет выделять конкретный IP-адрес для определенного устройства на основе его MAC-адреса.

**Дополнительные параметры для IPv6:**

* Префикс подсети (Subnet Prefix): Аналог маски подсети в IPv6.
* Глобальный адрес (Global Address): Уникальный адрес для интернета в IPv6-сетях.
* Link-local адрес: Адрес, используемый для связи внутри одной локальной сети (начинается с fe80::).

**Основные параметры для ручной настройки TCP/IP:**

* IP-адрес.
* Маска подсети.
* Шлюз по умолчанию.
* DNS-сервер.

22. Поясните понятие ***маска подсети***.

**Маска подсети** — 32-битное число, служащее битовой маской для разделения сетевой части (адреса подсети) и части хоста IP-адреса.

**Основные функции маски подсети:**

Маска подсети помогает устройствам в одной сети правильно маршрутизировать пакеты данных.

Определяет, какие IP-адреса относятся к той же локальной сети, а какие находятся в других сетях и требуют отправки через маршрутизатор (шлюз по умолчанию).

**Структура маски подсети:**

Маска подсети выглядит так же, как IP-адрес, но состоит из последовательности единиц и нулей. Единицы (1) в маске подсети указывают на сетевую часть IP-адреса, а нули (0) — на часть хоста.

**Пример использования:**

Предположим, у нас есть IP-адрес 192.168.1.10 с маской подсети 255.255.255.0.

В этом случае 192.168.1 — это сетевая часть адреса, которая одинакова для всех устройств в данной подсети.

Последний октет .10 — это часть хоста, которая уникальна для каждого устройства в сети.

Таким образом, все устройства в сети с адресами 192.168.1.1 – 192.168.1.254 будут в одной подсети и могут общаться напрямую, не используя маршрутизатор. Для связи с устройствами вне этой подсети (например, с IP-адресом 192.168.2.5) данные должны быть переданы через шлюз.

23. Основные ***отличия между IPv4 и IPv6***.

| Характеристика | IPv4 | IPv6 |
| --- | --- | --- |
| Размер IP-адреса | 32 бита | 128 бит |
| Формат IP-адреса | Четыре десятичных числа, разделённых точками | Восемь групп по четыре шестнадцатеричных числа, разделённых двоеточиями |
| Пример IP-адреса | 192.168.1.1 | 2001:0db8:85a3:0000:0000:8a2e:0370:7334 |
| Количество адресов | Около 4.3 миллиарда адресов | Около 340 ундециллионов (3.4 × 10³⁸ адресов) |
| Адресное пространство | Ограниченное и исчерпаемое | Практически неограниченное, с большим запасом на будущее |
| Фрагментация | Выполняется отправителем или маршрутизатором | Выполняется только отправителем |
| Метод назначения адресов | Статическая или динамическая (DHCP) | Автоконфигурация (SLAAC) или DHCPv6 |
| Широковещание (Broadcast) | Есть поддержка широковещательной рассылки | Нет поддержки, заменено мультикастом и anycast-рассылкой |
| Мультикаст | Поддерживается | Поддерживается |
| Anycast | Ограниченная поддержка | Полная поддержка |
| NAT (Network Address Translation) | Широко используется из-за нехватки адресов | Не требуется (благодаря большому количеству адресов) |
| Безопасность (IPSec) | Необязательная поддержка | Обязательная поддержка встроена в протокол |
| Конфигурация | Требует ручной настройки или использования DHCP | Автоматическая конфигурация с помощью SLAAC или DHCPv6 |
| Поддержка мобильности | Ограниченная | Улучшена за счёт встроенных возможностей |
| Маршрутизация | Более сложная за счёт роста таблиц маршрутизации | Упрощена за счёт более структурированной адресации |
| Преобразование имен в адреса | Использует записи A (для IPv4) | Использует записи AAAA (для IPv6) |

24. Поясните понятие ***сетевой порт***. На каком уровне модели TCP/IP это понятие определено.

**Порт** (англ. port) — натуральное число (как правило, в диапазоне от 1 до 216-1=65 535), записываемое в заголовках протоколов транспортного уровня (например, TCP), которое используется для определения процесса-получателя пакета в пределах одного хоста.

Процесс, получающий или отправляющий данные с помощью Транспортного уровня, идентифицируется номером, который называется ***номером порта.*** Таким образом, адресат в сети TCP/IP полностью определяется тройкой: IP-адресом, номером порта и типом протокола транспортного уровня (UDP или TCP).

**Сетевой порт** — это логический идентификатор, который используется для указания конкретного процесса или службы, работающей на устройстве, и помогает направлять сетевые данные к правильному приложению. Сетевые порты позволяют различным приложениям на одном устройстве взаимодействовать с другими устройствами в сети через один IP-адрес.

Каждый сетевой порт связан с протоколом передачи данных, например, TCP (Transmission Control Protocol) или UDP (User Datagram Protocol).

Порты идентифицируются 16-битными номерами, что позволяет использовать порты в диапазоне от 0 до 65535.

**Уровень модели TCP/IP:**

Понятие сетевого порта относится к **транспортному уровню** модели TCP/IP. Это третий уровень в модели TCP/IP (иногда также называется моделью OSI, где это четвертый уровень). Транспортный уровень отвечает за передачу данных между хостами и обеспечивает надежную доставку данных, управляя сессиями и контролем потоков.

**Протоколы на транспортном уровне, использующие порты:**

TCP (Transmission Control Protocol): обеспечивает установление соединения и надежную передачу данных с гарантией доставки. Использует порты для идентификации приложений, отправляющих и принимающих данные.

UDP (User Datagram Protocol): протокол без установления соединения, который обеспечивает быструю передачу данных без гарантии доставки. Также использует порты для направления данных.

**Классификация портов:**

* Порты "известных служб" (Well-Known Ports): от 0 до 1023.
* Зарезервированы для системных или широко известных служб (например, HTTP на порту 80, SSH на порту 22).
* Регистриуемые порты (Registered Ports): от 1024 до 49151.
* Используются для регистрации новых сетевых служб и приложений.
* Динамические или частные порты (Dynamic/Private Ports): от 49152 до 65535.

25. Как ***классифицируются сетевые порты***.

Номера портов, используемые для идентификации прикладных процессов (в соответствии с документами IANA), делятся на три диапазона: хорошо известные номера портов (well-known port number), зарегистрированные номера портов (registered port number), динамически номера портов (dynamic port number).

| Хорошо известные номера портов | 0 – 1023 |
| --- | --- |
| Зарегистрированные номера портов | 1024 – 49151 |
| Динамические номера портов | 49152– 65535 |

Хорошо известные номера портов присваиваются базовым системным службам (core services), имеющие системные привилегии. Зарегистрированные номера портов присваиваются промышленным приложениям и процессам. Распределение некоторых хорошо известных и зарегистрированных номеров портов приведено в таблице 2.5.2. Динамические номера портов (их часто называют эфемерными портами) выделяются, как правило, прикладным процессам специализированной службой операционной системы. Некоторые системы TCP/IP применяют диапазон значений от 1024 до 5000 для назначения эфемерных номеров портов.

26. Как посмотреть какими программами заняты сетевые порты на компьютере?

netstat -a

27. Поясните понятие архитектура ***клиент/сервер***.

Распределенное приложение, имеющее архитектуру клиент-сервер, подразумевает наличие в своем составе два вида процессов: процессы-серверы и процессы-клиенты. Далее эти процессы будем назвать просто серверами и клиентами.

Следует отметить, что приведенное в предисловии определение архитектуры клиент-сервер несколько упрощено. Дело в том, что некоторые процессы распределенного приложения могут выступать клиентом для некоторых процессов-серверов и одновременно являться серверами других процессов-клиентов.

Инициатором обмена данными между клиентом и сервером всегда является клиент. Для этого клиент должен обладать информацией о месте нахождения сервера или иметь механизмы для его обнаружения. Способы связи между клиентом и сервером могут быть различными и в общем случае зависят от интерфейсов, поддерживаемых операционной средой, в которой работает распределенное приложение. Клиент должен быть тоже распознан сервером, для того, чтобы сервер, во-первых, мог его отличить от других клиентов, а во-вторых, чтобы смог обмениваться с клиентом данными. Если основная вычислительная нагрузка ложится на сервер, а клиент лишь обеспечивает интерфейсом пользователя с сервером, то такой клиент часто называют тонким.

28. Что такое ***сетевая служба***. Приведите примеры сетевых служб.

Сетевая служба — это программное обеспечение или набор программ, которые предоставляют определённые функции через компьютерную сеть. Эти службы позволяют взаимодействовать с устройствами, программами и пользователями, находящимися в разных местах, через интернет или локальные сети. Главная цель сетевой службы — предоставление удалённого доступа к ресурсам и данным.

Примеры сетевых служб:

Веб-сервисы (HTTP/HTTPS) — позволяют пользователям просматривать веб-сайты и взаимодействовать с веб-приложениями через браузеры. Например, просмотр сайта через браузер осуществляется с помощью веб-сервиса HTTP или HTTPS.

Электронная почта (SMTP, IMAP, POP3) — это сервисы для отправки, получения и хранения электронных писем.

DNS (Domain Name System) — служба преобразования доменных имён (например, google.com) в IP-адреса, которые компьютеры используют для связи.

FTP (File Transfer Protocol) — служба для передачи файлов между устройствами через сеть.

DHCP (Dynamic Host Configuration Protocol) — автоматически назначает IP-адреса устройствам в сети.

29. Поясните понятие ***интерфейс внутренней петли***.

Большинство реализаций TCP/IP поддерживает интерфейс внутренней петли (loopback interface), который позволяет двум прикладным процессам, находящимся на одном хосте, обмениваться данными посредством протокола TCP/IP. При этом, как обычно, формируются дейтаграммы, но они не покидают пределы одного хоста. Для интерфейса внутренней петли, как уже упоминалось выше, зарезервирована сеть 127.0.0.0. В соответствии с общепринятыми соглашениями, большинство операционных систем назначают для интерфейса внутренней петли адрес 127.0.0.1 и присваивают символическое имя localhost.

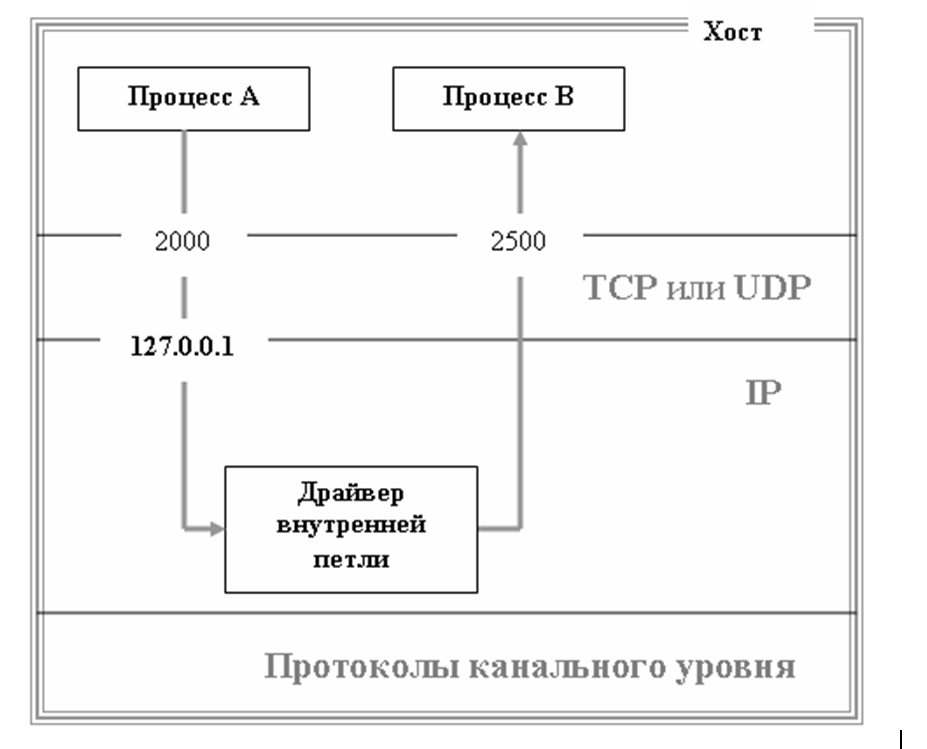


Рисунок 2.6.1. Схема работы интерфейса внутренней петли

На рисунке 2.6.1 приведена упрощенная схема обработки данных интерфейсом внутренней петли. Прикладной процесс A, изображенный на рисунке, используя номер порта 2000, отправляет данные процессу B. Указав в параметрах сокета процесса B сетевой адрес 127.0.0.1, процесс A обеспечил обработку посылаемых дейтаграмм на Межсетевом уровне драйвером внутренней петли, который направляет эти дейтаграммы во входную очередь модуля IP. IP- модуль, следуя обыкновенной логике своей работы, доставляет данные на Транспортный уровень. Далее протокол Транспортного уровня в соответствии с номером порта 2500 в заголовке сегмента (или пакета) направляет данные процессу B.

Следует обратить внимание на следующее: все данные пересылаемые по интерфейсу внутренней петли не только не покидают пределов хоста, но и не затрагивают никаких внешних механизмов за пределами стека TCP/IP.

30. Назначение сетевых служб ***DSN*** и ***DHCP***.

Служба DNS (Domain Name System) является одной из важнейших служб TCP/IP, само появление которой в 1980-х годах дало мощный толчок развитию TCP/IP и всемирной сети Internet. Дело в том, что DNS обеспечивает важную возможность преобразования символических доменных имен в соответствующие IP-адреса (разрешение имен). Например, для обращения к адресу серверу компании Microsoft, имеющему IP-адрес 207.46.230.229, можно обратиться, используя символическое имя microsoft.com. С одной стороны, это дает более наглядную нотацию, а с другой, появляется возможность не привязывать жестко получение услуг сервера к фиксированному адресу, который при реорганизации сети может измениться.

Службу DNS можно рассматривать, как распределенную иерархическую базу данных, основное назначение которой отвечать на два вида запросов: выдать IP-адрес по символическому имени хоста и наоборот – выдать символическое имя хоста по его IP-адресу. Облуживание этих запросов и поддержку базы данных в актуальном состоянии обеспечивают взаимодействующие глобально рассредоточенные в сети Internet серверы DNS. База данных имеет древовидную структуру, в корне которой нет ничего, а сразу под корнем находятся первичные сегменты (домены): .com, .edu, .gov, …, .ru, .by, .uk, … Наименование этих первичных доменов отражает деление базы данных DNS по отраслевому (домены, обозначенные трехбуквенным кодом) и национальному признакам (двухбуквенные домены в соответствии со стандартом ISO 3166). Доменом в терминологии DNS называется любое поддерево дерева базы данных DNS.

DHCP (Dynamic Host Configuration Protocol) – это сетевая служба (и протокол) Прикладного уровня TCP/IP, обеспечивающая выделение и доставку IP-адресов и сопутствующей конфигурационной информации (маска сети, адрес локального шлюза, адреса серверов DNS и т.п.) хостам. Применение DHCP дает возможность отказаться от фиксированных IP-адресов в зоне действия сервера DHCP. Описание протокола DHCP содержится в документах: RFC 1534, 2131, 2132, 2141.

Конструктивно служба DHCP состоит из трех модулей: сервера DHCP (DHCP Server), клиента DHCP (DHCP Client) и ретранслятора DHCP (DHCP Replay Agent) .

Логика работы протокола DHCP достаточно проста. При физическом подключении к сети, хост пытается подсоединиться к сети, используя для этого DHCP-клиент. Для обнаружения DHCP-сервера DHCP-клиент выдает в сеть широковещательный запрос (это процесс называется DHCP-поиском). Если в этом домене есть DHCP-сервер, то он окликается, посылая клиенту специальное сообщение, содержащее IP-адрес DHCP-сервера. Если доступны несколько DHCP-серверов, то, как правило, выбирается первый ответивший. Получив адрес сервера, клиент формирует запрос на выделение IP-адреса из пула адресов DHCP-сервера. В ответ на запрос, DHCP-сервер выделяет адрес клиенту на определенный период времени (аренда адреса). После получения IP-адреса TCP/IP-стек клиента начинает его использовать. Продолжительность аренды адреса устанавливается специально или по умолчанию (может колебаться от нескольких часов до нескольких недель). После истечения срока аренды DHCP-клиент пытается снова договорится с DHCP-сервером о продлении срока аренды или о выделении нового IP-адреса.

Ретранслятор DHCP используется в том случае, если на первоначальном этапе подключения к сети широковещательные запросы DHCP-клиента не могут быть доставлены (по разным причинам) DHCP-серверу. Ретранслятор в этом случае играет роль посредника между DHCP-клиентом и DHCP-сервером.

31. Организация, ведающая распределением ***IP-адресов***, поддержкой ***сетевых доменов Internet верхнего уровня***, ***регистрацией портов***.

Организацией, которая занимается распределением IP-адресов, поддержкой доменов верхнего уровня (TLD) и регистрацией портов, является ICANN — Internet Corporation for Assigned Names and Numbers (Корпорация по управлению доменными именами и IP-адресами).

ICANN координирует глобальное распределение IP-адресов через региональные интернет-регистратуры (RIR), такие как RIPE NCC (Европа), ARIN (Северная Америка), APNIC (Азия и Тихоокеанский регион), LACNIC (Латинская Америка) и AFRINIC (Африка).

ICANN контролирует систему доменных имен (DNS) и обеспечивает работу корневых серверов DNS. Она делегирует права на управление доменами верхнего уровня (например, .com, .org, .net, а также национальными доменами .ru, .uk и т.д.) различным регистраторам.

ICANN через IANA (Internet Assigned Numbers Authority) ведет регистрацию номеров портов, используемых для различных сетевых протоколов (например, порт 80 для HTTP, порт 443 для HTTPS). IANA также ведет реестр всех стандартных протоколов и их параметров.

Подразделение IANA (Internet Assigned Numbers Authority) которое является частью ICANN, отвечает непосредственно за:

* Назначение IP-адресов.
* Регистрацию доменов верхнего уровня.
* Регулирование и распределение номеров портов и идентификаторов протоколов для интернет-технологий.

32. Поясните понятие ***сетевой сокет***.

Сетевой сокет — комбинация IP адреса и номера порта, которые представляют собой способ адресации.

Адрес и порт обеспечивают нормальное взаимодействие большого количества приложений в рамках одной системы.

Один сокет не может использовать два приложения одновременно или два экземпляра одного приложения.

33. Назначение ***стандарта POSIX***.

API сокетов – это название программного интерфейса, предназначенного для обмена данными между процессами, находящимися на одном или на разных объединенных сетью компьютерах. Сокетом, кроме того, называют абстрактный объект, представляющий оконечную точку соединения. Впервые этот интерфейс появился в 1980-х годах в операционной системе BSD Unix (Berkeley Software Distribution), разработанной в университете Беркли (США, Калифорния) и описан в стандарте POSIX (Portable Operating System Interface for Unix).

Стандарт POSIX – это набор документов, описывающих интерфейсы между прикладной программой и операционной системой. Стандарт создан для обеспечения совместимости различных Unix-подобных операционных систем и переносимости исходных программ на уровне исходного кода. Официально стандарт определен как IEEE 1003, международное название стандарта ISO/IEC 9945.

34. ***Структура TCP/IP TCP-сервера***. Все функции и все параметры функций.

Шаги и функции TCP-сервера:

* Создание сокета (socket())
* Привязка сокета к адресу (bind())
* Ожидание подключений клиентов (listen())
* Принятие подключения (accept())
* Чтение и запись данных (recv() и send())
* Закрытие соединения (close())

### **1. Создание сокета (socket())**

Функция создает сокет для сетевого взаимодействия.

int socket(int domain, int type, int protocol);

* **domain** — указывает домен (семейство адресов):
  + AF\_INET — IPv4,
  + AF\_INET6 — IPv6.
* **type** — указывает тип сокета:
  + SOCK\_STREAM — для TCP (потоковое соединение),
  + SOCK\_DGRAM — для UDP (датаграммы).
* **protocol** — обычно 0, указывая на использование протокола по умолчанию (TCP для SOCK\_STREAM).

**Возвращает**: Дескриптор сокета (целое число). Если возникла ошибка, возвращает -1.

### **2. Привязка сокета к адресу (bind())**

Функция связывает сокет с локальным IP-адресом и портом.

int bind(int sockfd, const struct sockaddr \*addr, socklen\_t addrlen);

* **sockfd** — дескриптор сокета, полученный от socket().
* **addr** — указатель на структуру sockaddr, которая содержит адрес и порт.
  + Для IPv4 используется структура sockaddr\_in.  
    struct sockaddr\_in {

short sin\_family; // Семейство адресов, например AF\_INET

unsigned short sin\_port; // Порт в сетевом порядке байтов

struct in\_addr sin\_addr; // IP-адрес

char sin\_zero[8]; // Не используется (ноль)

};

* **addrlen** — размер структуры адреса (для IPv4 это sizeof(struct sockaddr\_in)).

**Возвращает**: 0 при успешном выполнении, -1 при ошибке.

### **3. Ожидание подключений (listen())**

Функция переводит сокет в режим пассивного прослушивания для приема входящих подключений.

int listen(int sockfd, int backlog);

* **sockfd** — дескриптор сокета, привязанного через bind().
* **backlog** — максимальное количество подключений, которые могут быть поставлены в очередь ожидания.

**Возвращает**: 0 при успешном выполнении, -1 при ошибке.

### **4. Принятие подключения (accept())**

Функция принимает входящее подключение и создает новый сокет для общения с клиентом.

int accept(int sockfd, struct sockaddr \*addr, socklen\_t \*addrlen);

* **sockfd** — дескриптор сокета, на котором ожидается подключение (сокет из listen()).
* **addr** — указатель на структуру, которая будет заполнена информацией об удаленном клиенте (его IP-адрес и порт).
* **addrlen** — указатель на переменную, которая содержит размер структуры addr. После вызова эта переменная будет содержать фактический размер.

**Возвращает**: Дескриптор нового сокета для общения с клиентом или -1 при ошибке.

### **5. Чтение и запись данных (recv() и send())**

Для чтения данных от клиента используется функция recv(), а для отправки данных клиенту — send().

#### **recv() — чтение данных из сокета:**

ssize\_t recv(int sockfd, void \*buf, size\_t len, int flags);

* **sockfd** — дескриптор сокета, который используется для приема данных.
* **buf** — указатель на буфер, куда будут записаны данные.
* **len** — размер буфера (максимальное количество байт, которое можно принять).
* **flags** — дополнительные флаги (обычно 0).

**Возвращает**: Количество байт, полученных от клиента, или -1 при ошибке.

#### **send() — отправка данных клиенту:**

ssize\_t send(int sockfd, const void \*buf, size\_t len, int flags);

* **sockfd** — дескриптор сокета, по которому будут отправлены данные.
* **buf** — указатель на буфер с данными для отправки.
* **len** — количество байт для отправки.
* **flags** — дополнительные флаги (обычно 0).

**Возвращает**: Количество байт, отправленных клиенту, или -1 при ошибке.

### **6. Закрытие соединения (close())**

Когда сервер завершает работу с сокетом (либо завершает сессию с клиентом), необходимо закрыть сокет, используя функцию close().

int close(int sockfd);

* **sockfd** — дескриптор сокета, который нужно закрыть.

**Возвращает**: 0 при успешном выполнении, -1 при ошибке.

35. ***Структура TCP/IP TCP-клиента***. Все функции и все параметры функций.

Основные шаги и функции TCP-клиента:

* Создание сокета (socket())
* Установка соединения с сервером (connect())
* Чтение и запись данных (recv() и send())
* Закрытие соединения (close())\

### **1. Создание сокета (socket())**

TCP-клиент, как и сервер, должен создать сокет для сетевого взаимодействия.

int socket(int domain, int type, int protocol);

* **domain** — определяет семейство адресов.
  + AF\_INET — для IPv4.
  + AF\_INET6 — для IPv6.
* **type** — указывает тип сокета.
  + SOCK\_STREAM — для TCP (потоковое соединение).
* **protocol** — обычно 0, что указывает на использование TCP по умолчанию.

**Возвращает**: Дескриптор сокета (целое число), или -1 при ошибке.

### **2. Установка соединения с сервером (connect())**

Клиент использует функцию connect() для установления TCP-соединения с удаленным сервером.

int connect(int sockfd, const struct sockaddr \*addr, socklen\_t addrlen);

* **sockfd** — дескриптор сокета, созданный с помощью socket().
* **addr** — указатель на структуру sockaddr, содержащую IP-адрес и порт сервера.

Для IPv4 используется структура sockaddr\_in:  
struct sockaddr\_in {

short sin\_family; // Семейство адресов (AF\_INET)

unsigned short sin\_port; // Порт (в сетевом порядке байтов)

struct in\_addr sin\_addr; // IP-адрес

char sin\_zero[8]; // Ноль (не используется)

};

* + Поле sin\_addr содержит IP-адрес сервера, а поле sin\_port — порт сервера.
* **addrlen** — размер структуры адреса (например, sizeof(struct sockaddr\_in) для IPv4).

**Возвращает**: 0 при успешном выполнении, -1 при ошибке.

### **3. Чтение и запись данных**

После установления соединения клиент может отправлять и получать данные с использованием функций send() и recv().

#### **Отправка данных (send())**

ssize\_t send(int sockfd, const void \*buf, size\_t len, int flags);

* **sockfd** — дескриптор сокета, который используется для отправки данных.
* **buf** — указатель на буфер с данными для отправки.
* **len** — размер данных в байтах, которые нужно отправить.
* **flags** — дополнительные флаги, обычно 0.

**Возвращает**: Количество байт, отправленных серверу, или -1 при ошибке.

#### **Прием данных (recv())**

ssize\_t recv(int sockfd, void \*buf, size\_t len, int flags);

* **sockfd** — дескриптор сокета для приема данных.
* **buf** — указатель на буфер, куда будут записаны данные.
* **len** — размер буфера (максимальное количество байт, которое можно принять).
* **flags** — дополнительные флаги, обычно 0.

**Возвращает**: Количество байт, полученных от сервера, или -1 при ошибке.

### **4. Закрытие соединения (close())**

После завершения работы клиент должен закрыть сокет с помощью функции close().

int close(int sockfd);

* **sockfd** — дескриптор сокета, который нужно закрыть.

**Возвращает**: 0 при успешном выполнении, -1 при ошибке.