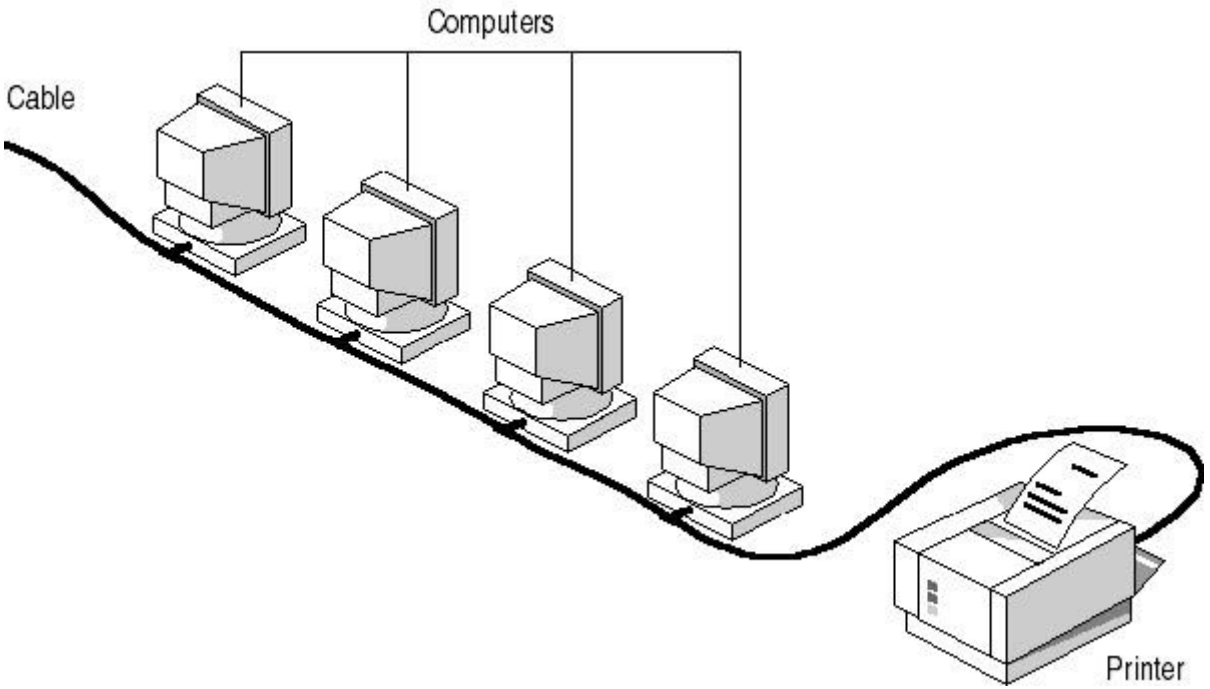
# Основы компьютерных сетей

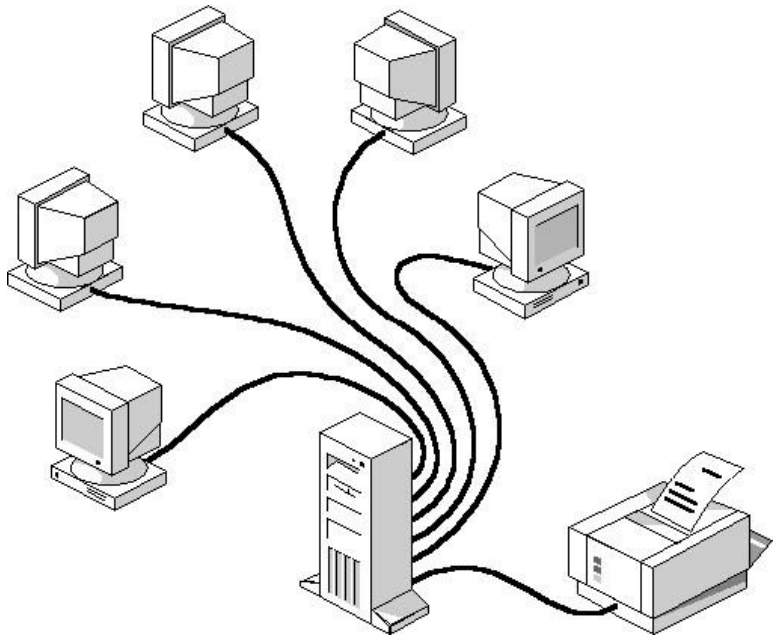
# Понятие компьютерной сети.

Сеть – группа соединенных компьютеров и других устройств



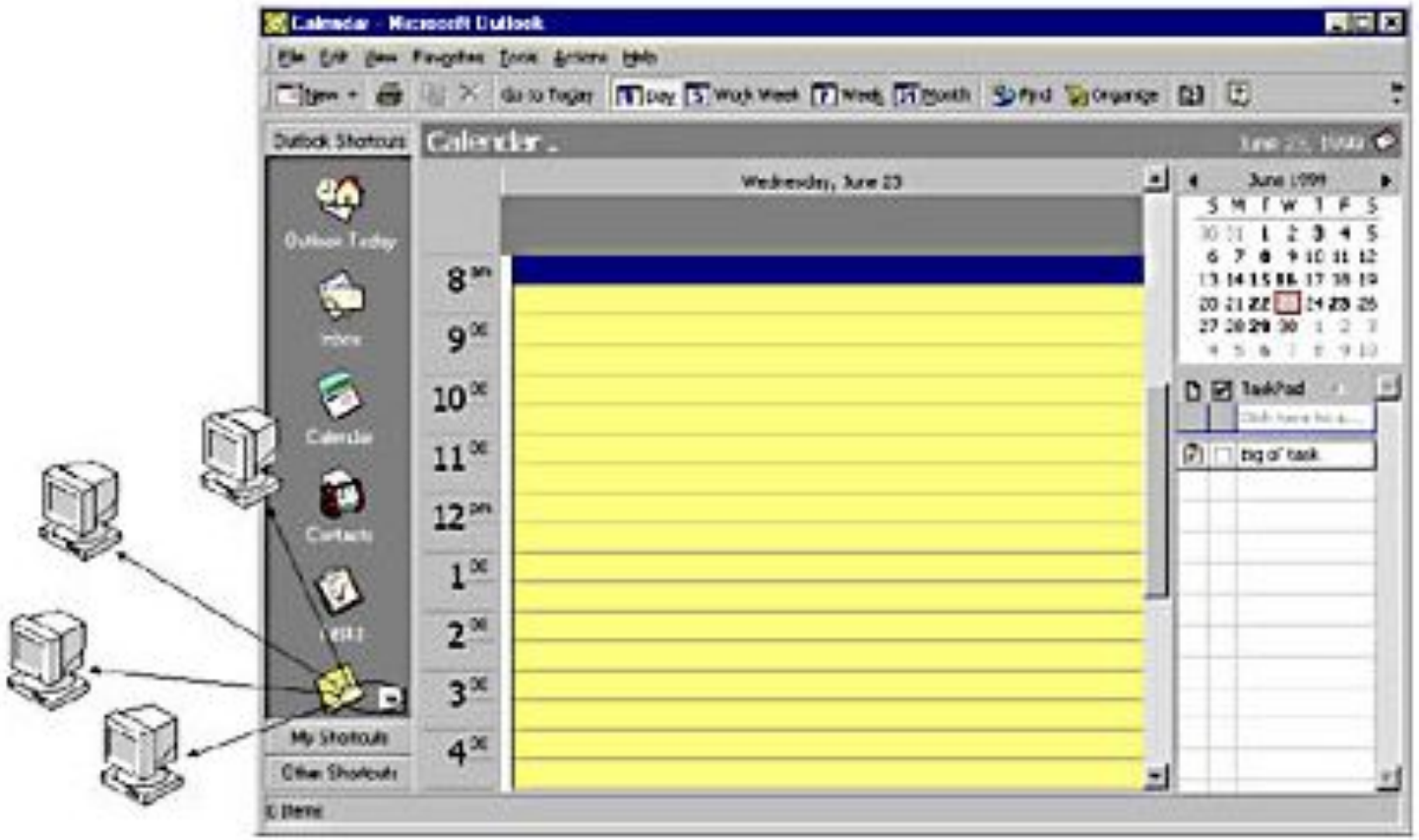
# Назначение сети

1. Совместное использование ресурсов
   - Периферийные устройства (например, принтер)
   - Данные
   - Прикладные программы



# 2. Связь в реальном режиме времени между пользователями сети (интерактивная связь)

Например, планирование совещания в Microsoft Outlook



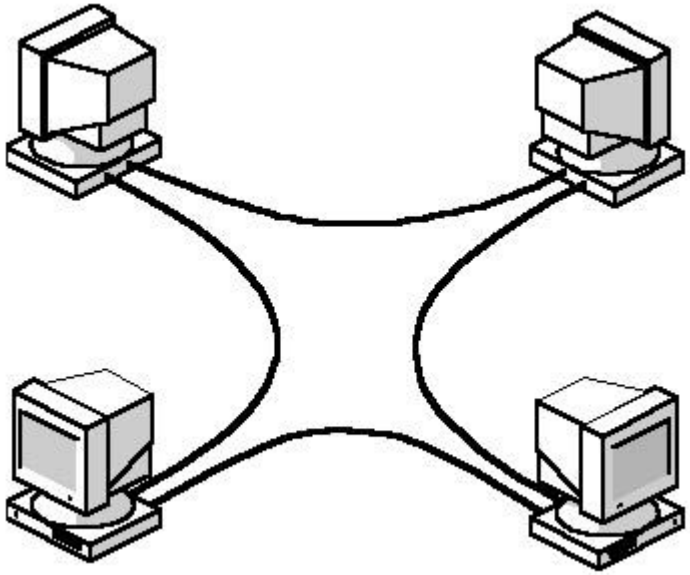
# ЛВС и ГВС (LAN and WAN)

Сети разделяют на две группы в зависимости от их размеров и назначения

1. ЛВС - локальная вычислительная сеть (LAN, local area network )

   ЛВС – основа для построения сети любого масштаба.

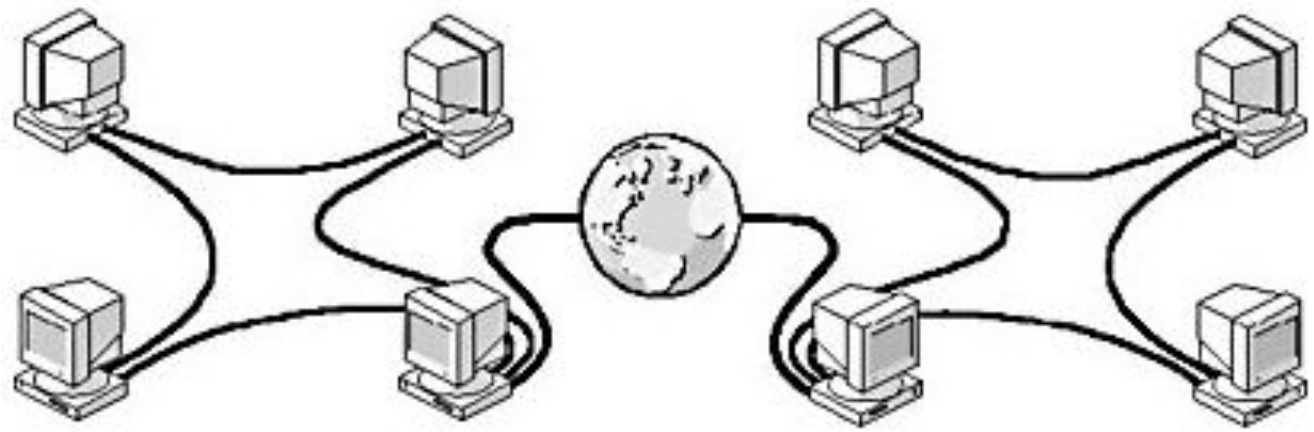
   Отличительная черта – все устройства, входящие в нее расположены на ограниченной территории.



2. ГВС - глобальная вычислительная сеть (WAN, wide area network )

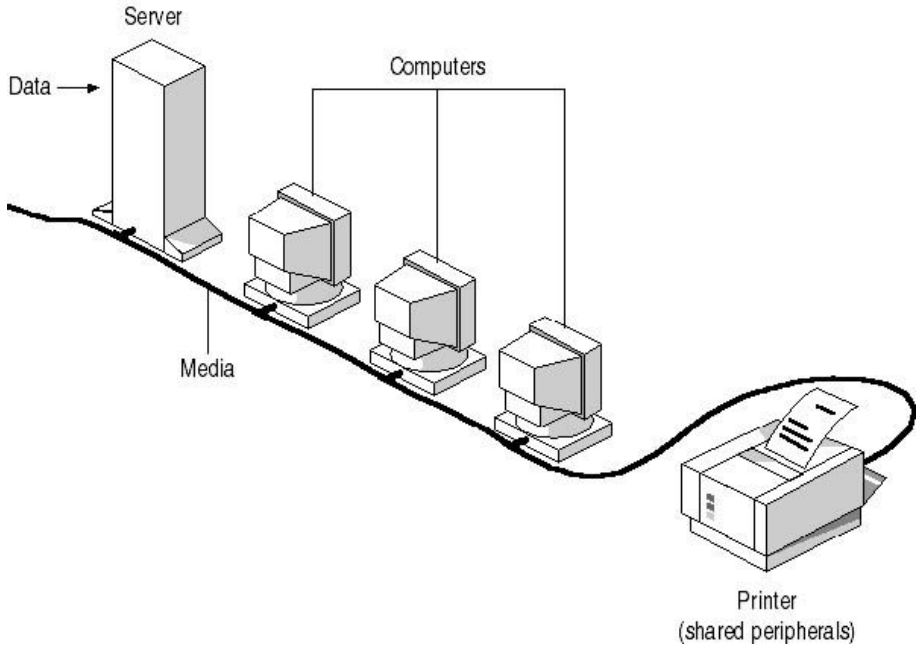
ГВС не ограничена территориально.

Обычно ГВС создается на основе нескольких ЛВС. Самая крупная ГВС - Internet



# Два типа сетей.

## Общие компоненты любой сети:




**Сервер (server)** – компьютер, предоставляющий свои ресурсы для совместного использования.

**Клиент (client)** – компьютер, осуществляющий доступ к сетевым ресурсам, предоставляемым сервером.

**Среда передачи (media)** – способ соединения компьютеров.

**Сети разделяют на два типа:**

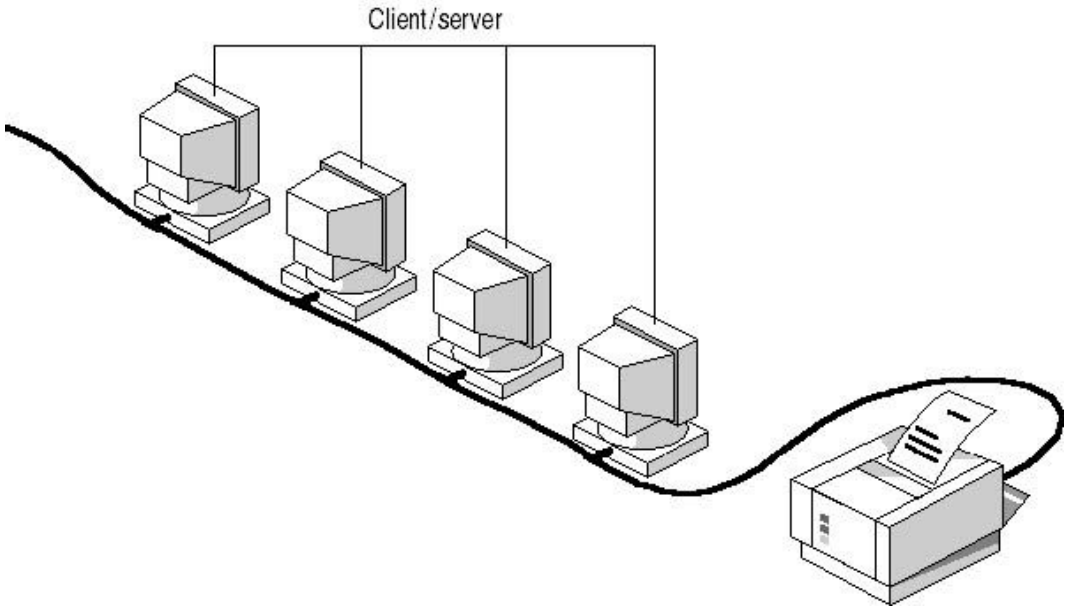
- **Одноранговые (peer-to-peer)**
- **На основе сервера (server based).**

**Различия между ними принципиальны, т.к. предоставляют разные возможности этих сетей.**

**Выбор типа сети зависит от многих факторов:**

- **Размера предприятия**
- **Необходимой степени безопасности данных**
- **Доступности административной поддержки**
- **Объема сетевого трафика**
- **Потребностей сетевых пользователей**
- **Уровня финансирования**

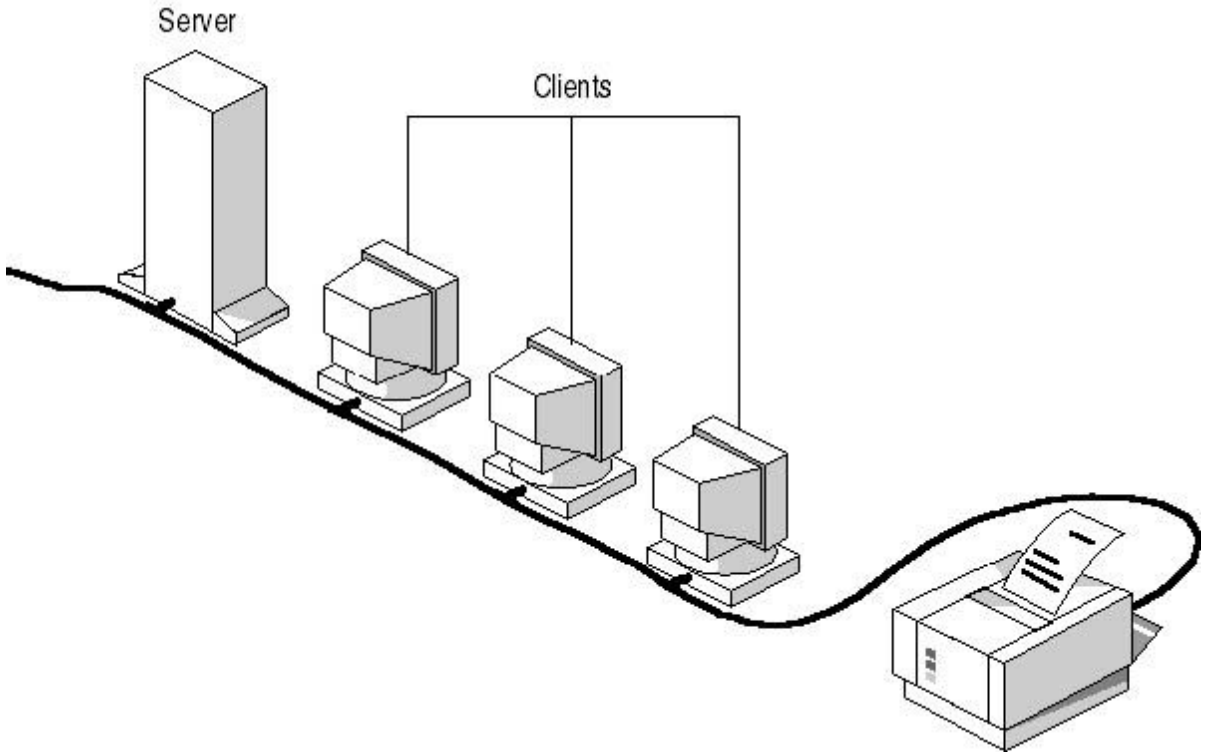
# 1. Одноранговые сети (peer-to-peer)




**В такой сети все компьютеры равноправны:**

- **нет иерархии среди компьютеров - каждый компьютер функционирует и как клиент, и как сервер;**
- **нет выделенного (dedicated) сервера – пользователи сами решают, какие ресурсы на своем компьютере сделать доступными по сети.**

# 2. Сети на основе сервера (Server based network )

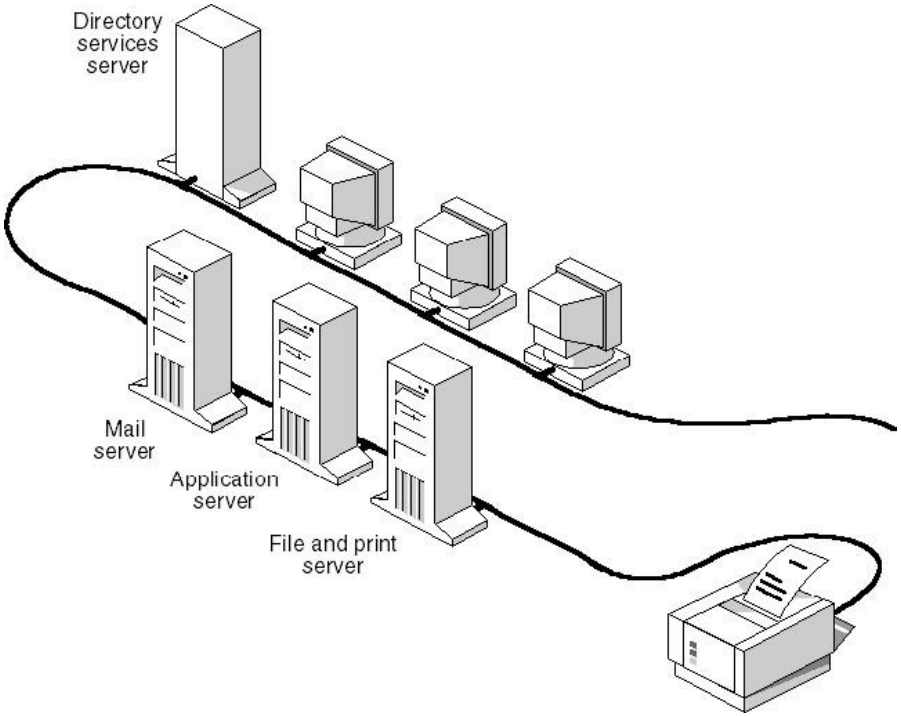



**Работают на основе выделенного (dedicated) сервера.**

**Выделенным называется сервер, который функционирует только как сервер и не используется в качестве клиента.**

**Он оптимизирован для быстрой обработки запросов от сетевых клиентов.**

# Специализированные серверы



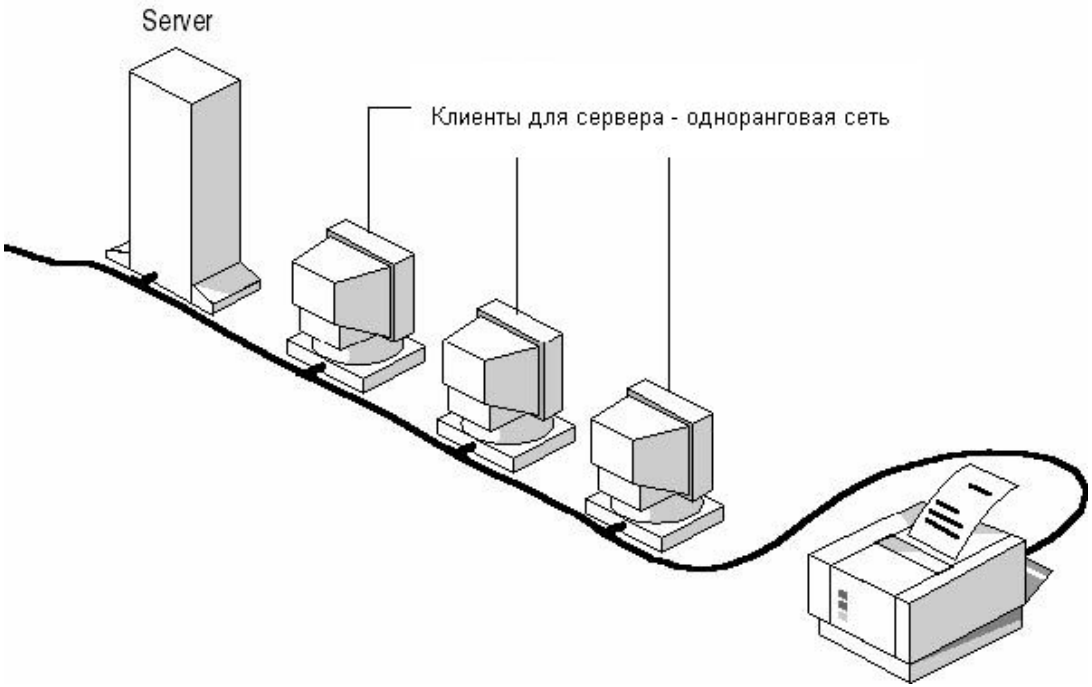

**В большой сети могут работать специализированные серверы.**

**Серверы файлов и печати - управляют доступом пользователей к общим файлам и принтерам**

**Серверы приложений – выполняют прикладные (серверные) части клиент-серверных приложений, а на клиентский компьютер (клиентская часть) пересылаются только результаты.**

**Почтовые серверы – управляют передачей сообщений между пользователями сети.**

# 3. Комбинированные сети




На рабочих станциях работают операционные системы, которые используют ресурсы выделенных серверов и в то же время – по мере необходимости – предоставляют в совместное использование свои ресурсы.

# Понятие сетевая архитектура

Сетевая архитектура (network architecture) – это комбинация топологий, методов доступа к среде передачи данных и протоколов, необходимых для создания работоспособной сети.

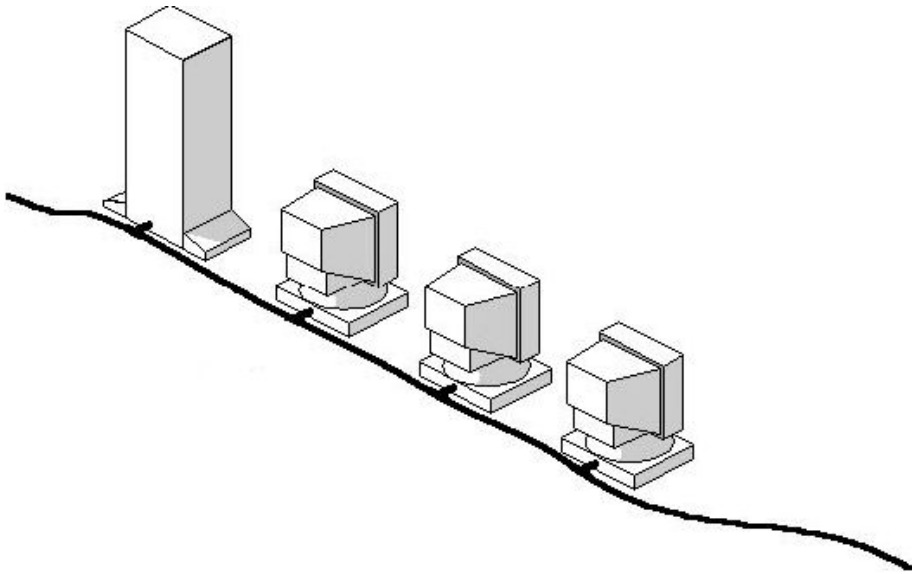
# Топология сети

Термин **сетевая топология** обозначает физическое расположение компьютеров, кабелей и других сетевых компонентов.

Существуют три **базовые топологии** сети:

- **Шина (Bus)**
- **Звезда (Star)**
- **Кольцо (Ring)**

# Шина (Bus)



**Используется один кабель, именуемый магистралью или сегментом, к которому подключены все компьютеры сети.**

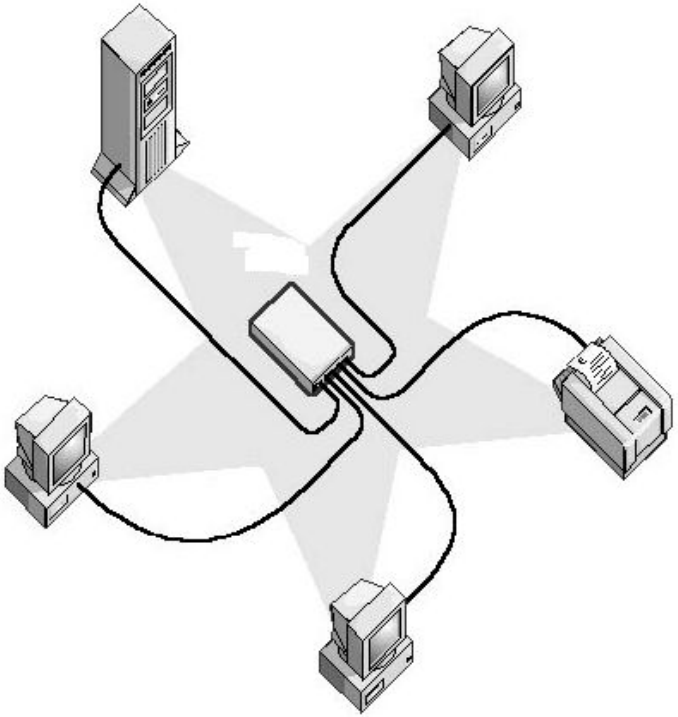
**Данные передаются всем компьютерам сети, однако информацию принимает только один компьютер, чей адрес соответствует адресу получателя, присутствующему среди передаваемых данных.**

**В каждый момент времени передачу может вести только один компьютер.**

***Шина – пассивная топология*. Компьютеры не перемещают данные от отправителя к получателю. Если один компьютер выходит из строя, это не скажется на работе сети.**

**В активных топологиях компьютеры регенерируют сигналы и передают их дальше по сети.**

# Звезда (Star)

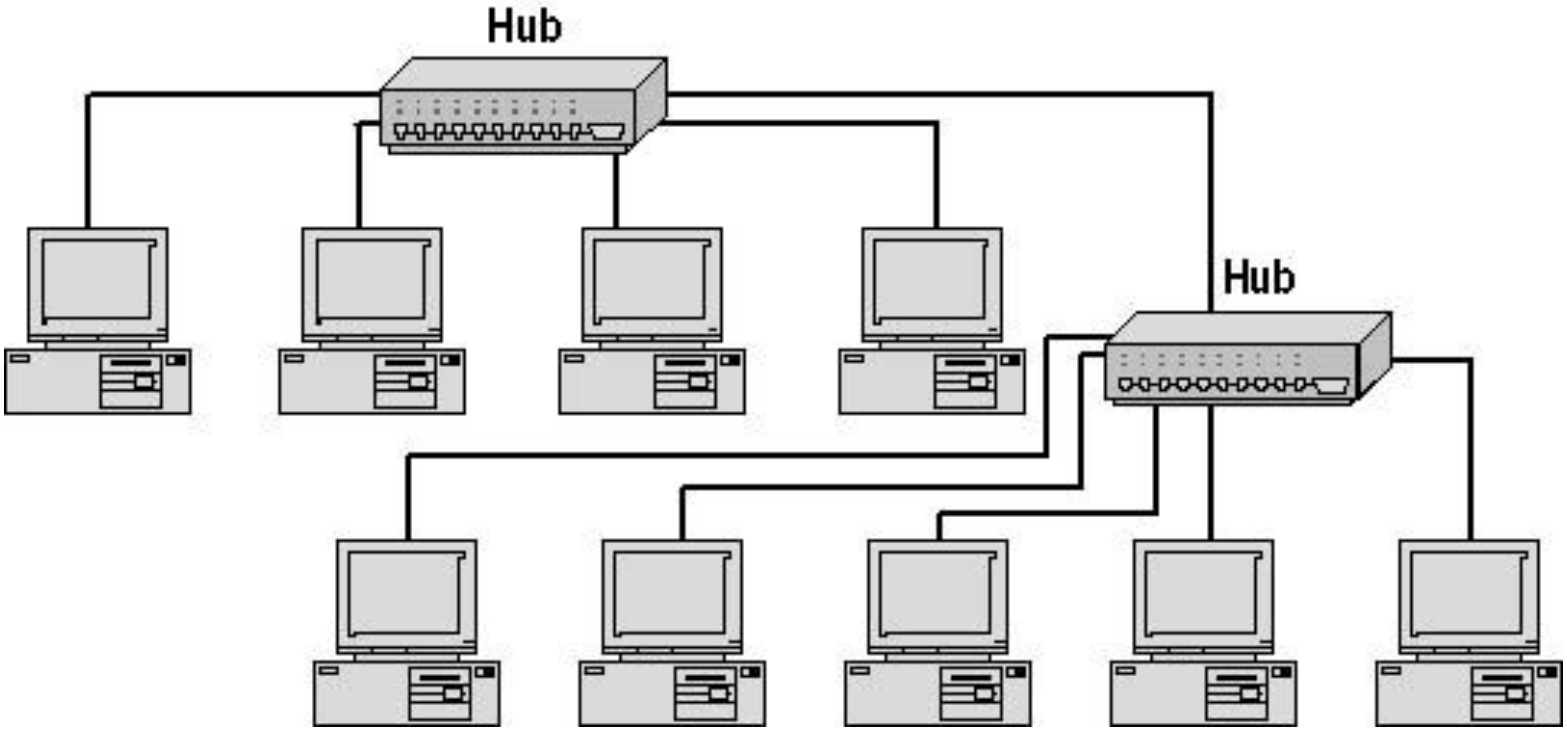


Все компьютеры с помощью сегментов кабеля подключаются к центральному устройству.

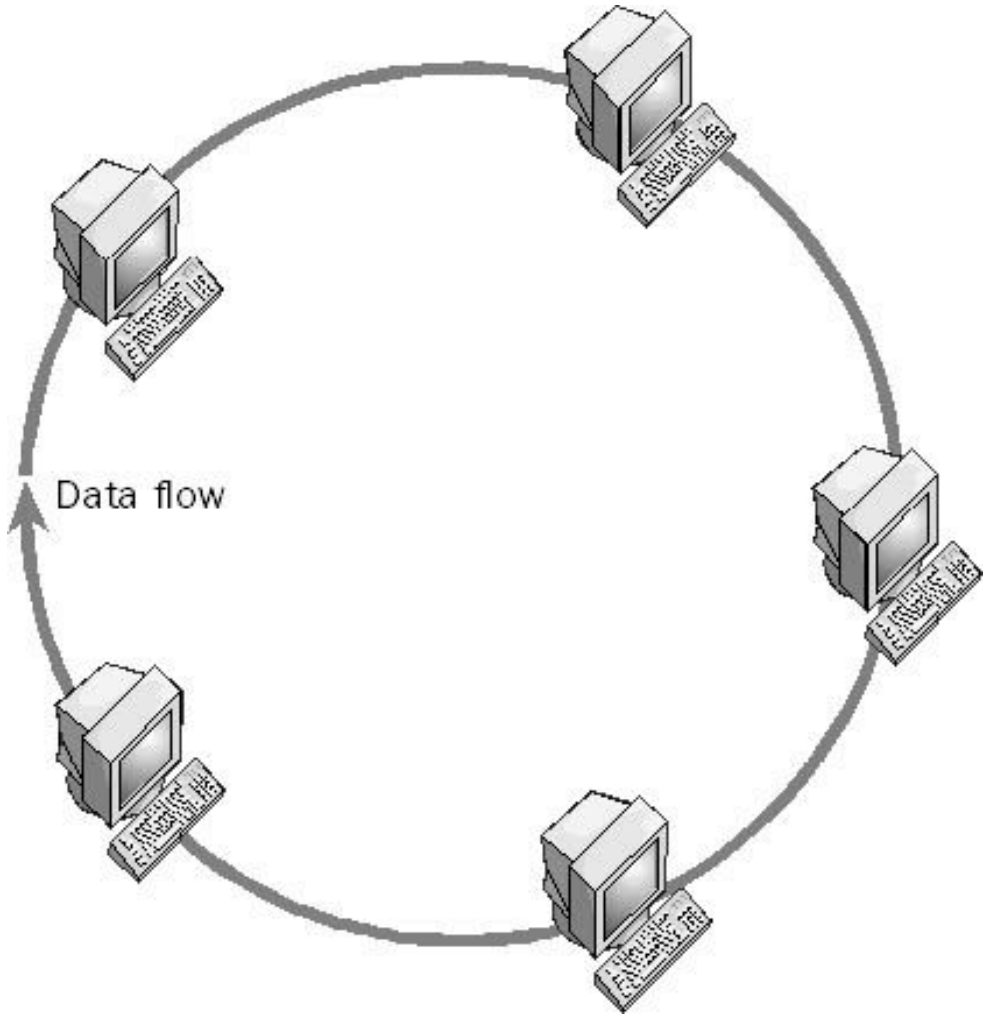
При выходе из строя одного компьютера или одного сегмента кабеля, только этот компьютер не работает в сети.

Если центральный компонент выходит из строя, не работает вся сеть.

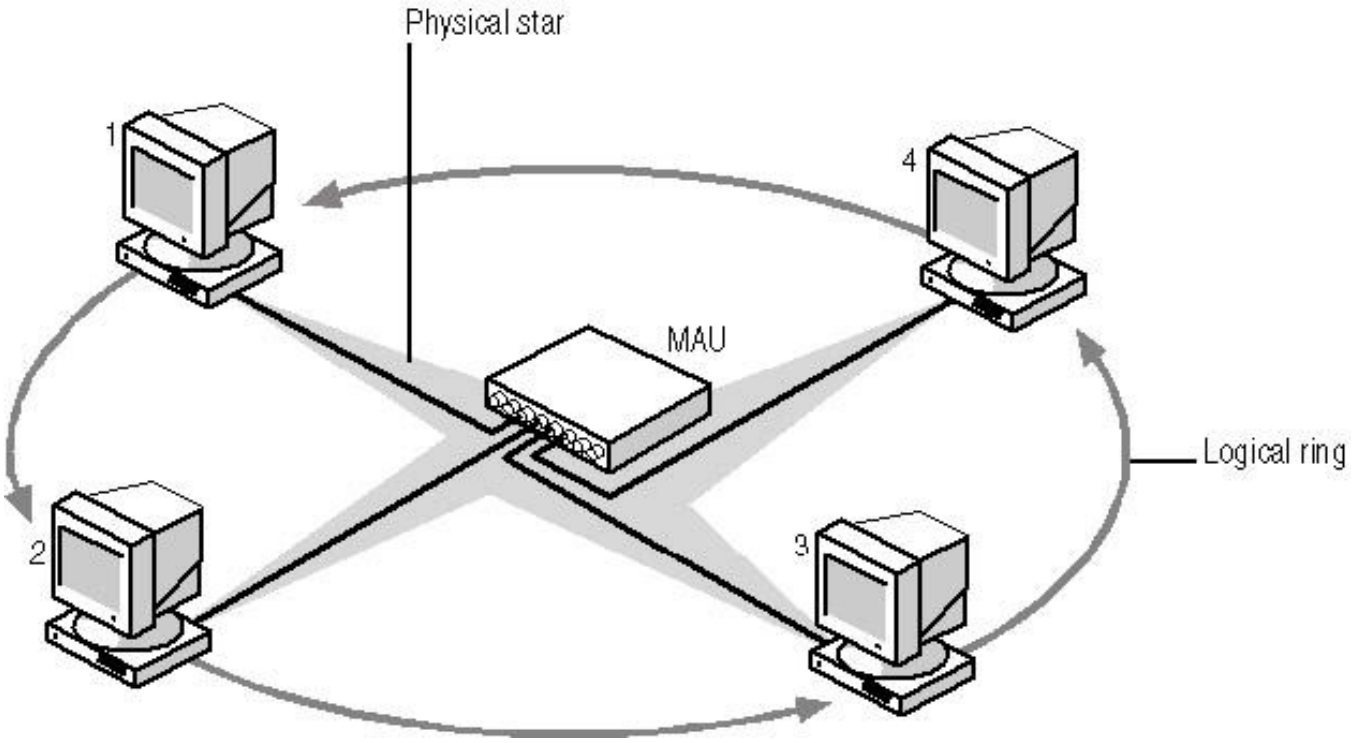
# Топология иерархическая звезда – используется несколко концентраторов



# Кольцо (Ring)

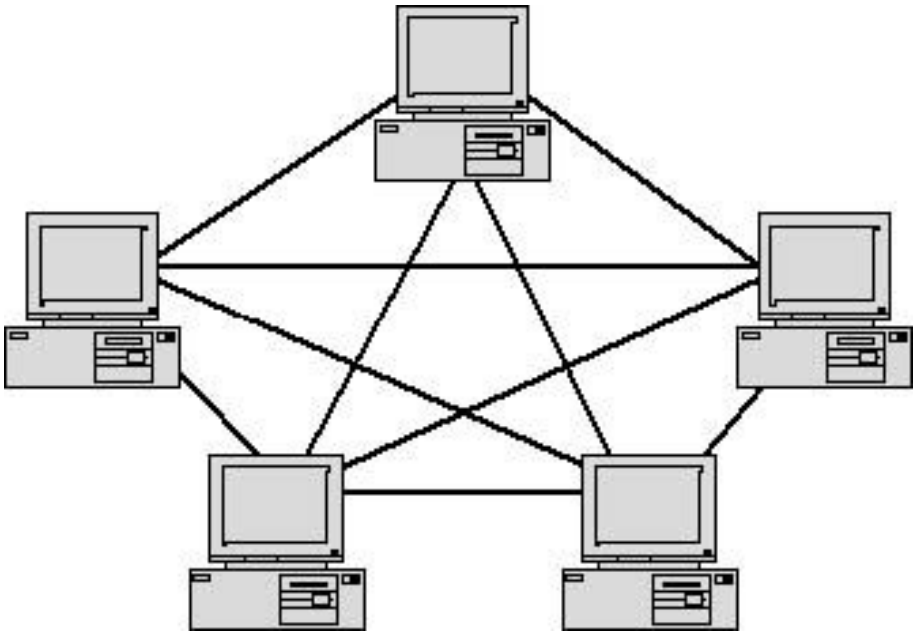


# Логическое кольцо - Физическая звезда

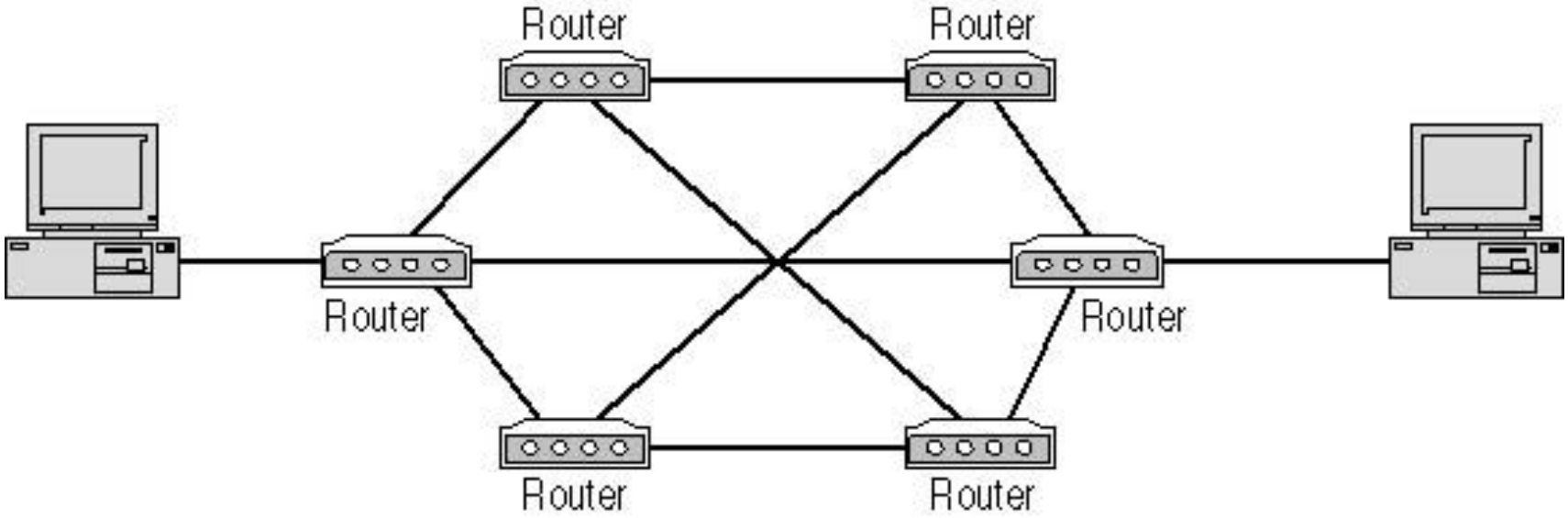


# Ячеистая топология (mesh topology)

**Все компьютеры связаны друг с другом отдельными соединениями. Для локальных сетей существует скорее в виде теоретической концепции (реальный пример – соединение двух компьютеров).**

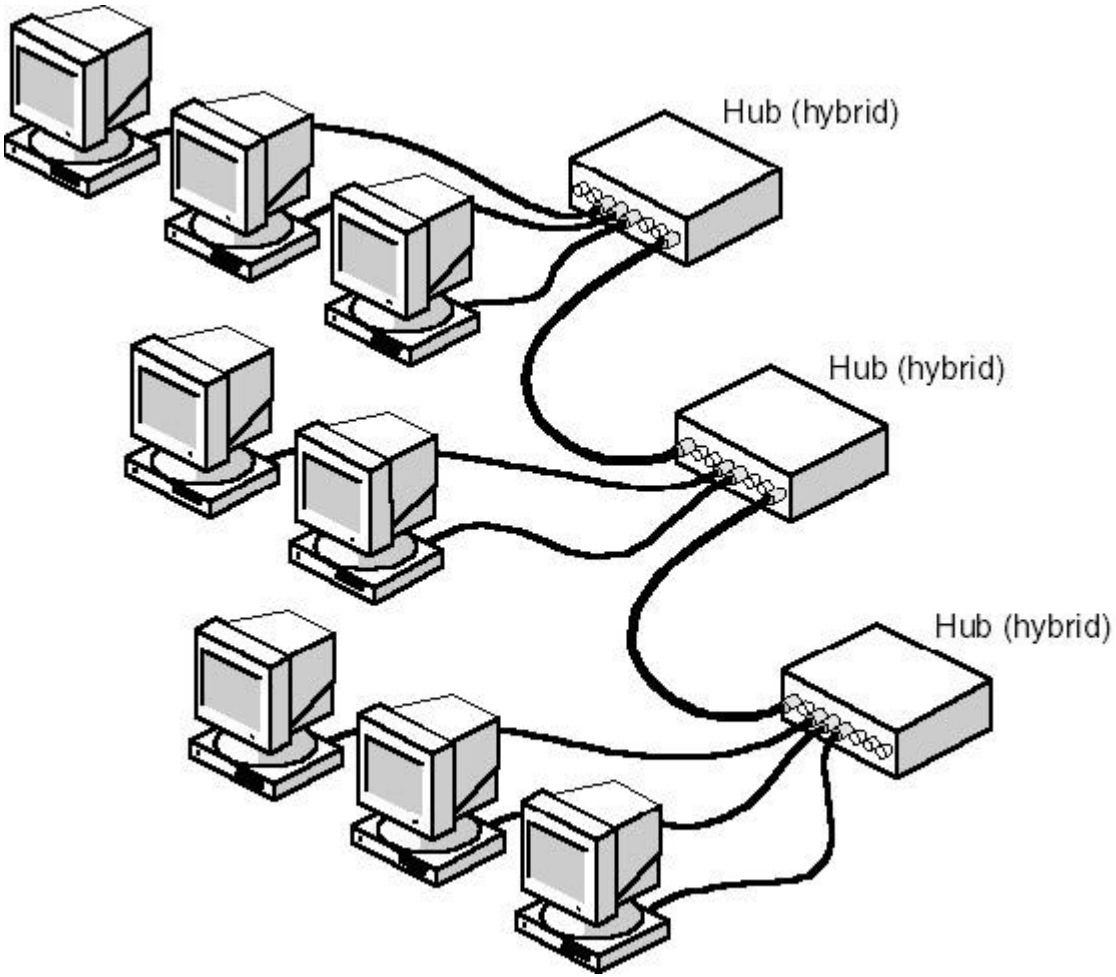


**В ГВС (интерсети) ячеистая топология используется. В такой сети благодаря использованию избыточных маршрутизаторов данные могут доставляться от одной системы к другой несколькими путями.**

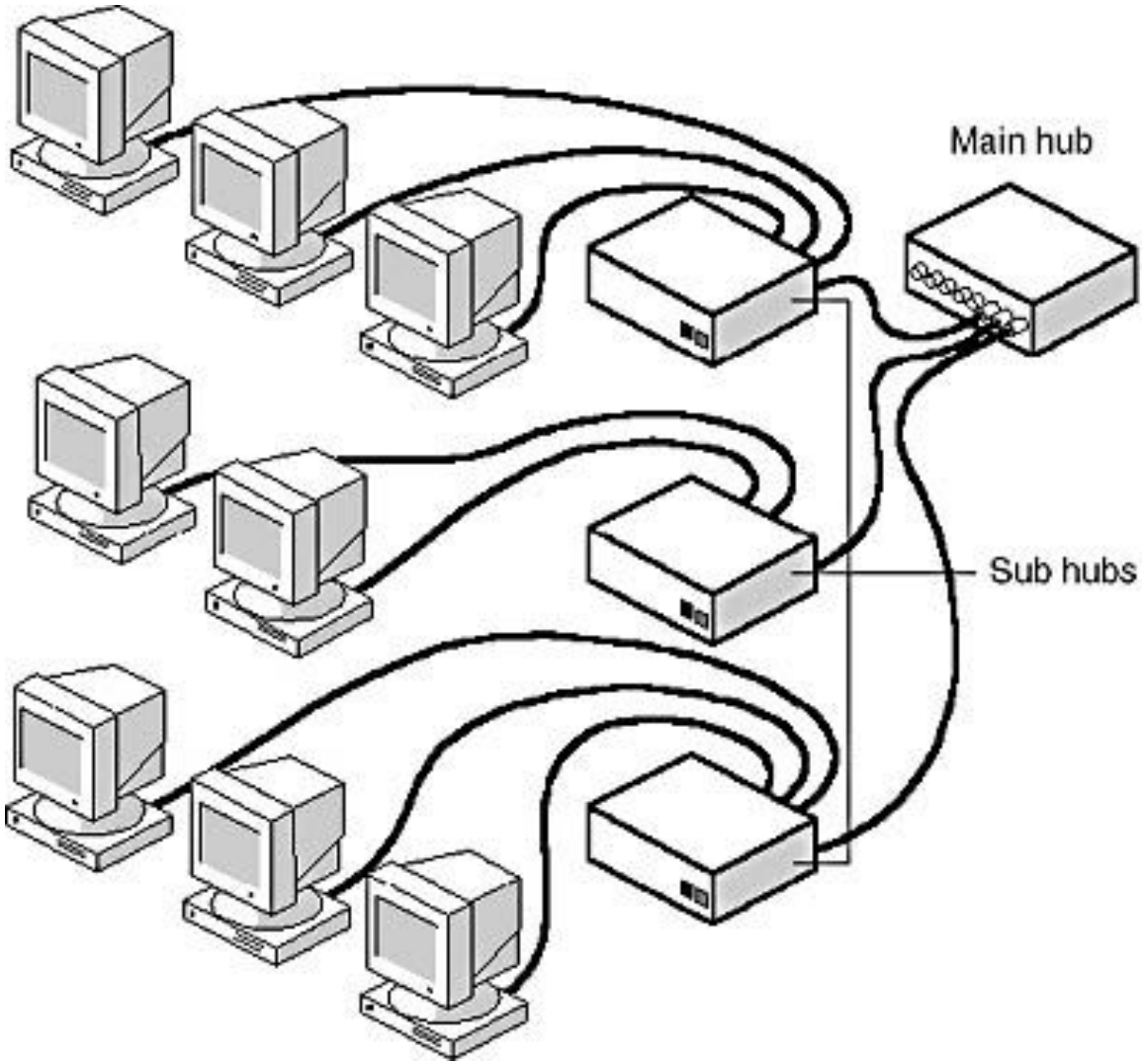


# Комбинированные топологии

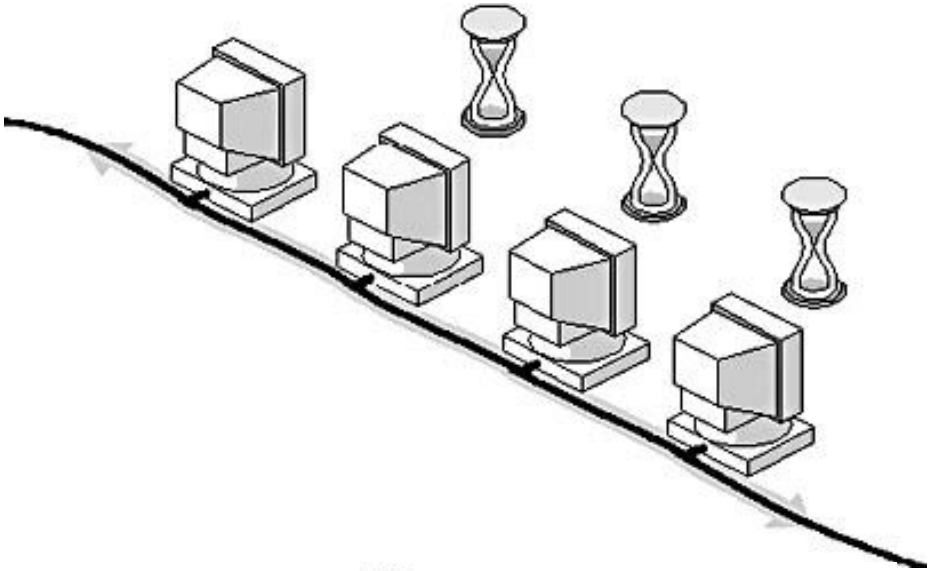
## Звезда-шина (star-bus)



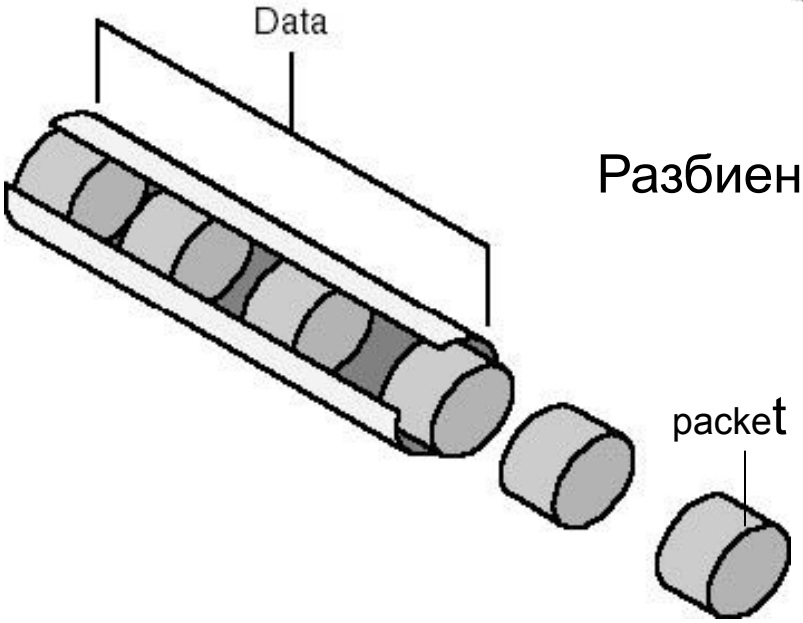
# Звезда-кольцо (star-ring)



# Передача данных по сети

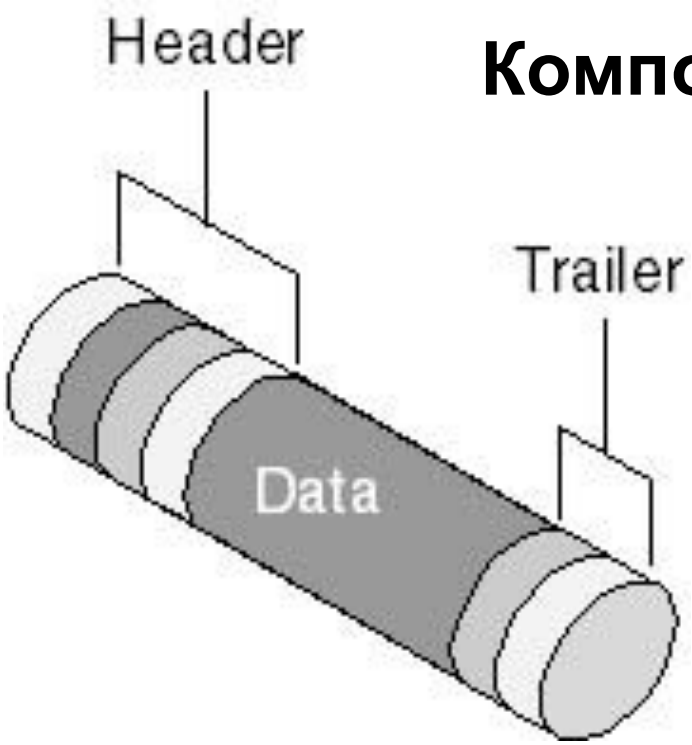


Большие блоки данных замедляют работу сети




Разбиение данных на пакеты

# Компоновка пакета




## Заголовок (header)

- Сигнал “говорящий” о том, что передается пакет
- Адрес источника
- Адрес получателя

## Данные (data)

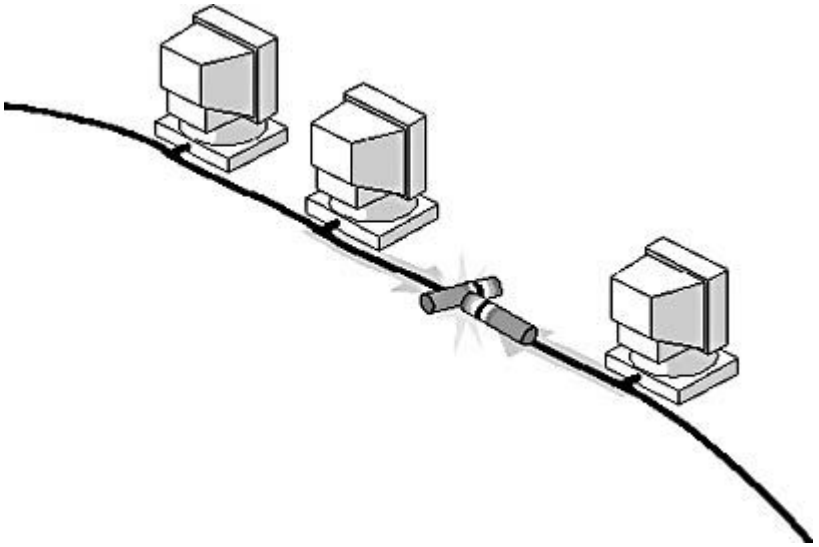
- Передаваемые данные (в зависимости от типа сети от 0,5 до 4 Кбайт)

## Трейлер (trailer или footer)

- Содержимое зависит от протокола связи. Чаще всего содержит информацию для проверки ошибок при передаче. (Циклический избыточный код – Cyclical Redundancy Code, CRC)

# Методы доступа к среде передачи данных

Метод доступа – набор правил, которые определяют, как компьютер должен отправлять и принимать данные.



**При попытке одновременной передачи данных по сети пакеты сталкиваются, информация портится – возникает коллизия**

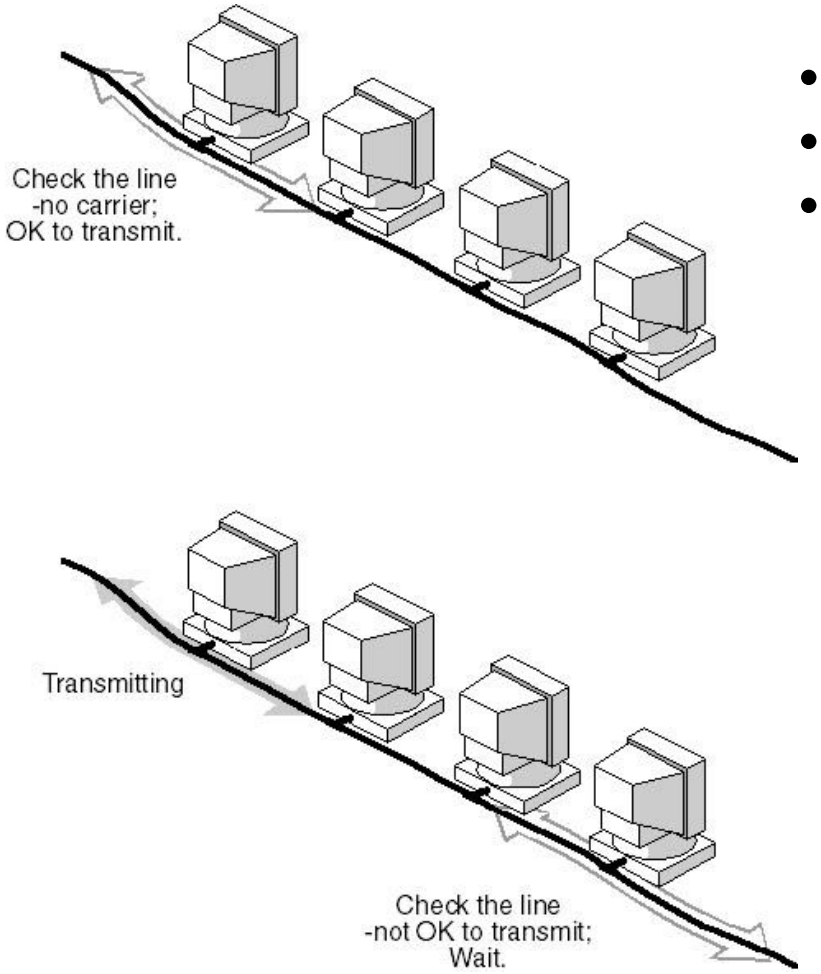
# Основные методы доступа

**Существует три способа предотвратить одновременную передачу – три основных метода доступа:**

- **Множественный доступ с контролем несущей и с обнаружением коллизий**
- **Множественный доступ с контролем несущей и с предотвращением коллизий**
- **Доступ с передачей маркера**

# Множественный доступ с контролем несущей и с обнаружением коллизий
# Carrier-Sense Multiple Access with Collision Detection

# (CSMA/CD)




- Передающий проверяет сеть
- Несущая отсутствует
- Можно передавать

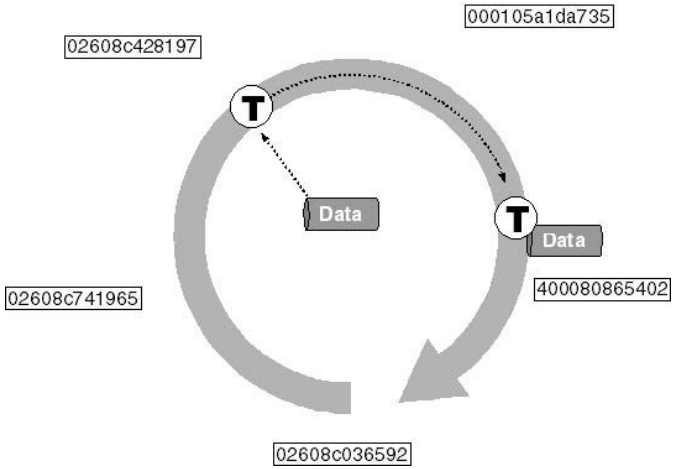
- Передающий проверяет сеть
- Несущая обнаружена
- Передавать нельзя - ожидание

- После обнаружения коллизии передающий компьютер обязан прекратить передачу, сделать паузу, затем может снова попытаться передавать.

# Множественный доступ с контролем несущей и с предотвращением коллизий
# Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA)

- Передающий проверяет сеть
- Несущая отсутствует
- Сигнализирует о намерении передавать
- Передает

# Доступ с передачей маркера
# Token Passing




- Пакет особого типа, маркер (token), циркулирует по кольцу от компьютера к компьютеру
- Компьютер выполняющий передачу захватывает маркер и наполняет своими данными. Пока маркер захвачен другие компьютеры не могут передавать информацию
- Маркер достигает приемника
- Приемник копирует информацию в буфер и делает отметку о получении информации
- Когда маркер вновь достигает отправителя, тот удостоверяется, что передача прошла успешно, изымает из маркера свои данные и отправляет маркер в сеть

# Сетевые протоколы

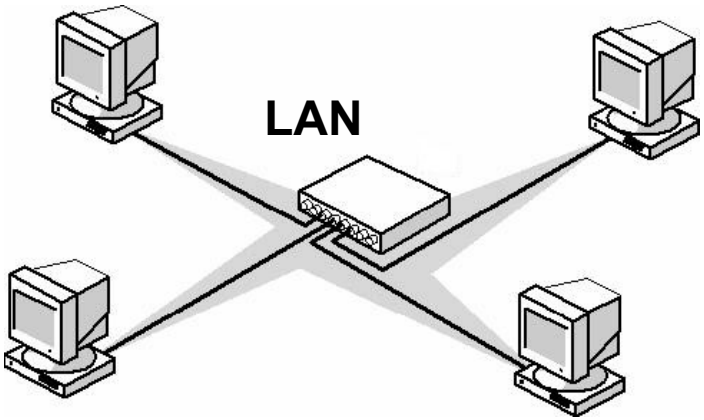
**Протокол – набор правил и процедур, регулирующих порядок осуществления некоторой связи (например, дипломатический протокол).**

***Сетевой протокол*** **– правила и технические процедуры, позволяющие компьтерам , объединенным в сеть, осуществлять соединение и обмен данными.**

**Три основные момента, касающиеся протоколов:**

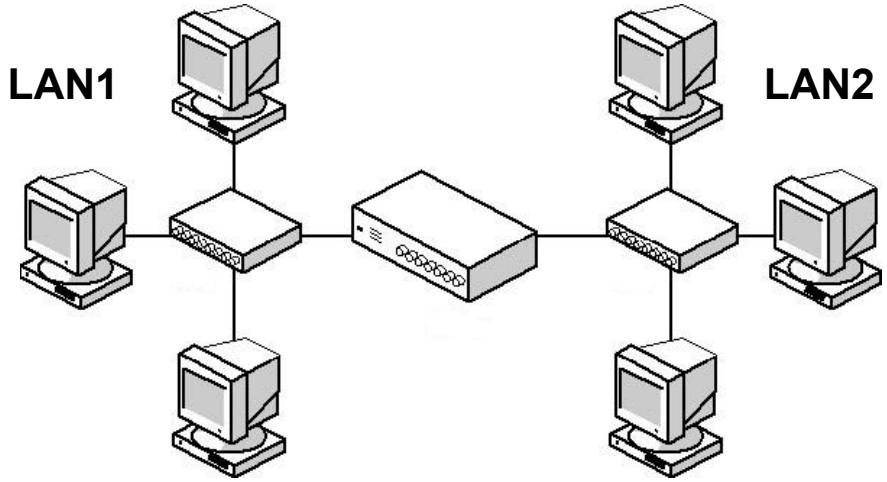
- **Существует множество протоколов. И хотя все они участвуют в реализации связи, каждый протокол имеет различные цели, выполняет различные задачи.**
- **Протоколы работают на разных уровнях модели OSI (см. ниже). Функции протокола определяются уровнем, на котором он работает.**
- **Несколько протоколов могут работать совместно. В этом случае они образуют так называемый *стэк протоколов* или набор протоколов.**

# Маршрутизируемые и немаршрутизируемые протоколы




**Немаршрутизируемые протоколы**

- **могут обеспечить связь между компьютерами только внутри локальной сети**




**Маршрутизируемые протоколы**

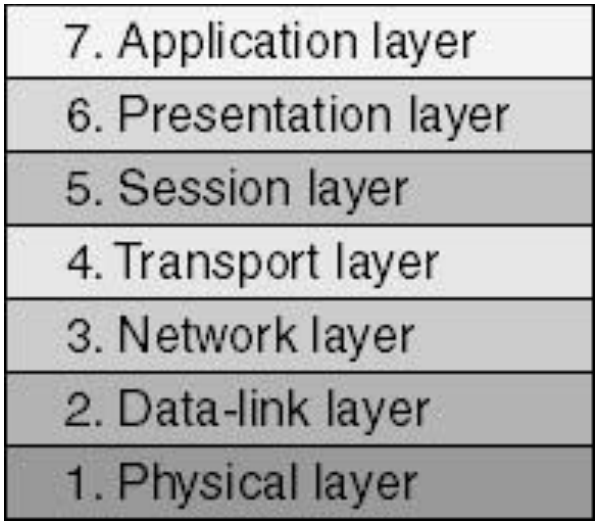
- **могут обеспечить связь между компьютерами внутри локальной сети**
- **могут обеспечить связь между локальными сетями (между компьютерами из разных локальных сетей)**

# Модель OSI

***Сетевая модель OSI*** **(эталонная модель взаимодействия открытых систем — англ. Open Systems Interconnection Reference Model-OSI) — абстрактная модель для сетевых коммуникаций и разработки сетевых протоколов.**

***Назначение модели OSI*** **состоит в обобщенном представлении средств сетевого взаимодействия.**

**Для наглядности процесс работы сети разделен на семь уровней. В верхней части модели располагается приложение, которому нужен доступ к сети, в нижней – сетевая среда передачи данных. По мере того, как данные продвигаются от уровня к уровню вниз, действующие на этих уровнях протоколы постепенно подготавливают эти данные для передачи по сети. Каждый уровень обслуживает свою часть процесса взаимодействия.**




**7. Прикладной уровень (Application Layer)**

**6. Уровень представления (Presentation Layer)**

**5. Сеансовый уровень (Session Layer)**

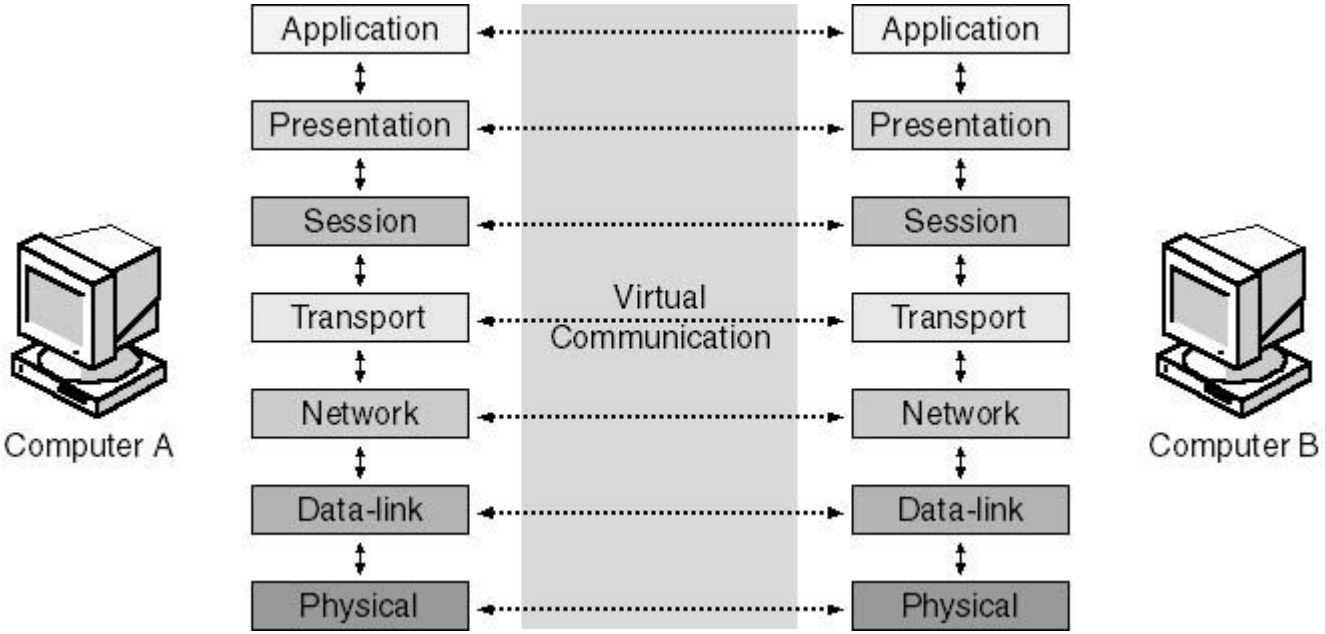
**4. Транспортный уровень (Transport Layer)**

**3. Сетевой уровень (Network Layer)**

**2. Канальный уровень (Data-Link Layer)**

**1. Физический уровень (Physical Layer)**

# Взаимодействие уровней модели OSI




**Задача каждого уровня – предоставление услуг смежному уровню, «маскируя» детали реализации этих услуг.**

**Каждый уровень на компьютере-отправителе работает так, будто он напрямую связан с таким же уровнем на получателе – это логическая или виртуальная связь.**

**В действительности связь осуществляется между смежными уровнями одного компьютера – программное обеспечение, работающее на каждом уровне реализует сетевые функции в соответствии с набором протоколов этого уровня.**

# Инкапсуляция данных

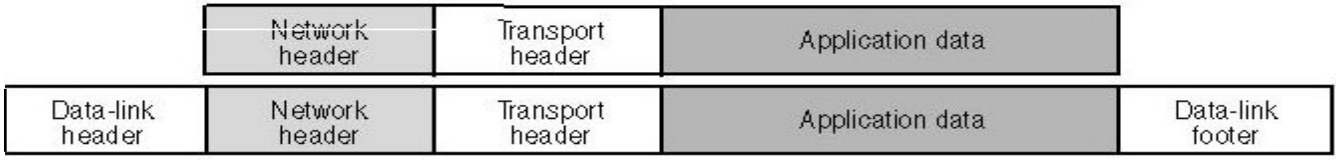
**По сути, взаимодействие протоколов, работающих на разных уровнях модели OSI, состоит в том, что каждый протокол добавляет свой заголовок (header), содержащий поля с информацией специфичной для данного уровня, к информации, полученной с уровня, расположенного выше.**

**При этом, информация полученная с верхнего уровня – заголовок и данные, становятся данными для протокола текущего уровня.**

**На канальном уровне добавляется заголовок и трейлер (footer). Итог - пакет, готовый к передаче по сети.**

**Процесс добавления заголовков к запросу, сгенерированному приложением, называется *<u>инкапсуляция</u>***

***Пример:***



# Основные функции протоколов уровней OSI

**1. Физический уровень (Physical Layer)**

**Имеет дело с передачей битов по физическим каналам связи (различные типы кабелей, беспроводные каналы). На этом уровне определяется тип сигнала для передачи данных по сетевой среде (электрический сигнал, световой импульс и т.д.) и его характеристики (уровень, частота и т.д).**

**2. Канальный уровень (Data-Link Layer)**

**Основные функции протокола канального уровня:**

**А. Формирование кадра (пакета) для передачи по сети.**

**Протокол канального уровня добавляет к данным полученным от сетевого уровня заголовок и трейлер, превращая их в кадр.**

**В заголовке содержатся адреса системы-отправителя и системы получателя пакета. Это так называемые аппаратные адреса или MAC-адреса, присвоенные сетевым адаптерам на заводе изготовителе (MAC – Media Access Control – управление доступом к среде).**

**Б. Реализация механизма контроля доступа к среде (методы доступа CSMA/CD, CSMA/CA, Token Passing и др).**

**Функции протокола канального уровня реализуются сетевыми адаптерами и их драйверами.**

## 3. Сетевой уровень (Network Layer)

**Протоколы сетевого уровня обеспечивают «сквозную» передачу пакета от передающего до принимающего компьютера (end-to-end). При этом передатчик и приемник могут находится в одной ЛВС или в разных ЛВС, соединенных между собой специальными устройствами – маршрутизаторами (шлюзами).**

**Пример: протокол сетевого уровня – IP (Internet Protocol), который входит в стек протоколов TCP/IP.**

## 4. Транспортный уровень (Transport Layer)

**Протоколы транспортного уровня обеспечивают приложениям ту степень надежности доставки сообщения, которая им требуется.**

**Существует два типа протоколов транспортного уровня:**

**А. Протоколы ориентированные на соединение (connection-oriented)**

**Такие протоколы перед передачей данных обмениваются сообщениями, чтобы установить связь друг с другом. После установки связи выполняется передача, а затем протоколы обмениваются сообщениями о доставке пакета.**

**Пример: протокол TCP (Transmission Control Protocol) – входит в стек протоколов TCP/IP, обеспечивает приложениям гарантированную доставку данных с подтверждением приема, обнаружением и коррекцией ошибок.**

**Б. Протоколы не ориентированные на соединение (connectionless)**

**Передают информацию целевой системе не проверяя готова ли она к приему и существует ли она вообще**

**Пример: протокол UDP (User Datagram Protocol) – входит в стек протоколов TCP/IP, не обеспечивает приложениям гарантированную доставку данных.**

## 5. Сеансовый уровень (Session Layer)

**Обеспечивает процесс взаимодействия сторон, фиксирует какая из сторон сейчас является активной и предоставляет средства синхронизации сеанса. Эти средства позволяют в ходе длинных передач сохранять информацию о состоянии этих передач в виде контрольных точек, чтобы в случае отказа можно было вернуться назад к последней контрольной точке, а не начинать все сначала. Этот уровень редко реализуется в виде отдельных протоколов. Функции этого уровня часто объединяют с функциями прикладного уровня и реализуют в одном протоколе.**

## 6. Уровень представления (Presentation Layer)

**На этом уровне выполняется функция трансляции синтаксиса между различными системами (например, различная кодировка символов в разных системах – ASCII и EBCDIC).**

## 7. Прикладной уровень (Application Layer)

**Это набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к ресурсам, таким как файлы, принтеры, гипертекстовые документы, а также организуют свою совместную работу, например, по протоколу электронной почты.**

**Единица данных, которой оперирует прикладной уровень, обычно называется сообщением.**

# Стандартные стеки и уровни протоколов

Стек протоколов – это некоторая комбинация протоколов, которые работают в сети одновременно и обеспечивают следующие операции с данными:

- Подготовку
- Передачу
- Прием

Работа различных протоколов должна быть скоординирована так, чтобы исключить конфликты или незаконченные операции – этого можно достичь с помощью разбиения стеков протоколов на уровни.

В компьютерной промышленности в качестве стандартных моделей разработано несколько стеков протоколов. Наиболее известные из них:

- NetWare фирмы Novell
- AppleTalk фирмы Apple
- TCP/IP – стек протоколов Internet

Коммуникационные задачи, которые возложены на сеть, позволяют выделить среди протоколов разных стеков три типа (три уровня) протоколов:

- Прикладные
- Транспортные
- Сетевые

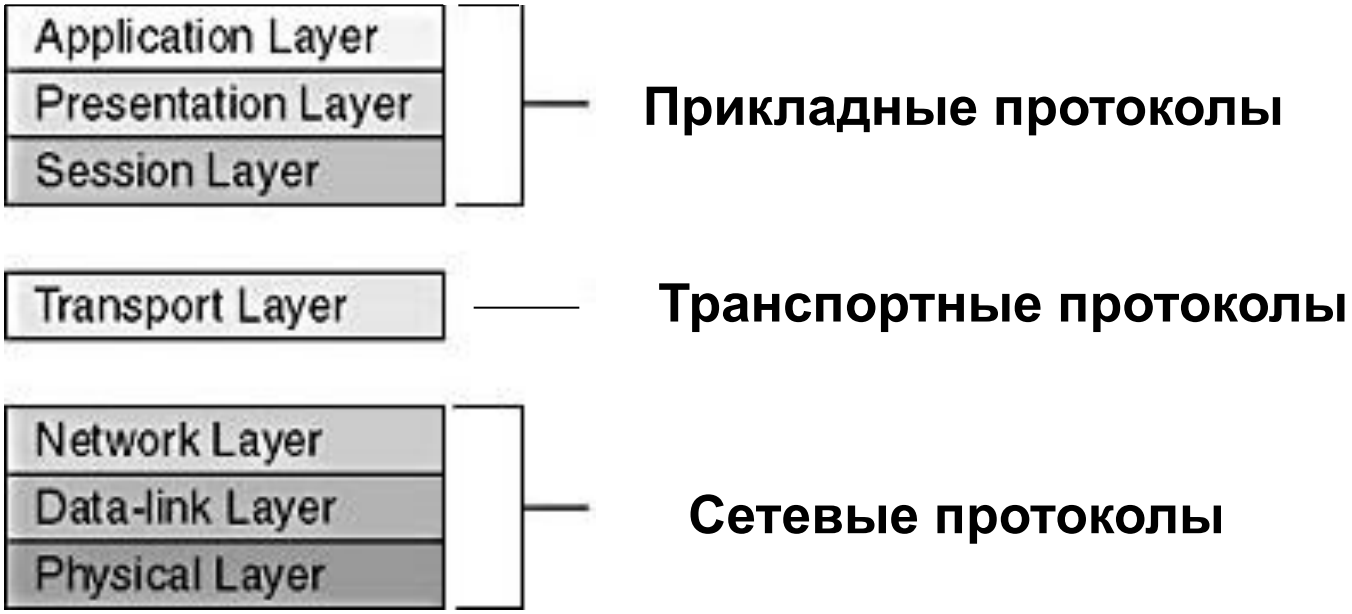



Схема расположения этих протколов соответствует уровням модели OSI

# Стек протоколов TCP/IP
## (Transmission Control Protocol/Internet Protocol)

**TCP/IP – набор протоколов, которые обеспечивают связь в гетерогенной (неоднородной) среде, т.е. обеспечивает совместимость между компьютерами разных типов.**

**Совместимость – одно из основных преимуществ TCP/IP, поэтому большинство ЛВС поддерживает его.**

**TCP/IP маршрутизируемый протокол – используется в качестве межсетевого протокола.**

**TCP/IP стал стандартом де-факто для межсетевого взаимодействия**

# Четырехуровневая модель TCP/IP

**Протоколы TCP/IP соответствуют четырехуровневой модели, известной как модель DARPA.**

**Каждый уровень этой модели соответствует одному или нескольким уровням модели OSI.**

<table>
<tr><th colspan="2">OSI</th><th colspan="7">TCP/IP</th></tr>
<tr><td>7</td><td>Прикладной</td><td colspan="6">Прикладной</td><td rowspan="3">I</td></tr>
<tr><td>6</td><td>Представления</td><td rowspan="2">Telnet</td><td rowspan="2">FTP</td><td rowspan="2">SMTP</td><td rowspan="2">HTTP</td><td rowspan="2">RIP</td><td rowspan="2">SNMP</td></tr>
<tr><td>5</td><td>Сеансовый</td></tr>
<tr><td rowspan="2">4</td><td rowspan="2">Транспортный</td><td colspan="6">Транспортный</td><td rowspan="2">II</td></tr>
<tr><td colspan="3">TCP</td><td colspan="3">UDP</td></tr>
<tr><td rowspan="2">3</td><td rowspan="2">Сетевой</td><td colspan="6">Межсетевой</td><td rowspan="2">III</td></tr>
<tr><td colspan="6">IP</td></tr>
<tr><td>2</td><td>Канальный</td><td colspan="6" rowspan="2">Сетевой интерфейс<br>Не регламентируется:<br>Ethernet, TokenRing, X.25, ATM и т.д. ….</td><td rowspan="2">IV</td></tr>
<tr><td>1</td><td>Физический</td></tr>
</table>

# Основные протоколы стека TCP/IP

I. **Прикладной уровень –** обеспечивает приложениям доступ к сервисам других уровней и определяют протоколы, по которым приложения могут обмениваться данными

На этом уровне предусмотрено много протоколов и постоянно разрабатываются новые.

- Telnet – протокол эмуляции терминала, используется для регистрации на удаленных компьютерах
- FTP (File Transport Protocol) – протокол для передачи файлов
- HTTP (Hypertext Transfer Protocol) – протокол для работы с гипертекстовыми документами, образующими содержимое Web-страниц в World Wide Web

Следующие протоколы упрощают использование и управление TCP/IP-сетями

- SMTP (Simple Mail Transfer Protocol) – протокол для передачи почтовых сообщений
- SNMP (Simple Network Management Protocol) – протокол управления сетью
- RIP (Routing Information Protocol) – протокол маршрутизации

## II. Транспортный уровень

**Предоставляет прикладному уровню сеансовые коммуникационные службы.**

- **TCP (Transmission Control Protocol) – обеспечивает надежную, требующую логического соединения связь только между двумя компьютерами. Отвечает за установление соединения, упорядочивание посылаемых пакетов и восстановление пакетов, потерянных в процессе передачи.**
- **UDP (User Datagram Protocol) – обеспечивает ненадежную, не требующую логического соединения связь. Используется, когда объем данных невелик (например, данные могут уместиться в одном пакете), когда издержки установления TCP соединения нежелательны либо когда протоколы верхнего уровня или приложения гарантируют надежную доставку. UDP используется для передачи данных на несколько компьютеров с использованием многоадресной рассылки, например, многоадресная рассылка потокового мультимедиа при проведении видеоконференций в реальном времени.**

# III. Межсетевой уровень

- **IP (Internet Protocol – межсетевой протокол) – маршрутизируемый протокол, отвечающий за IP-адресацию, маршрутизацию, фрагментацию и восстановление пакетов. В его задачу входит продвижение пакета между сетями – от одного маршрутизатора до другого до тех пор, пока пакет не попадет в сеть назначения. В отличие от протоколов прикладного и транспортного уровней протокол IP разворачивается не только на хостах, но и на всех шлюзах (маршрутизаторах). Этот протокол работает без установления соединения, без гарантированной доставки.**
- **ARP (Address Resolution Protocol) – обеспечивает преобразование адресов межсетевого уровня (IP-адресов) в адреса уровня сетевого интерфейса (MAC-адреса)**
- **ICMP (Internet Control Message Protocol) – поддерживает диагностические функции и сообщает об ошибках в случае неудачной доставки IP-пакетов**
- **IGMP (Internet Group Management Protocol) – управляет членством компьютера (хоста) в группах. Хосты входящие в группу слушают трафик, направляемый на определенный адрес (адрес групповой рассылки) и принимают все пакеты, присылаемые на этот адрес.**

## IV. Уровень сетевых интерфейсов

**Уровень сетевых интерфейсов в стеке TCP/IP отвечает за организацию взаимодействия с технологиями сетей, входящими в составную сеть. Этот уровень в стеке TCP/IP не регламентируется. Он поддерживает все популярные технологии (Ethernet, TokenRing и т.д.). Обычно при появлении новой сетевой технологии она быстро включается в стек TCP/IP путем разработки соответствующей документации**

# Единицы передачи данных для протоколов различных уровней

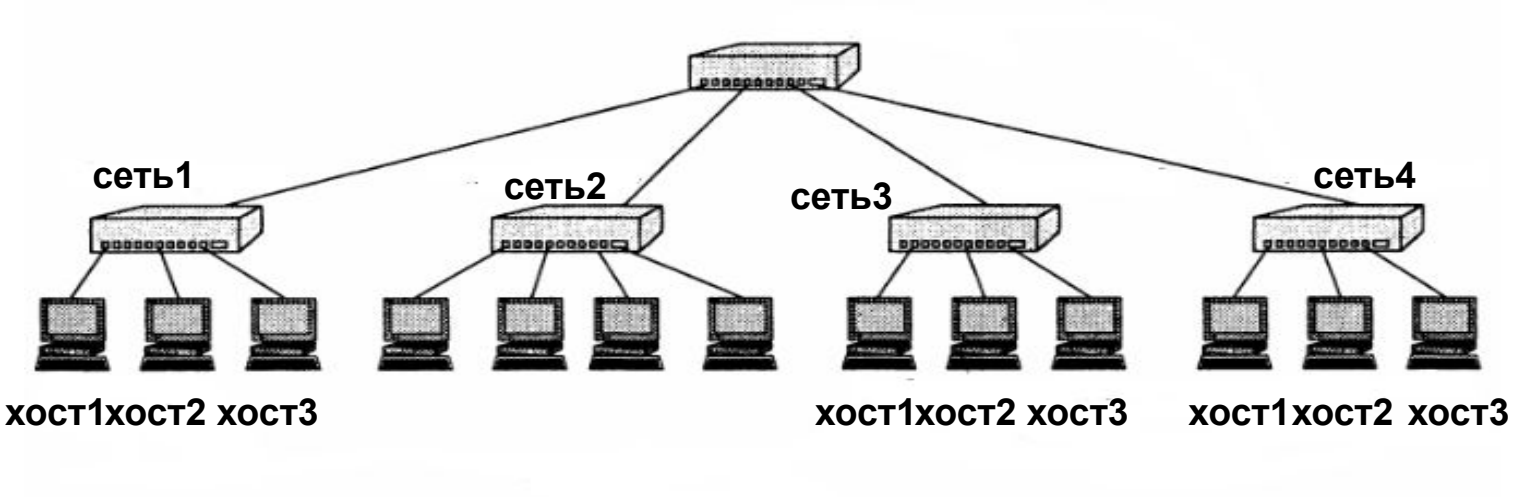
# IP-адресация

**IP-адрес – уникальный идентификатор, однозначно определяющий узел (хост) в сети, использующей протокол TCP/IP.**

**Узел (node) или хост (host) – устройство, подключенное к сети и способное взаимодействовать с другими устройствами.**

Сетевой адрес состоящий из номера сети и номера хоста в сети позволит уникальным образом идентифицировать каждый хост в большой составной сети.

**В технологии TCP/IP сетевой адрес называют IP-Адрес**



# IPv4

**IP-адрес – 32-х разрядное двоичное число**

**Для удобства записывается в специальном формате – десятичное с точкой (dotted decimal)**

| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8-р - октет | | | | | | | | 8-р - октет | | | | | | | | 8-р - октет | | | | | | | | 8-р - октет | | | | | | | |
| Десятичное число | | | | | | | | Десятичное число | | | | | | | | Десятичное число | | | | | | | | Десятичное число | | | | | | | |
| W | | | | | | | | X | | | | | | | | Y | | | | | | | | Z | | | | | | | |

**W.X.Y.Z – десятичное с точкой**

**181.252.30.115**

# Преобразование двоичного формата в десятичный

| 8-р - октет | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| весовые коэффициенты | | | | | | | |
| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

**Пример:**

**1 0 0 1 1 0 0 1**

**128+ 0+ 0+ 16+ 8+ 0+ 0+ 1 =153**

***IP-адрес назначается не по принципу последовательного перечисления хостов, а разбивается на две части:***

- ***<u>Идентификатор сети (network ID)</u> – определяет физическую сеть. Он одинаков для всех узлов в одной сети и уникален для каждой сети, включенной в объединенную сеть.***
- ***<u>Идентификатор хоста (host ID)</u> – соответствует конкретному узлу (компьютеру, маршрутизатору и т. д) в данной сети***

| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Идентификатор сети** | | | | | | | | | | | | | | | | **Идентификатор хоста** | | | | | | | | | | | | | | | |

**Идентификатор сети занимает старшую часть IP-адреса, идентификатор хоста - младшую**

# Классы IP-адресов (классы сетей)

**Каждый класс IP-адреса определяет, какая часть адреса отводится под ID-сети, а какая под ID-хоста.**

**В соответствии с классами IP-адресов различают классы сетей.**

| Класс | 8 разрядов | 8 разрядов | 8 разрядов | 8 разрядов |
|---|---|---|---|---|
| A | ID сети | ID хоста | | |
| B | ID сети | | ID хоста | |
| C | ID сети | | | ID хоста |
| D | Диапазон адресов 224.0.0.0 – 239.255.255.255 – групповая рассылка | | | |
| E | Диапазон адресов 240.0.0.0 – 247.255.255.255 - зарезервировано | | | |

Признаком, на основании которого IP-адрес относится к тому или иному классу, являются значения нескольких первых битов адреса.

| **Класс** | **Первые биты** | **Наименьший номер сети** | **Наибольший номер сети** | **Максимальное число узлов в сети** |
|---|---|---|---|---|
| A | 0 | 1.0.0.0 (0 — не используется) | 126.0.0.0 (127 — зарезервирован) | $2^{24}$, поле 3 байта |
| B | 10 | 128.0.0.0 | 191.255.0.0 | $2^{16}$, поле 2 байта |
| C | 110 | 192.0.0.0 | 223.255.255.0 | $2^{8}$, поле 1 байт |
| D | 1110 | 224.0.0.0 | 239.255.255.255 | Групповые адреса |
| E | 11110 | 240.0.0.0 | 247.255.255.255 | Зарезервировано |

# Маска подсети. Бесклассовая модель сети (CIDR)

**Рассмотрим таблицу**

Характеристики классов IP-адресов

| Класс | Значения октета *w*[1] | Октеты для ID сети | Октеты для ID хоста | Максимальное число сетей | Число хостов в сети |
|---|---|---|---|---|---|
| A | 1 126 | *w* | *x.y.z* | 126 | 16 777 214 |
| B | 128- 191 | *w.x* | *y.z* | 16 384 | 65 534 |
| C | 192–223 | *w.x.y* | *z* | 2 097 152 | 254 |

**Предположим, в локальной сети, подключаемой к Интернет, находится 2000 компьютеров. Каждому из них требуется выдать IP-адрес. Для получения необходимого адресного пространства нужны либо 8 сетей класса С, либо одна сеть класса В. Сеть класса В вмещает 65534 адреса, что много больше требуемого количества. При общем дефиците IP-адресов такое использование сетей класса В расточительно. Однако если мы будем использовать 8 сетей класса С, возникнет следующая проблема: каждая такая IP-сеть должна быть представлена отдельной строкой в таблицах маршрутов на маршрутизаторах, потому что с точки зрения маршрутизаторов — это 8 абсолютно никак не связанных между собой сетей, маршрутизация дейтаграмм в которые осуществляется независимо, хотя фактически эти IP-сети и расположены в одной физической локальной сети и маршруты к ним идентичны. Таким образом, экономя адресное пространство, мы многократно увеличиваем служебный трафик в сети и затраты по поддержанию и обработке маршрутных таблиц.**

**С другой стороны, нет никаких формальных причин проводить границу сеть-хост в IP-адресе именно по границе октета. Это было сделано исключительно для удобства представления IP-адресов и разбиения их на классы. Если выбрать длину сетевой части в 21 бит, а на номер хоста отвести, соответственно, 11 бит, мы получим сеть, адресное пространство которой содержит 2046 IP-адресов, что максимально точно соответствует поставленному требованию. Это будет *одна* сеть, определяемая своим уникальным 21-битным номером, следовательно, для ее обслуживания потребуется только *одна* запись в таблице маршрутов.**

**Единственная проблема, которую осталось решить: как определить, что на сетевую часть отведен 21 бит? В случае классовой модели старшие биты IP-адреса определяли принадлежность этого адреса к тому или иному классу и, следовательно, количество бит, отведенных на номер сети.**

**В случае адресации вне классов, с произвольным положением границы сеть-хост внутри IP-адреса, к IP-адресу прилагается 32-битовая маска, которую называют *маской сети* (netmask) или *маской подсети* (subnet mask). Сетевая маска конструируется по следующему правилу:**

- **на позициях, соответствующих номеру сети, биты установлены в “1”;**
- **на позициях, соответствующих номеру хоста, биты установлены в “0”.**

**Снабжая каждый IP-адрес маской, можно отказаться от понятий классов и сделать более гибкой систему адресации сетей хостов.**

**Пример.**
**IP-адрес: 129.64.134.5**
**Маска: 255.255.128.0**

**- адрес сети**

**- адрес хоста**

**- Часть адреса, дополненная “0”**

| 129 | | | | | | | | 64 | | | | | | | | 134 | | | | | | | | 5 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 255 | | | | | | | | 255 | | | | | | | | 128 | | | | | | | | 0 | | | | | | | |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Адрес сети | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 129 | | | | | | | | 64 | | | | | | | | 128 | | | | | | | | 0 | | | | | | | |
| Адрес хоста | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | | | | | | | | 0 | | | | | | | | 6 | | | | | | | | 5 | | | | | | | |

**Адрес сети: 129.64.128.0**
**Адрес хоста: 0.0.6.5**

# Указание маски подсети

**1. В формате десятичное с точкой (dotted decimal)**

**IP-адрес: 129.64.134.5**

**Маска: 255.255.128.0**

**2. В виде префикса сети (network prefix)**

- **Префикс – число разрядов маски, установленных в “1”**
- **Записывается в виде: /<число разрядов>**

**129.64.134.5/17**

**Маски подсетей для классов сетей:**

**Класс A 255.0.0.0 /8**

**Класс B 255.255.0.0 /16**

**Класс C 255.255.255.0 /24**

# Зарезервированные IP-адреса

1. **Адрес обратной связи (шлейфовый адрес) - 127.0.0.1**
   - Посылаемое сообщение не передается в сеть, а передается программным модулям верхнего уровня. Используется для тестирования ПО TCP/IP на локальном компьютере (сетевой адаптер не проверяется)
2. **0.0.0.0 (все нули) – неопределенный адрес**
   - Обозначает адрес узла, который сгенерировал этот пакет
3. **Адрес сети – в поле адреса хоста все “0”**
   - Позволяет адресовать всю сеть
   - Пример:
     - Адрес класса С: 195.33.19.0
4. **Групповой адрес (широковещание – broadcast) - в поле адреса хоста все “1”**
   - Пакет рассылается все хостам ЛВС, номер которой указан в поле адреса сети
   - Пример:
     - Адрес класса С: 195.33.19.255

**5. Ограниченное широковещание (limited broadcast) – все разряды адреса “1”.**

- **Пакет рассылается все хостам той же ЛВС, в которой находится хост посылающий сообщение. Ограниченность означает, что пакет не выйдет за границы данной сети**
- **Пример:**
  - **255.255.255.255**

**6. Хост в данной сети - в поле адреса сети все “0”.**

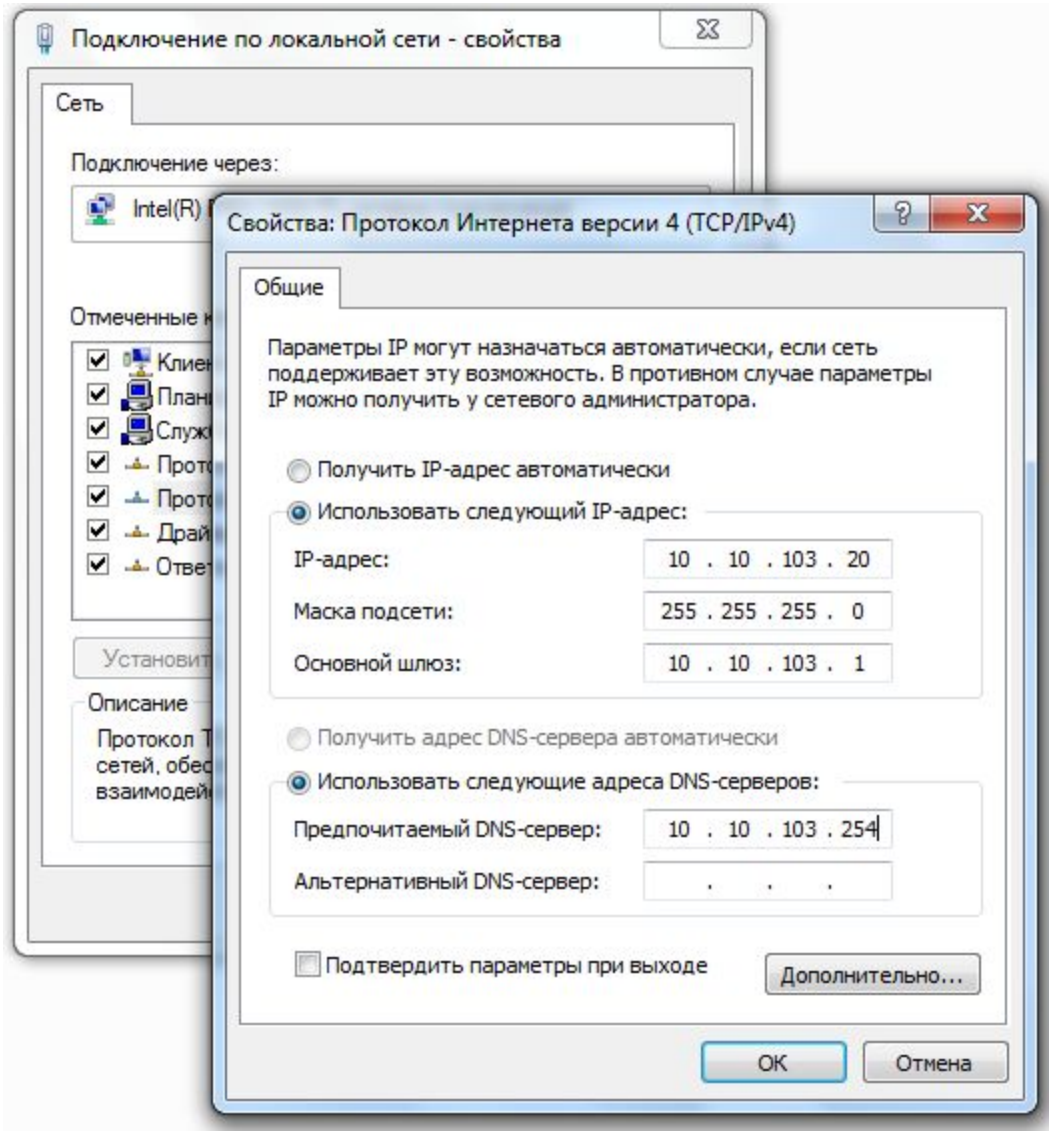
- **Хост назначения принадлежит той же сети, что и хост который отправил пакет. Может использоваться только как адрес отправителя**
- **Пример:**
  - **Адрес класса С: 0.0.0.5**

# Автономные (частные) сети

| Идентификатор сети | Маска | Количество сетей |
|---|---|---|
| 10.0.0.0 | 255.0.0.0 ( /8 ) | 1 сеть класса A |
| 172.16.0.0 – 172.31.0.0 | 255.255.0.0 ( /16 ) | 16 сетей класса B |
| 192.168. 0.0 – 192.168.255.0 | 255.255.255.0 ( /24 ) | 256 сетей класса C |

# Настройка IP-адресов.

1. **Ручная настройка.**



## 2. Автоматическая настройка

### А. Динамическое распределение IP-адресов — DHCP

**DHCP (англ. Dynamic Host Configuration Protocol — протокол динамической конфигурации узла) — это сетевой протокол, позволяющий компьютерам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP. Для этого компьютер обращается к специальному серверу, называемому сервером DHCP. Сетевой администратор может задать диапазон адресов, распределяемых среди компьютеров. Это позволяет избежать ручной настройки компьютеров сети и уменьшает количество ошибок. Протокол DHCP используется в большинстве крупных сетей TCP/IP.**

## Б. Автоматическая назначение частных IP-адресов – APIPA (Automatic Private IP Addressing)

Если попытка найти DHCP-сервер не удалась, то на компьютере с Windows DHCP-клиент сам настраивает TCP/IP на IP-адрес, который выбирается из диапазона адресов зарезервированной организацией IANA (Internet Assigned Numbers Authority) сети 169.254.0.0 класса B с маской подсети 255.255.0.0. Далее DHCP-клиент проверяет, не совпадает ли выбранный IP-адрес с одним из уже используемых, и, если совпадает, выбирает другой IP-адрес. Эта операция может повторяться до 10 раз. Подобрав незанятый адрес, DHCP-клиент настраивает на него локальный интерфейс. Параллельно клиент в фоновом режиме через каждые 5 минут проверяет, не стал ли доступен DHCP-сервер, и, если да, отменяет свои настройки и принимает те, которые предлагаются DHCP-сервером.

# Введение в IPv6

В начале 90-х годов стек протоколов TCP/IP столкнулся с серьезными проблемами. Именно в это время началось активное промышленное использование Интернета: переход к построению сетей предприятий на основе транспорта Интернета, применение веб-технологии для доступа к корпоративной информации, ведение электронной коммерции через Интернет, внедрение Интернета в индустрию развлечений (распространение видеофильмов, звукозаписей, интерактивные игры).

Все это привело к резкому росту числа узлов сети (в начале 90-х годов новый узел в Интернете появлялся каждые 30 секунд), изменению характера трафика и к ужесточению требований, предъявляемых к качеству обслуживания сетью ее пользователей.

В результате сообщество Интернета после достаточно долгого обсуждения решило подвергнуть протокол IP серьезной переработке, выбрав в качестве основных целей модернизации:

- создание масштабируемой схемы адресации;
- сокращение объема работ, выполняемых маршрутизаторами;
- предоставление гарантий качества транспортных услуг;
- обеспечение защиты данных, передаваемых по сети.

# Масштабируемая система адресации

Новая, шестая версия протокола IP (IPv6) внесла существенные изменения в систему адресации IP-сетей (RFC 2373). И, прежде всего, это коснулось *увеличения разрядности адреса.*

IPv6-адрес состоит из 128 бит, или 16 байт. Это дает возможность пронумеровать огромное количество узлов:

340 282 366 920 938 463 463 374 607 431 762 211 456.

Вместо прежних двух уровней иерархии адреса (номер сети и номер узла) в IPv6 имеется 4 уровня, из которых три уровня используются для идентификации сетей, а один — для идентификации узлов сети. За счет увеличения числа уровней иерархии в адресе новый протокол эффективно поддерживает технологию CIDR. Благодаря этому, а также усовершенствованной системе групповой адресации и введению нового типов адресов новая версия IP позволяет *снизить затраты на маршрутизацию.*

Произошли и чисто внешние изменения — разработчики стандарта предложили использовать вместо десятичной *шестнадцатеричную* форму записи IP-адреса. Каждые четыре шестнадцатеричные цифры отделяются друг от друга двоеточием. Вот как, например, может выглядеть адрес IPv6:

FEDC:0A98:0:0:0:0:7654:3210.

Если в адресе имеется длинная последовательность нулей, то запись адреса можно сократить. Например, приведенный выше адрес можно записать и так:

FEDC:0A98::7654:3210.

Сокращение в виде двух двоеточий (::) может употребляться в адресе только один раз. Можно также опускать незначащие нули в начале каждого поля адреса, например, вместо FEDC:0A98::7654:3210 можно писать FEDC:A98::7654:3210.

Для сетей, поддерживающих обе версии протокола (IPv4 и IPv6), разрешается использовать для младших 4 байт традиционную для IPv4 десятичную запись: 0:0:0:0:0:FFFF:129.144.52,38 или ::FFFF:129.144.52.38.

# TCP- и UDP-порты

**В TCP/IP-сетях порт – это механизм, позволяющий компьютеру поддерживать сразу несколько коммуникационных сеансов с программами и другими компьютерами.**

**В сети многие приложения могут одновременно взаимодействовать друг с другом. Когда эти приложения функционируют на одном сетевом хосте для протокола TCP/IP требуется метод, позволяющий различать эти приложения.**

**Для этой цели, т.е. для задания нужного приложения, в протоколе TCP/IP используются порты.**

**Порт является идентификатором приложения на компьютере.**

**Порт связан с протоколами TCP или UDP транспортного уровня и называется соответственно портом TCP или UDP.**

**Порт может задаваться любым числом, находящемся в диапазоне от 0 до 65535.**

**Для портов наиболее распространенных сетевых приложений используются хорошо известные зарезервированные номера портов, значения которых меньше 1024.**

## Некоторые хорошо известные (well-known) TCP-порты

| TCP Port Number | Description |
|---|---|
| 20 | FTP (Data Channel) |
| 21 | FTP (Control Channel) |
| 23 | Telnet |
| 80 | HyperText Transfer Protocol (HTTP) used for the World Wide Web |

## Некоторые хорошо известные (well-known) UDP-порты

| UDP Port Number | Description |
|---|---|
| 53 | Domain Name System (DNS) |
| 69 | Trivial File Transfer Protocol (TFTP) |
| 137 | NetBIOS name service |
| 161 | Simple Network Management Protocol (SNMP) |

**В ОС для идентификации приложений работающих на конкретных хостах используются сокеты – Sockets.**

**Сокет представляет собой комбинацию IP-адреса и порта TCP или UDP.**

**Приложение создает сокет, указывая IP-адрес и порт, отслеживаемый приложением.**

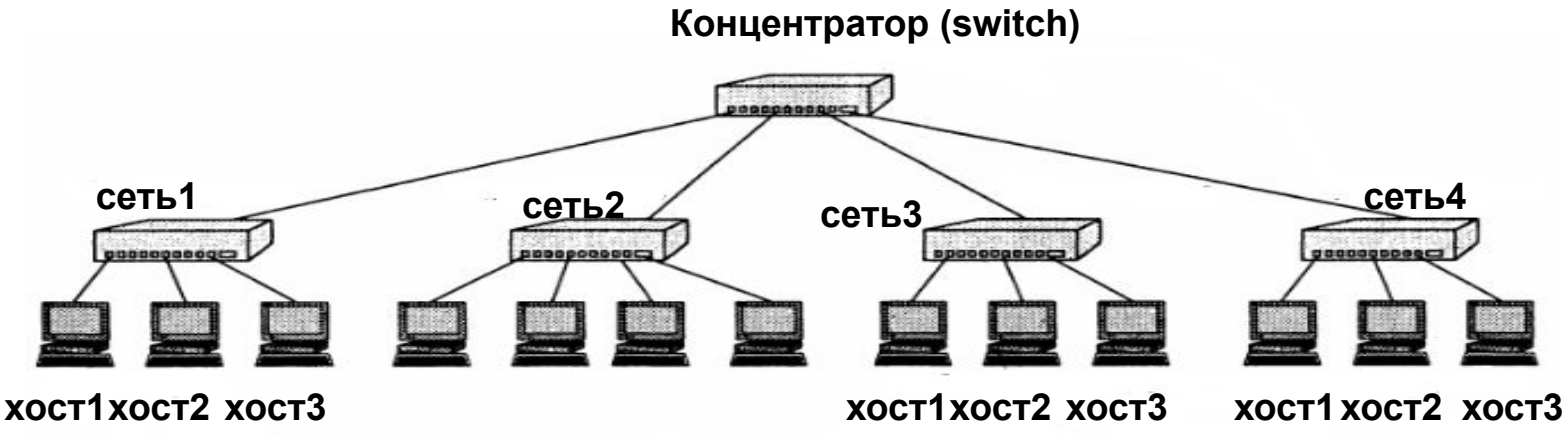
**IP-адрес в сокете позволяет идентифицировать и определять конечный компьютер, а порт указывает приложение, которому будут посланы данные.**

# Подсети. Деление сетей на подсети.

**Подсеть (subnet) – физический сегмент TCP/IP сети, в котором используются IP-адреса с общим идентификатором сети.**

**Использование подсетей имеет ряд преимуществ:**

- **Совместное использование различных сетевых технологий в разных подсетях (Ethernet, Token Ring)**
- **Уменьшение нагрузки на сеть путем уменьшения числа широковещательных запросов.**



**Чтобы разделить сеть на несколько подсетей необходимо использовать различные идентификаторы подсети для каждого сегмента.**

**Для этого требуется разбить идентификатор узла общей сети на две группы разрядов (бит)**

- **первая служит для идентификации подсети (S)**
- **вторая для идентификации конкретного хоста в подсети (H)**

<table>
<tr><td>ID сети общей сети</td><td colspan="2">ID хоста общей сети</td></tr>
<tr><td></td><td>S-идентификатор подсети</td><td>H-идентификатор хоста в подсети</td></tr>
<tr><td colspan="2">ID подсети</td><td>ID хоста в подсети</td></tr>
</table>

# Алгоритм разделения сети на подсети

## I. Определение маски подсети

**1. Необходимо определить число разрядов в идентификаторе подсети – S.**

**Используется следующее правило:**

**$N = 2^S$**

**N – число подсетей в общей сети**

**S – число разрядов в идентификаторе подсети**

**Делим сеть на две подсети:**

**$2=2^1$ => S = 1**

**2. Запишите S единиц подряд и добавьте справа столько 0, чтобы общее количество разрядов соответствовало разрядности идентификатора хоста общей сети (для сети класса A – 8 р; для сети класса B – 16 р и т.д.)**

- **Сеть класса A: 10000000 => 128**
- **Сеть класса B: 10000000.00000000 => 128.0**

**3. Запишите полученное число на месте ID-хоста в маске, определяющей общую сеть.**

- **Сеть класса A: 255.255.255.128 => /25**
- **Сеть класса B: 255.255.128.0 => /17**

# II. Определение адресов подсетей

**Адрес сети (подсети) – адрес в котором ID-хоста заменяется нулями.**

**Для задания идентификаторов подсетей используется то же число разрядов S, что и для соответствующей маски**

**Запишите все двоичные числа, образованные изменением S разрядов и добавьте справа столько 0, чтобы общее количество разрядов соответствовало числу разрядов в ID-хоста маски подсети. Запишите полученное число на месте ID-хоста в адресе общей сети.**

**Сеть класса A: 0 0000000 => 0**

**1 0000000 => 128**

**Подсети: W.X.Y.0**

**W.X.Y.128**

**Сеть класса B: 0 0000000.00000000 => 0.0**

**1 0000000.00000000 => 128.0**

**Подсети: W.X.0.0**

**W.X.128.0**

## III. Определение диапазонов адресов для узлов подсети

**Начало диапазона – увеличенный на 1 адрес подсети**

**Конец диапазона – уменьшенный на 2 адрес следующей возможной подсети (все 0 в адресе хоста – адрес сети; все 1 в адресе хоста – широковещание)**

**Сеть класса A: W.X.Y.1 - W.X.Y.126**
**W.X.Y.129 - W.X.Y.254**

**Сеть класса B: W.X.0.1 - W.X.127.254**
**W.X.128.1 - W.X.255.254**

## IV. Определение количества хостов подсети

**1. Запишите маску подсети и определите число разрядов в идентификаторе узла – H**

**Сеть класса A: 255.255.255.128 => /25 => H = 7**

**Сеть класса B: 255.255.128.0 => /17 => H = 15**

**2. Число возможных адресов – $2^H$**

**3. Число доступных адресов – $2^H$-2 (все 0 в адресе хоста – адрес сети; все 1 в адресе хоста – широковещание)**

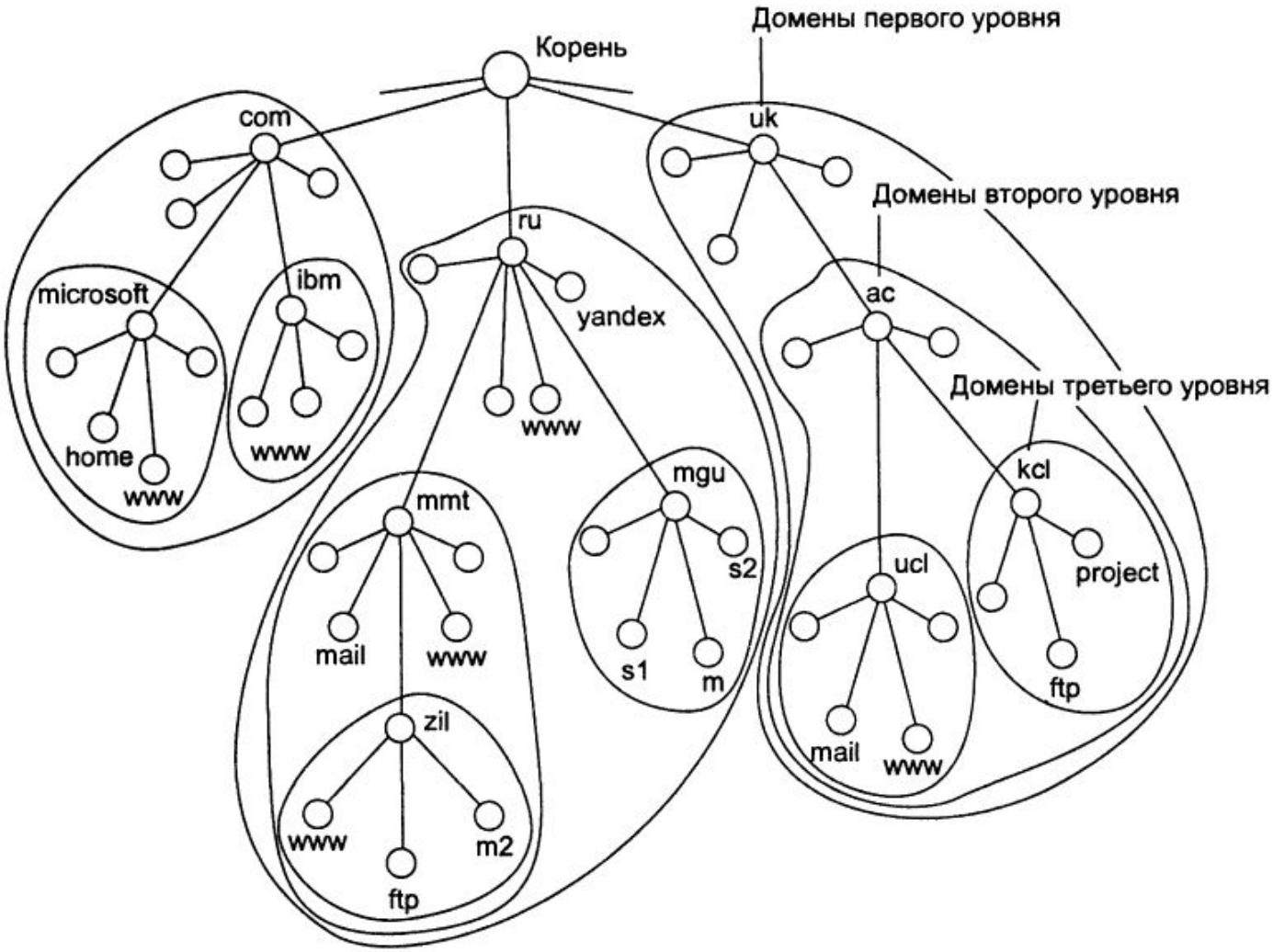
# Адресация хостов в сети.

**Каждый компьютер в сети имеет уникальное имя.**

1. **В IP-сетях в качестве уникального имени хоста используется IP-адрес.**
2. **Локальный (аппаратный) адрес или MAC-адрес.**
   - **Адресс, присвоенный сетевому адаптеру на заводе изготовителе. (MAC – Media Access Control – управление доступом к среде). Слово локальный означает «действующий не во всей составной сети, а лишь в пределах локальной сети (подсети). Внутри ЛВС хосты устанавливают связь друг с другом, используя эти адреса (канальный уровень OSI).**

## 3. DNS-имя.

– **DNS (Domain Name System) – доменная система имен. Реализуется в виде иерархического пространства имен, в котором имя представляет собой последовательность простых символьных имен, разделенных точками.**



**DNS (Domain Name System) – это распределенная база данных, которая распределена между специальными компьютерами сети – DNS-серверами.**

**Домен – группа сетевых хостов, имеющая уникальное имя.**

**DNS-сервер создается в каждом домене.**

**DNS-сервер хранит доменные имена и соответствующие им IP-адреса.**

**4. NetBIOS-имя (плоское имя).**

**Имя, присваиваемое компьютеру внутри локальной сети. Состоит из последовательности символов, не разделенных на части.**

**Используется только для связи внутри локальной сети (например, для доступа к общим каталогам и принтерам).**

# Разрешение имен.

**Разрешение имени – установление однозначного соответствия (отображение) между именами разного типа.**

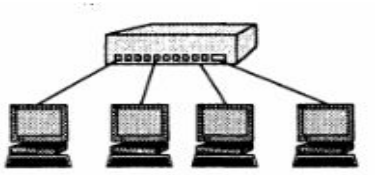
**1. Отображение IP-адресов на MAC-адреса.**

**Каждый сетевой интерфейс (сетевой адаптер) имеет IP-адрес и MAC-адрес.**

**Для определения MAC-адреса по IP-адресу используется протокол ARP (Address Resolution Protocol).**

**Протокол ARP поддерживает для каждого сетевого адаптера или маршрутизатора таблицу ARP. Первоначально, при включении компьютера или маршрутизатора в сеть, все его таблицы маршрутизации пусты. В них накапливается информация в ходе работы сети.**

| IP-адрес | MAC-адрес | Тип записи |
|---|---|---|
| 194.85.135.65 | 00E0F77F1920 | Динамический |
| 194.85.135.75 | 008048EB7E60 | Динамический |
| 194.85.60.21 | 008048EB7567 | Статический |



A B

**Пусть IP-протокол узла А направляет пакет узлу В с адресом IP1.**

**Для решения этой задачи:**

- **Протокол IP обращается к протоколу APR**
  - **«Какой MAC-адрес имеет узел с адресом IP1?»**
- **Работа ARP начинается с просмотра APR-таблицы. Предположим, что в ней отсутствует запись об адресе IP1**
- **Протокол ARP формирует ARP-запрос и рассылает в сеть всем хостам (широковещание) сети.**
- **Хосты направляют запрос своему протоколу ARP. Он сравнивает полученный в запросе адрес IP1 со своим IP-адресом.**
- **ARP, который констатировал совпадение, формирует ARP-ответ, в котором указывает свой MAC-адрес. Широковещания здесь нет, т.к. в ARP-запросе был указан MAC-адрес отправителя.**

**Зона ARP-запросов ограничивается локальной сетью, т.к. маршрутизаторы не передают эти запросы в другие сети.**

# Утилита “ARP”

**Отображение и изменение таблиц преобразования IP-адресов в физические, используемые протоколом разрешения адресов (ARP).**

- **ARP -s inet_addr eth_addr [if_addr]**
- **ARP -d inet_addr [if_addr]**
- **ARP -a [inet_addr] [-N if_addr]**
- **-a Отображает текущие ARP-записи, опрашивая текущие данные протокола. Если задан inet_addr, то будут отображены IP и и физический адреса только для заданного компьютера. Если более одного сетевого интерфейса используют ARP, то будут отображаться записи для каждой таблицы.**
- **-g То же, что и ключ -a.**
- **inet_addr Определяет IP-адрес.**
- **-N if_addr Отображает ARP-записи для заданного в if_addr сетевого интерфейса.**
- **-d Удаляет узел, задаваемый inet_addr. inet_addr может содержать символ шаблона * для удаления всех узлов.**
- **-s Добавляет узел и связывает internet адрес inet_addr с физическим адресом eth_addr. Физический адрес задается 6 байтами (в шестнадцатеричном виде), разделенных дефисом. Эта связь является постоянной.**
- **eth_addr Определяет физический адрес.**
- **if_addr Если параметр задан, - он определяет интернет адрес интерфейса, чья таблица преобразования адресов должна измениться. Если не задан, - будет использован первый доступный интерфейс.**
- **Пример:**
- **arp -a ... Выводим ARP-таблицу.**

# Разрешение NetBIOS-имен

**Некоторые приложения, работающие в ОС используют плоские NetBIOS-имена для связи с приложениями на удаленных компьютерах. Пример – служба доступа к файлам и принтерам сетей Microsoft. Так как в IP-сети устройства адресуются IP-адресами, должен существовать механизм разрешения NetBIOS-имени в IP-адреса.**

**Это обеспечивает протокол NetBT – NetBIOS поверх TCP/IP.**

**NetBT создает локальный кэш в котором хранятся записи в виде пар: NetBIOS-имя – IP-адрес.**

**Таблица удаленного буфера NetBIOS-имен**

| Имя | Тип | Адрес узла | Время жизни [с] |
|---|---|---|---|
| VERA-FLORENCE | <00> Уникальный | 192.168.0.11 | 430 |
| VERA-FLORENCE | <20> Уникальный | 192.168.0.11 | 430 |

**Имена NetBIOS:**

- **Уникальные – идентификация конкретного компьютера**
- **Групповые – группа компьютеров (домен или рабочая группа)**

**Каждому NetBIOS-имени присваивается специальный идентификатор, который определяет тип имени:**

**[00] – имя сервиса клиента**

**[20] – имя сервиса сервера**

**[03] – имя пользователя**

**[1C] – имя контроллера домена**

**[1E] – имя рабочей группы**

**Основной метод разрешения NetBIOS-имен для протокола NetBT– широковещание (аналогично ARP).**

**Кроме этого можно использовать файл LMHOSTS - текстовый файл, содержащий записи типа:**

**IP-адрес – NetBIOS-имя**

**Файл можно редактировать вручную.**

# Утилита “nbtstat”

**Отображение статистики протокола и текущих подключений TCP/IP с помощью NBT (NetBIOS через TCP/IP).**

**NBTSTAT [-a Узел] [-A IP-адрес] [-c] [-n]**
**[-r] [-R] [-RR] [-s] [-S] [интервал] ]**

**-a (adapter status) Вывод таблицы имен узла, указанного по имени.**
**-A (Adapter status) Вывод таблицы имен узла, указанного по IP-адресу.**
**-c (cache) Вывод буфера имен удаленных узлов, включая адреса IP.**
**-n (names) Вывод локальных имен NetBIOS.**
**-r (resolved) Вывод имен, определенных с помощью рассылки и WINS.**
**-R (Reload) Очистка и перезагрузка таблицы удаленного буфера имен.**
**-S (Sessions) Вывод таблицы сеансов с IP-адресами.**
**-s (sessions) Вывод таблицы сеансов с преобразованием IP-адресов в имена NETBIOS.**
**-RR (ReleaseRefresh) Отсылка пакетов освобождения имени (Name Release)на WINS-сервер, а затем запуск обновления (Refresh)**
**Узел Имя удаленного компьютера.**
**IP-адрес IP-адрес удаленного компьютера.**
**интервал Повторный вывод статистических данных через указанный интервал в секундах. Для прекращения вывода нажмите клавиши <Ctrl>+<C>.**

# Разрешение DNS-имен

**1. <u>Использование текстового файла HOSTS</u> – содержит записи типа:**

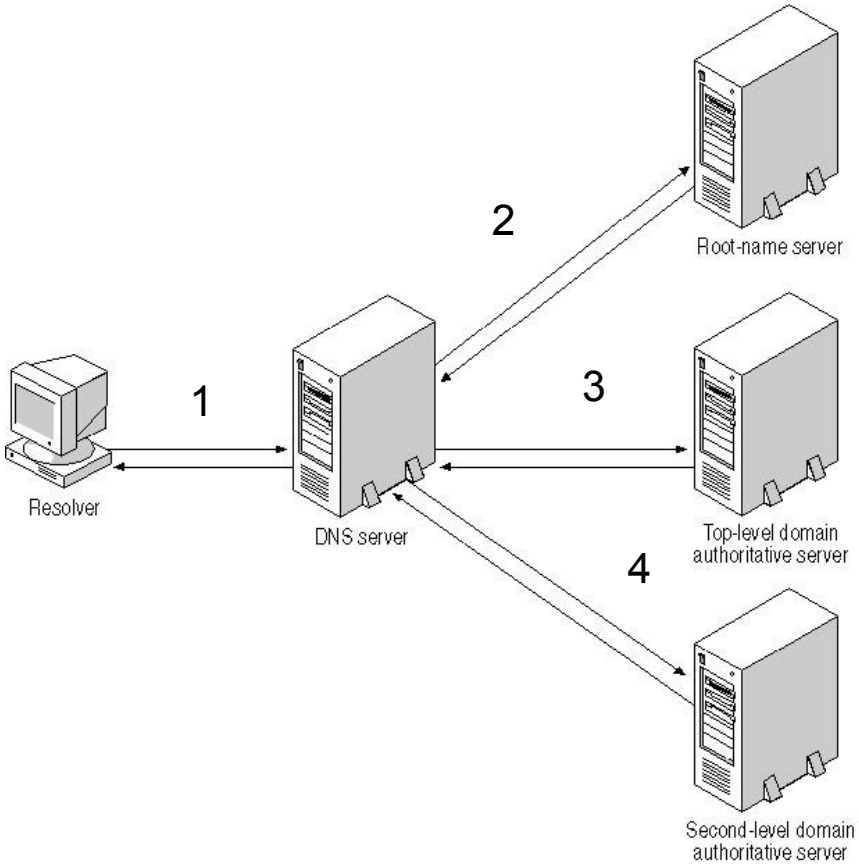
**IP-адрес – DNS-имя**

**Файл можно редактировать вручную.**

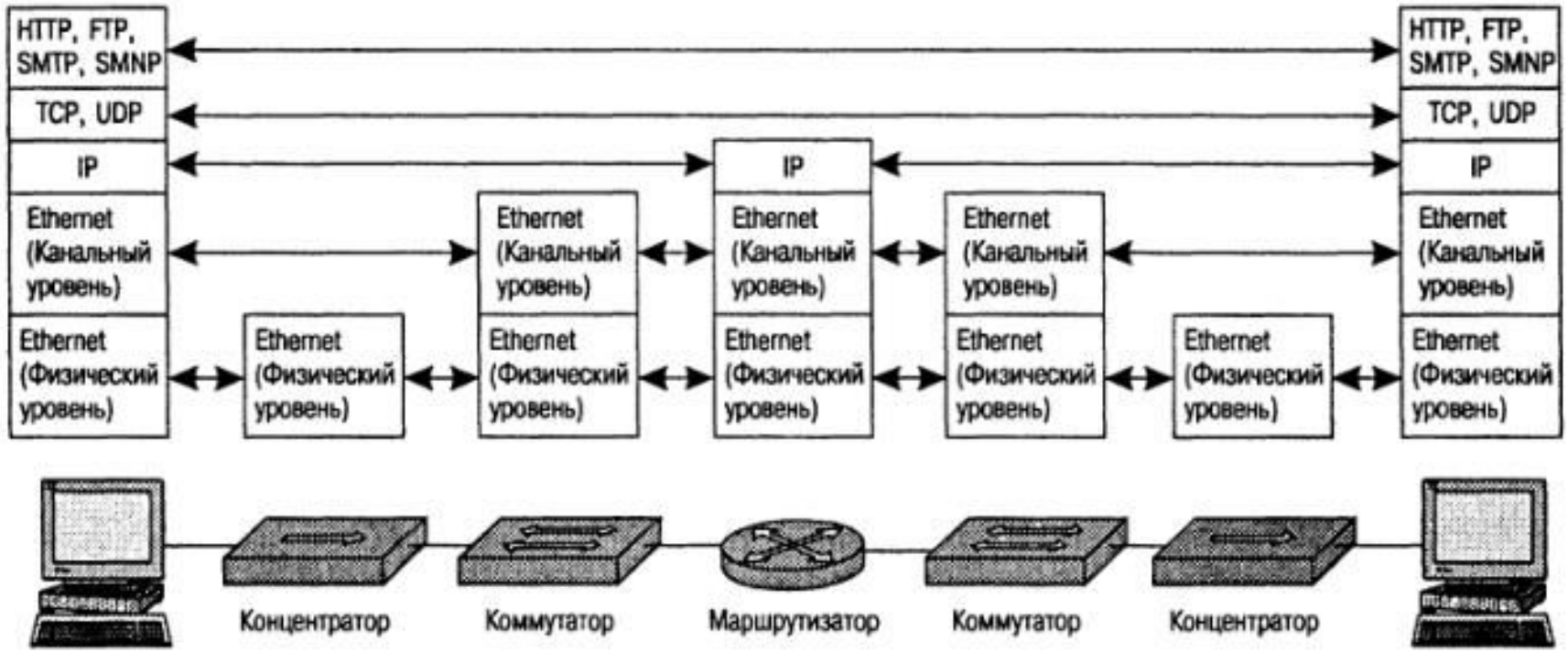
**2. <u>Использование DNS-сервера.</u>**

**DNS-сервер создается в каждом домене.**

**DNS-сервер хранит доменные имена и соответствующие им IP-адреса.**

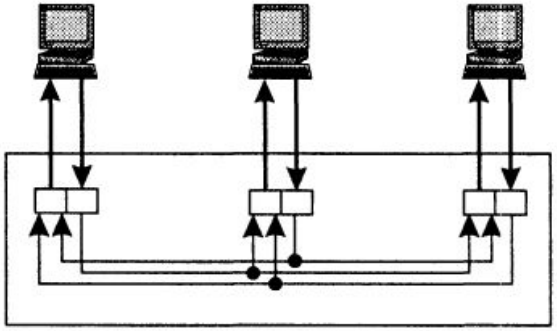


# Концентраторы, коммутаторы, маршрутизаторы

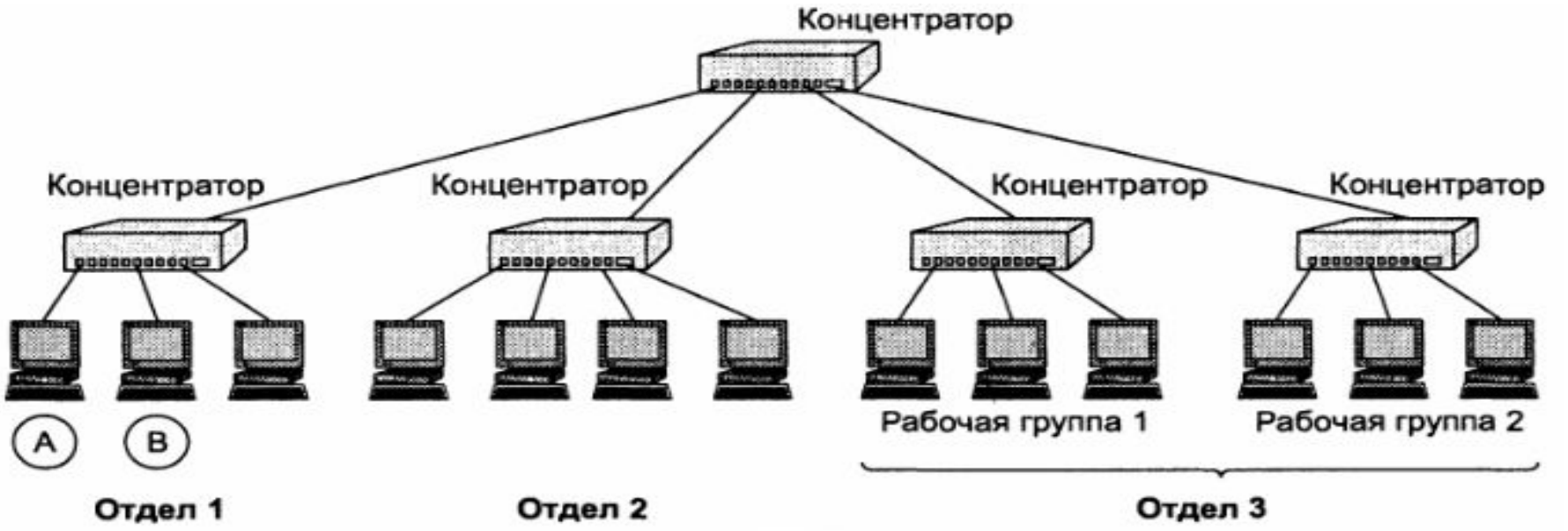



Соответствие функций различных устройств сети уровням модели OSI

# Концентратор (HUB)



**Концентратор работает на физическом уровне сетевой модели OSI, повторяет приходящий на один порт сигнал на все активные порты. (сигнал, приходящий на вход воспринимается концентраторм как неструктурированный поток “0” и “1”). В случае поступления сигнала на два и более порта одновременно возникает коллизия, и передаваемые кадры данных теряются. Таким образом, все подключенные к концентратору устройства находятся в одном домене коллизий.**

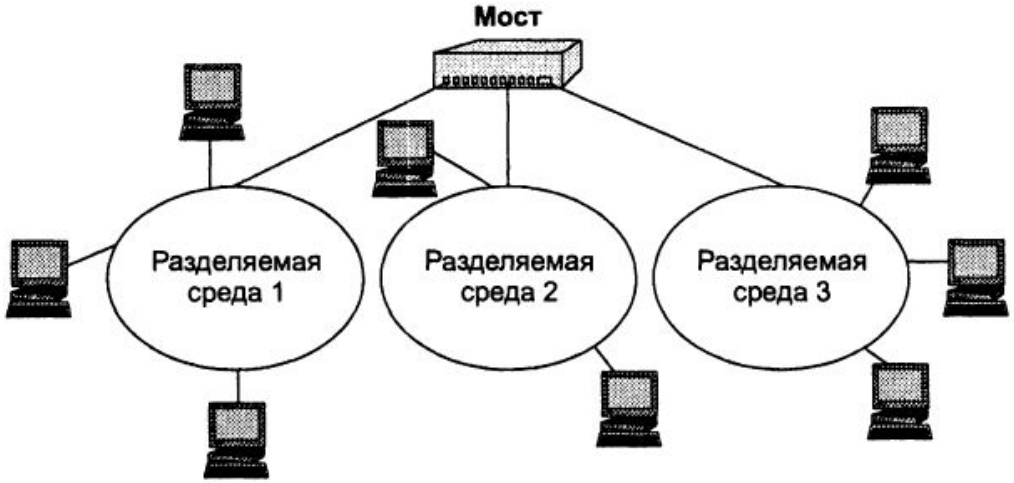


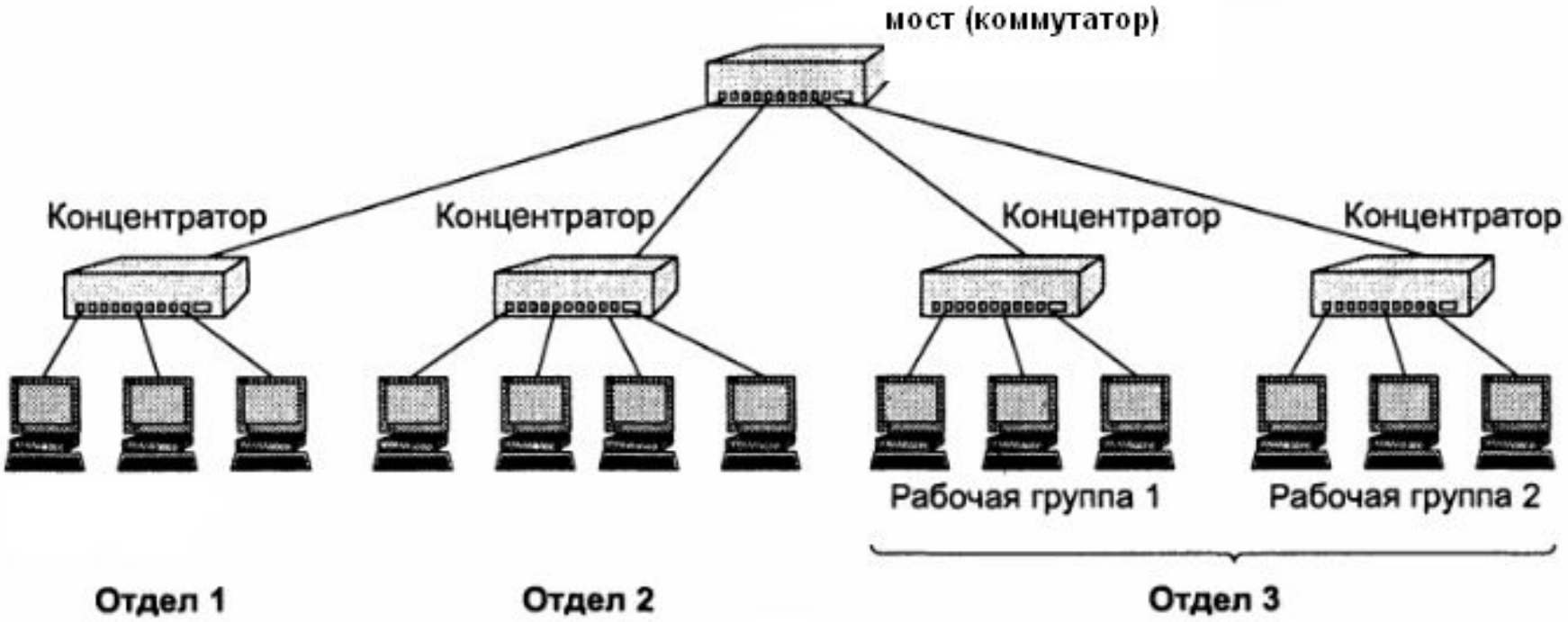

**Локальная сеть:**

- **Один домен коллизий**

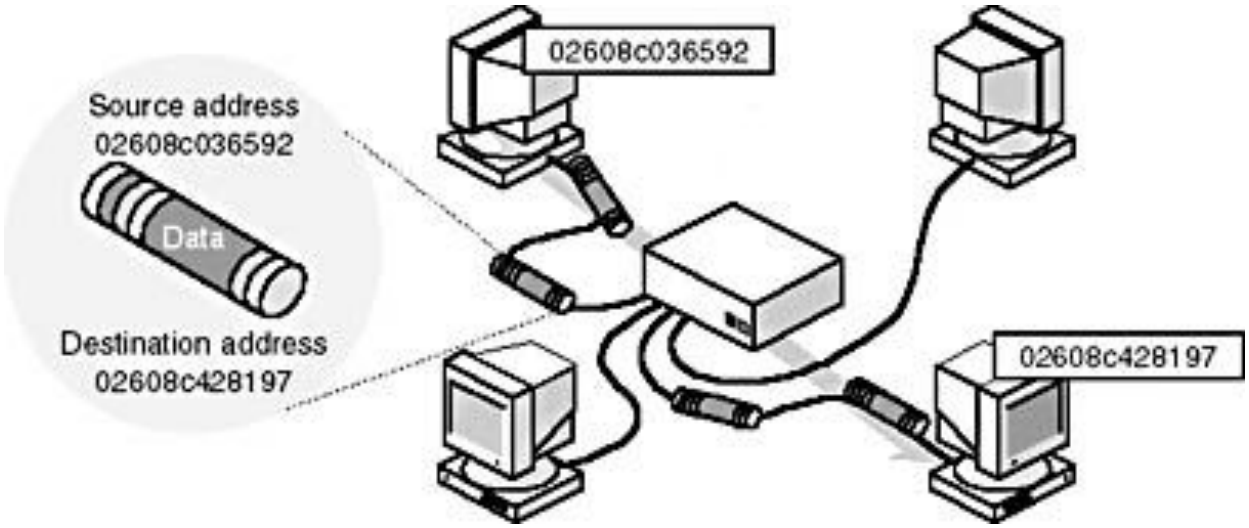
# Мост (bridge), коммутатор(switch)

**Мост** (bridge) делит единую среду передачи на части (часто называемые **логическими сегментами**), передавая информацию из одного сегмента в другой только в том случае, если такая передача действительно необходима, то есть если адрес компьютера назначения принадлежит другому сегменту (рис. 3.19). Тем самым мост изолирует трафик одного сегмента от трафика другого, повышая общую производительность сети.


мост (коммутатор)
Концентратор
Концентратор
Концентратор
Концентратор
Рабочая группа 1
Рабочая группа 2
Отдел 1
Отдел 2
Отдел 3

**Мост работает на канальном уровне модели OSI – использует для локализации трафика MAC-адреса компьютеров.**




**Во время работы мост строит таблицу маршрутизации. В начале работы эта таблица пуста. При работе сети мост извлекает из поступивших кадров MAC-адреса отправителей и помещает их таблицу маршрутизации, где фиксирует номер своего порта и поступивший адрес**

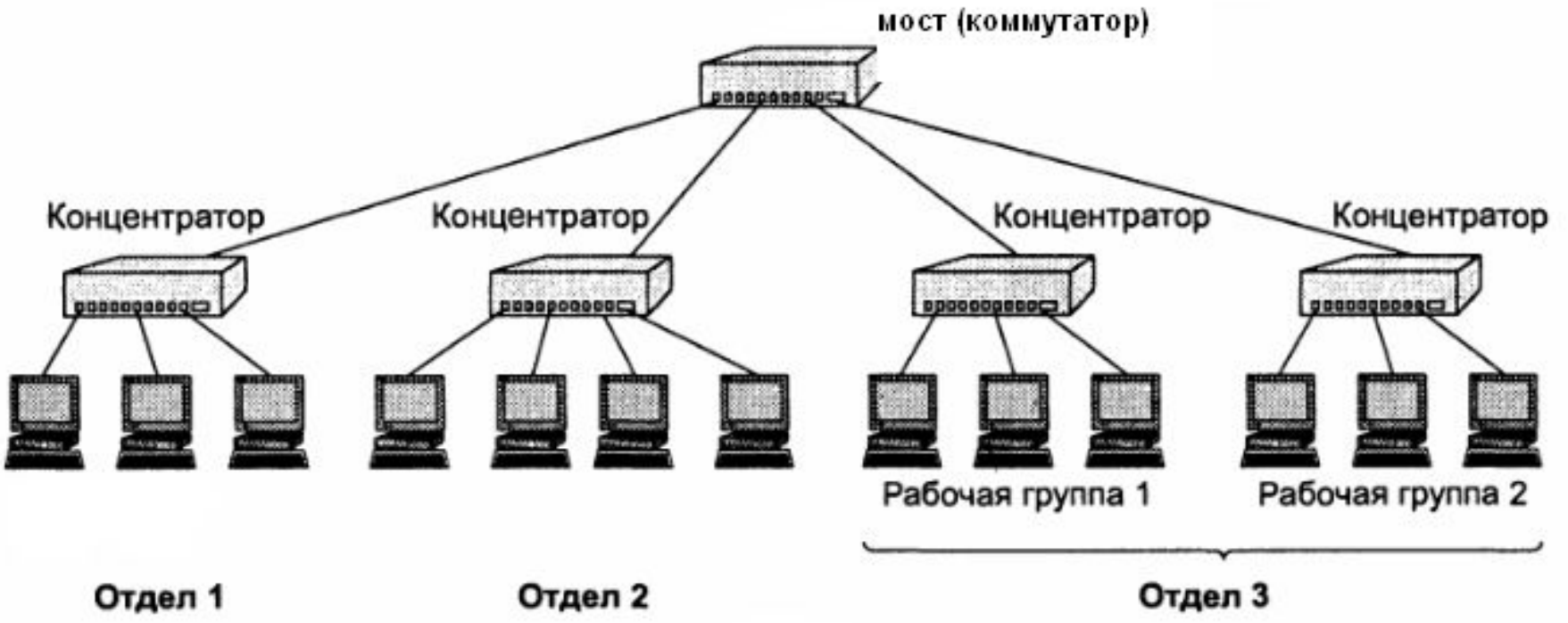
| Номер порта | MAC-адрес |
|---|---|
| 1 | 02608C036592 |
| 4 | 02608C428197 |

**В дальнейшем мост передает кадр только в тот порт, где находится получатель**

**Коммутатор** (switch) функционально подобен мосту и отличается от моста в основном более высокой производительностью. Каждый интерфейс коммутатора оснащен специализированным процессором, который обрабатывает кадры по алгоритму моста независимо от процессоров других портов. За счет этого общая производительность коммутатора обычно намного выше производительности традиционного моста, имеющего один процессорный блок. Можно сказать, что коммутаторы — это усовершенствованные мосты, которые обрабатывают кадры в параллельном режиме. Когда стало экономически оправданно использовать отдельные специализированные процессоры на каждом порту коммуникационного устройства, коммутаторы локальных сетей полностью вытеснили мосты.

**Все компьютеры подключенные к мосту (коммутатору) образуют один широковещательный домен. Мост передает широковещательные сообщения (например, запросы ARP) во все свои порты. Именно принадлежность к одному широковещательному домену позволяет сегментам сети, подключенным к разным портам оставаться одной и той же ЛВС.**

**При этом каждый сегмент сети представляет собой отдельный домен коллизий.**




**Локальная сеть:**

- **Один широковещательный домен**
- **Четыре домена коллизий**

# Маршрутизатор (router)

**Маршрутизация – процесс выбора пути для передачи пакетов**

**Маршрутизатор (router) – это устройство, подключенное к двум или нескольким сетям. Обеспечивает маршрутизацию. Работает на сетевом уровне модели OSI.**

**Например, Ethernet и TokenRing.**

**В большой составной сети к одному маршрутизатору может подключаться несколько ЛВС, а к одной ЛВС может подключаться несколько маршрутизаторов. Поэтому к одной и той же цели пакеты могут доставляться различными путями. Если один маршрутизатор выйдет из строя, пакеты в обход его дойдут до места назначения.**

**В сложных сетях перед маршрутизатором стоит важная задача – выбор наиболее эффективного пути для доставки пакетов.**

**Если пакету на пути к конечному пункту приходится проходить через множество сетей, каждый обрабатывающий его маршрутизатор называют транзитом (hop). Маршрутизатор часто оценивает эффективность маршрута по числу транзитов от исходной до целевой системы.**

# Таблица маршрутизации

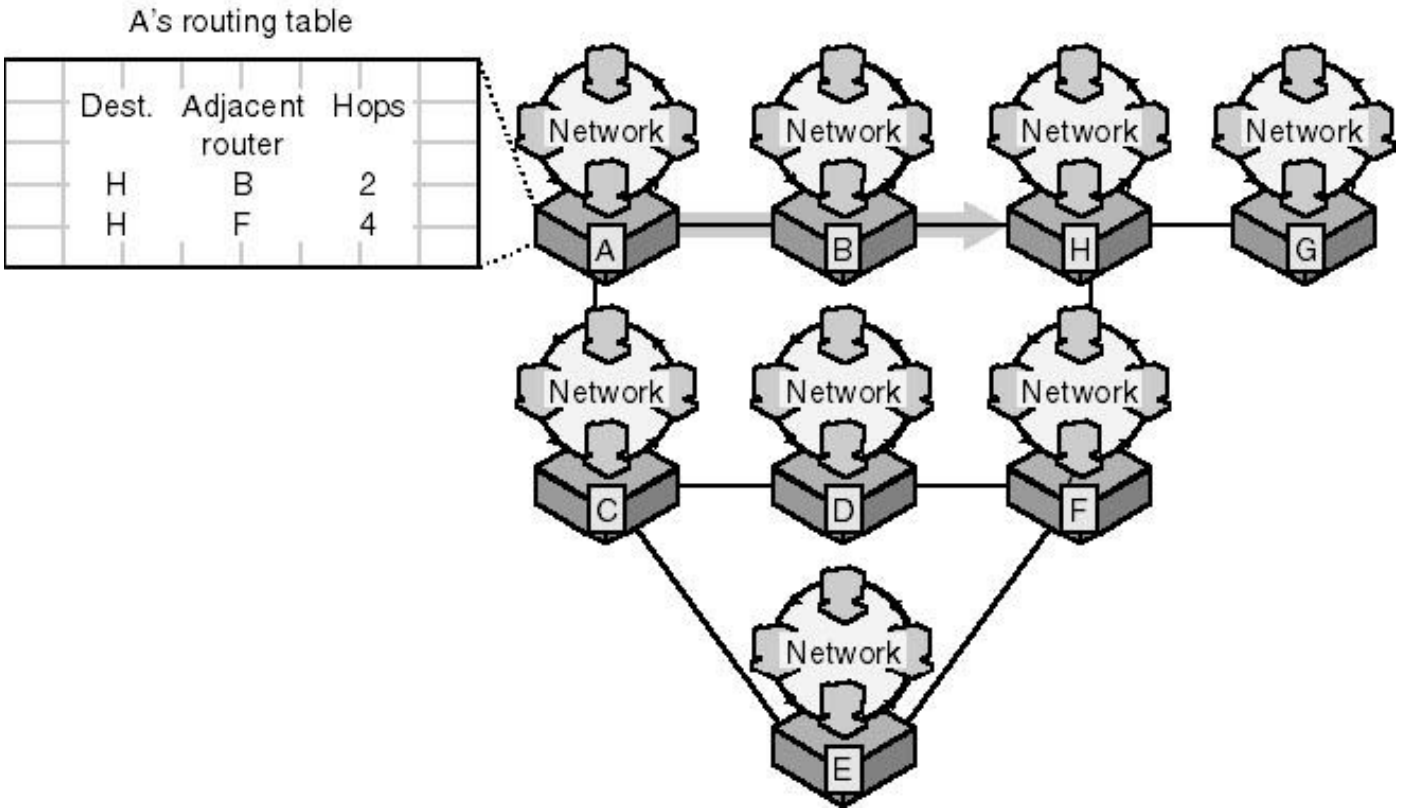



**Таблица маршрутизации – это сердце маршрутизатора. Без нее маршрутизатор не узнает, куда пересылать получаемые пакеты.**

**В отличие от коммутаторов маршрутизаторы не умеют составлять таблицы на основе информации из получаемых пакетов – ее там нет.**

**Два способа создания таблиц маршрутизации:**

**Статическая маршрутизация – создание таблицы вручную**

**Динамическая маршрутизация – маршрутизаторы с помощью специальных протоколов обмениваются информацией друг о друге и сетях, к которым они подключены.**

**По сути таблица маршрутизации представляет собой список сетей и адресов маршрутизаторов, к которым система должна обращаться для передачи данных в эти сети.**

| Сетевой адрес | Маска | Адрес шлюза | Интерфейс | Метрика |
|---|---|---|---|---|
| 127.0.0.0 | 255.0.0.0 | 127.0.0.1 | 127.0.0.1 | 1 |
| 0.0.0.0 | 0.0.0.0 | 198.21.17.7 | 198.21.17.5 | 1 |
| 56.0.0.0 | 255.0.0.0 | 213.34.12.4 | 213.34.12.3 | 15 |
| 116.0.0.0 | 255.0.0.0 | 213.34.12.4 | 213.34.12.3 | 13 |
| 129.13.0.0 | 255.255.0.0 | 198.21.17.6 | 198.21.17.5 | 2 |
| 198.21.17.0 | 255.255.255.0 | 198.21.17.5 | 198.21.17.5 | 1 |
| 198.21.17.5 | 255.255.255.255 | 127.0.0.1 | 127.0.0.1 | 1 |

# Пример таблицы маршрутизации на локальном хосте в ОС Windows

Активные маршруты:

| Сетевой адрес | Маска сети | Адрес шлюза | Интерфейс | Метрика |
|---|---|---|---|---|
| 0.0.0.0 | 0.0.0.0 | 192.168.0.1 | 192.168.0.175 | 20 |
| 127.0.0.0 | 255.0.0.0 | 127.0.0.1 | 127.0.0.1 | 1 |
| 192.168.0.0 | 255.255.255.0 | 192.168.0.175 | 192.168.0.175 | 20 |
| 192.168.0.175 | 255.255.255.255 | 127.0.0.1 | 127.0.0.1 | 20 |
| 192.168.0.255 | 255.255.255.255 | 192.168.0.175 | 192.168.0.175 | 20 |
| 224.0.0.0 | 240.0.0.0 | 192.168.0.175 | 192.168.0.175 | 20 |
| 255.255.255.255 | 255.255.255.255 | 192.168.0.175 | 192.168.0.175 | 1 |

**Столбцы:**

**Сетевой адрес(network address) – адрес пункта назначения (адрес сети или хоста, информация о маршруте к которым записана в других столбцах**

**Маска сети (network mask) – маска подсети для адреса в первом столбце**

**Адрес шлюза (gateway address) – адрес маршрутизатора, которому необходимо посылать пакет, чтобы доставить его хосту с адресом из первого столбца**

**Интерфейс (interface) – адрес сетевого адаптера, через который необходимо передавать пакеты маршрутизатору, адрес которого указан в столбце «адрес шлюза».**

**Метрика (metric) – число, позволяющее сравнить относительную эффективность различных путей к одной цели (фактически показывает сколько маршрутизаторов надо пройти, чтобы добраться до цели).**

**Строки:**

**(1) 0.0.0.0 0.0.0.0 192.168.0.1 192.168.0.175 20 – шлюз по умолчанию (default gateway) – маршрут к любой сети, не описанной в таблице маршрутизации. Любой IP-адрес объединенный с маской 0.0.0.0 дает 0.0.0.0 . На компьютере, не являющимся маршрутизатором в столбце «Адрес шлюза» указывается его IP-адрес, а столбец «Интерфейс» указан IP-адрес сетевого интерфейса (адаптера), соединяющего систему с сетью.**

**(2) 127.0.0.0 255.0.0.0 127.0.0.1 127.0.0.1 1 – адрес обратной связи (локальной заглушки). Протокол IP автоматически направляет все пакеты, адресованные на любой адрес сети 127.0.0.0, обратно в очередь входящих пакетов. В качестве маршрутизатора используется свой собственный обратный адрес 127.0.0.1**

**(3) 192.168.0.0 255.255.255.0 192.168.0.175 192.168.0.175 20 – маршрут к ЛВС, в которую включен локальный компьютер. В столбцах «Адрес шлюза» и «Интерфейс» указывается IP-адрес локального компьютера, указывающий что в качестве маршрутизатора он должен использовать самого себя.**

**(4) 192.168.0.175 255.255.255.255 127.0.0.1 127.0.0.1 20 – маршрут к локальному компьютеру (ссылка на этот компьютер). Управление пакетом направленным к локальному компьютеру должно выполнятся внутри него (127.0.0.1), в обход сетевого адаптера.**

**( (5) 192.168.0.255 255.255.255.255 192.168.0.175 192.168.0.175 20 – адрес отправки широковещательных сообщений в локальной сети (192.168.0.255). Пакеты перелаются компьютерам локальной сети, поэтому система использует в качестве маршрутизатора саму себя (192.168.0.175).**

**(6) 224.0.0.0 240.0.0.0 192.168.0.175 192.168.0.175 20 – адрес групповой рассылки. Система использует в качестве маршрутизатора саму себя.**

**(7) 255.255.255.255 255.255.255.255 192.168.0.175 192.168.0.175 1 – адрес широковещательных сообщений по все сети**

# Команда “route”

**Просмотр и изменение таблицы маршрутизации**

**ROUTE [-f] [-p] [команда [узел]**
**[MASK маска] [шлюз] [METRIC метрика] [IF-интерфейс]**

**-f** Очистка таблиц маршрутов от записей для всех шлюзов. При указании одной из команд, таблицы очищаются до выполнения команды.

**-p** При использовании с командой ADD задает сохранение маршрута при перезагрузке системы. По умолчанию маршруты не сохраняются при перезагрузке. Игнорируется для остальных команд, изменяющих соответствующие постоянные маршруты. Этот параметр не поддерживается в Windows 95.

**команда** Одна из четырех команд
- **PRINT** Печать маршрута
- **ADD** Добавление маршрута
- **DELETE** Удаление маршрута
- **CHANGE** Изменение существующего маршрута

**узел** Адресуемый узел.

**MASK** Если вводится ключевое слово MASK, то следующий параметр интерпретируется как параметр "маска".

**маска** Значение маски подсети, связываемое с записью для данного маршрута. Если этот параметр не задан, по умолчанию подразумевается 255.255.255.255.

**шлюз** Шлюз.

**METRIC** Определение параметра метрика/цена для адресуемого узла.

**интерфейс** адрес сетевого адаптера, который система должна использовать для передачи данных маршрутизатору, адрес которого указан в поле шлюз

# Команда “ping”

**Проверка связи с хостом.**

**Использование: ping [-t] [-a] [-n число] [-l размер] [-f] [-i TTL] [-v TOS]**
**[-r число] [-s число] [[-j списокУзлов] | [-k списокУзлов]]**
**[-w таймаут] конечноеИмя**

**Параметры:**

| | |
|---|---|
| **-t** | **Отправка пакетов на указанный узел до команды прерывания. Для вывода статистики и продолжения нажмите <Ctrl>+<Break>, для прекращения - <Ctrl>+<C>.** |
| **-a** | **Определение адресов по именам узлов.** |
| **-n число** | **Число отправляемых запросов.** |
| **-l размер** | **Размер буфера отправки.** |
| **-f** | **Установка флага, запрещающего фрагментацию пакета.** |
| **-i TTL** | **Задание срока жизни пакета (поле "Time To Live").** |
| **-v TOS** | **Задание типа службы (поле "Type Of Service").** |
| **-r число** | **Запись маршрута для указанного числа переходов.** |
| **-s число** | **Штамп времени для указанного числа переходов.** |
| **-j списокУзлов** | **Свободный выбор маршрута по списку узлов.** |
| **-k списокУзлов** | **Жесткий выбор маршрута по списку узлов.** |
| **-w таймаут** | **Таймаут каждого ответа в миллисекундах.** |

**TTL – Time To Live – время жизни пакета –предельный срок, в течении которого пакет может перемещаться по сети. Задается в секундах. Часто интерпретируется как максимальное количество маршрутизаторов, через которые может пройти пакет. Каждый маршрутизатор вычитает из TTL единицу. Когда TTL=0 – уничтожается**

**[практика: пар. IP без шлюза по умолчанию_ping_route_ping]**

# Команда “tracert” (traceroute в UNIX)

**Отображает список маршрутизаторов, которые в настоящий момент пересылают пакеты к целевому хосту.**

**Использование: tracert [-d] [-h максЧисло] [-j списокУзлов] [-w интервал] имя**

**Параметры:**

- **-d** Без разрешения в имена узлов.
- **-h максЧисло** Максимальное число прыжков при поиске узла.
- **-j списокУзлов** Свободный выбор маршрута по списку узлов.
- **-w интервал** Интервал ожидания каждого ответа в миллисекундах.

***Пример:***

**C:\Documents and Settings\user>tracert www.ya.ru**

**Трассировка маршрута к ya.ru [213.180.204.8]**
**с максимальным числом прыжков 30:**

| | | | | |
|---|---|---|---|---|
| 1 | <1 мс | <1 мс | <1 мс | 192.168.0.1 |
| 2 | 1 ms | 1 ms | 1 ms | 192.168.32.2 |
| 3 | 1 ms | 1 ms | 1 ms | 195.234.109.12 |
| 4 | 1 ms | 2 ms | 1 ms | 172.31.77.10 |
| 5 | 2 ms | 2 ms | 2 ms | 89.19.160.97 |
| 6 | 2 ms | 2 ms | 2 ms | 89.19.160.74 |
| 7 | 2 ms | 3 ms | 2 ms | 213.79.69.217 |
| 8 | 4 ms | 4 ms | 4 ms | ix1-m10.yandex.net [193.232.246.93] |
| 9 | 4 ms | 4 ms | 4 ms | ya.ru [213.180.204.8] |

**Для каждого перехода приводятся интервалы, прошедшие между передачей и приемом трех сообщений**

**Трассировка завершена.**

# Команда “hostname”

**Отображает имя локального компьютера и не предусматривает никаких действий.**

```
C:\Documents and Settings\user>hostname
TRAINER-I945G
```

# Команда “ipconfig”

**Просмотр параметров сетевых интерфейсов, а также настройка продления или прекращения DHCP-аренды.**

**ipconfig [/? | /all | /release [адаптер] | /renew [адаптер] |**
**/flushdns | /displaydns /registerdns |**
**/showclassid адаптер |**
**/setclassid адаптер [устанавливаемый_код_класса_dhcp] ]**

**Где**

**адаптер Полное имя или имя, содержащие подстановочные знаки "*" и "?" (* - любое количество знаков, ? - один любой знак). См. примеры**

**ключи:**

**/? Отобразить это справочное сообщение.**
**/all Отобразить полную информацию о настройке параметров.**
**/release Освободить IP-адрес, полученный от DHCP, для указанного адаптера.**
**/renew Обновить аренду IP-адреса от DHCP, для указанного адаптера.**
**/flushdns Очистить кэш разрешений DNS.**
**/registerdns Обновить все DHCP-аренды и перерегистрировать DNS-имена**
**/displaydns Отобразить содержимое кэша разрешений DNS.**
**/showclassid Отобразить все допустимые для этого адаптера коды (IDs) DHCP-классов.**
**/setclassid Изменить код (ID) DHCP-класса.**

**•По умолчанию отображается только IP-адрес, маска подсети и стандартный шлюз**
**•для каждого подключенного адаптера, для которого выполнена привязка с**
**•TCP/IP.**

**•Для ключей /Release и /Renew, если не указано имя адаптера, то будет**
**•освобожден или обновлен IP-адрес, выданный для всех адаптеров,**
**•для которых существуют привязки с TCP/IP.**

**•Для ключа SetClassID, если не указан код класса (ID),**
**•то существующий код класса будет удален.**

**•Примеры:**
- **> ipconfig - Отображает краткую информацию.**
- **> ipconfig /all - Отображает полную информацию.**
- **> ipconfig /renew - Обновляет сведения для всех адаптеров.**