GeekBrains

Компьютерные сети

Введение в компьютерные сети

Как работают компьютерные сети. IP-адреса и доменные имена. Основные определения. Сетевые модели. Графы и топологии

Введение

Компьютерные сети — вещь сложная, но интересная. Мы будем изучать их на простых примерах. Вы узнаете, почему сети работают так, а не иначе, научитесь настраивать их и использовать полученные знания в разработке и построении и обслуживании инфраструктуры и в программировании — чтобы понимать, как работают ваши сетевые и веб-приложения.

Большинство людей не видят разницы между понятиями «Интернет» и «компьютерная сеть» (а иногда даже и браузером и Интернетом). Интернет — самая большая и известная сеть, которая объединяет множество более мелких сетей во всех уголках мира в единую среду. Она позволяет передавать информацию из одной точки земного шара в другую, хранить данные и организовывать электронные библиотеки и архивы, не беспокоясь о том, где конкретно эти данные хранятся, и обеспечивая распределенные вычисления и удаленный доступ к приложениям, которые могут быть установлены на компьютере, находящемся за много километров от вашего.

Эту сеть сетей иногда называют всемирной или глобальной сетью, но чаще всего мы говорим просто «Интернет». Технологическая основа сети Интернет (и не только) — стек протоколов TCP/IP, который является одной из основных тем большинства наших занятий. Что это такое, мы рассмотрим чуть позже.

На основе Интернета работает Всемирная паутина (World Wide Web, или WWW). Ее тоже не стоит путать с Интернетом, хотя она и доступна через него. Также от него зависят множество других служб, о которых мы с вами будем говорить в этом курсе. Большинство мобильных приложений, которые предоставляют пользователям функции обмена информацией, используют сетевые технологии. Все веб-приложения размещаются на веб-серверах, и доступ к ним пользователи получают через компьютерные сети. Сложные информационные системы, которые установлены на всех крупных предприятиях, зависят от работоспособности сети. Знание работы компьютерных сетей позволит не просто написать программу или сервер, который обеспечит обмен информацией, но и выбрать правильную архитектуру и протоколы, которые будут справляться с потребностями пользователей, вовремя доставлять нужную информацию и гарантировать ее целостность.

Данный курс подробно рассматривает сетевые технологии, используемые на самом разном уровне: от физических основ построения сетей до механизмов взаимодействия ПО с сетью. Так или иначе сетевые проблемы решают самые разные специалисты. Программисты, разрабатывающие веб- и мобильные приложения, системные администраторы, настраивающие сетевое ПО, сетевые инженеры, обслуживающие телекоммуникационное оборудование. Это очень разные профессии, но каждая из них вносит важный вклад в функционирование компьютерных сетей как единого целого. При этом работники каждой из этих трех профессий, как правило, плохо понимают то, что знают о сетевых технологиях их коллеги. Данный курс призван устранить эти пробелы в знаниях: окончив его, вы будете понимать сетевые технологии на высоком уровне. Программа курса будет полезна программистам, системным администраторам, DevOps-инженерам и специалистам по

информационной безопасности. Кроме того, знания, которые вы получите на этом курсе, позволят вам сделать первые шаги в мире сетей, а начинающим сетевым инженерам пригодятся для дальнейшей сдачи экзамена на сертификат CCNA.

Зачем программисту знать, как работают сетевые технологии

Современное общество все больше зависит от работы информационных и инфокоммуникационных систем. В дальнейшем эти тенденции будут только усиливаться. Электронная коммерция и Интернет вещей (IoT, Internet of Things) — яркие примеры использования информационными системами сетевых возможностей. Если раньше можно было писать десктопные приложения, не задумываясь о сетевых технологиях, сейчас все большее распространение получают мобильные и веб-приложения, вычислительные машины становятся частью инфраструктуры, где ряд задач решается не на компьютере или смартфоне, а на удаленном сервере. Таким образом, в программировании так или иначе придется иметь дело с сетью.

Заниматься программированием, даже веб-программированием, без понимания работы сетей можно. Но высококвалифицированный программист не только использует инструменты, но и понимает их работу. Большинство задач позволяют не вникать в стек TCP/IP, но есть и другие, в которых могут всплывать неожиданные и не вполне очевидные проблемы. Почему приложение не работает, при том что клиент и сервер настроены правильно? Необходимо выполнить диагностику, проанализировать входящий и исходящий трафик. Может быть, проблема в шлюзе провайдера или организации. Может быть, там установлен прозрачный прокси-сервер, с которым ваше приложение несовместимо. А может быть, организация пытается контролировать и изменять трафик, из-за чего приложение становится неработоспособным. Такие вещи могут показаться неочевидными, но специалисты, которые успешно справляются с такими задачами, всегда могут претендовать на хорошие должности и оклады, а самое главное — гордиться интересной и не самой легкой работой.

Еще один пример: безопасна ли передача пользовательских данных на сервер? Не все догадываются, что использовать протокол FTP в чистом виде для передачи данных на сервер — дурной тон. С другой стороны, FTP все еще используется для хранения общедоступных архивов. А как оградить свое приложение (блог, форум, чат) от назойливых посетителей. Как забанить злоумышленника? По MAC-адресу? По IP-адресу? По куки (cookies)?

Продиагностировать проблемы в собственной сети, настроить домашний роутер, определить, почему не работает сеть, — все эти задачи рано или поздно встают почти перед каждым из нас.

Знание сетевых технологий, понимание работы стека протоколов TCP/IP, клиент-серверной архитектуры и протоколов прикладного уровня — основа для разработки сетевых приложений. Не важно, что нужно написать — веб-приложение, игру или сложную банковскую систему, — все эти программы используют компьютерные сети для получения и передачи данных.

Без знания компьютерных сетей невозможно использовать облачные решения. Что такое IP-адрес, чем отличаются локальный и глобальный адреса, что означают слова «сетевой мост», «маршрутизатор»? Без знания сетевых технологий невозможно использовать такие облачные решения, как AWS (Amazon Web Services) или MCS (Mail Cloud Solutions)

Как работают компьютерные сети

Прежде чем разбирать детали работы сетей, давайте рассмотрим общий принцип, на примере открытия сайта в браузере.

Откроем браузер и наберем адрес одного из известнейших сайтов — библиотеки Максима Мошкова:

```
http://lib.ru
```

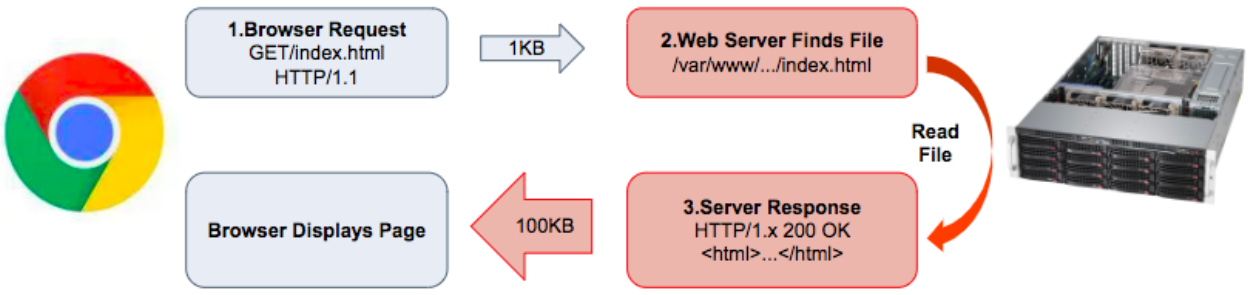
Здесь мы указали протокол HTTP и доменное имя сайта: lib.ru — символическое имя сайта, которое проще запомнить, чем набор цифр, идентифицирующий сервер, где находится машина: 81.176.66.163

Но даже если мы напишем без указания протокола —

```
lib.ru
```

…современный браузер все равно догадается, что мы будем подключаться по протоколу HTTP.

Что же произойдет, когда мы нажмем Enter? Чтобы отобразить страницу, браузер должен сформировать HTTP-запрос, после чего запрос будет отправлен на веб-сервер, сервер прочитает файл главной страницы (index.html) с диска и пришлет нам ответ.



Казалось бы, что может быть проще? Но это только верхушка айсберга. Давайте посмотрим, что происходит «под капотом».

Как только мы набрали адрес страницы (lib.ru), операционной системе необходимо узнать, что такое lib.ru, какой у сервера, к которому нужно обратиться, цифровой адрес (далее мы будем называть его IP-адресом). Пока ни браузер, ни операционная система этого еще не знают.

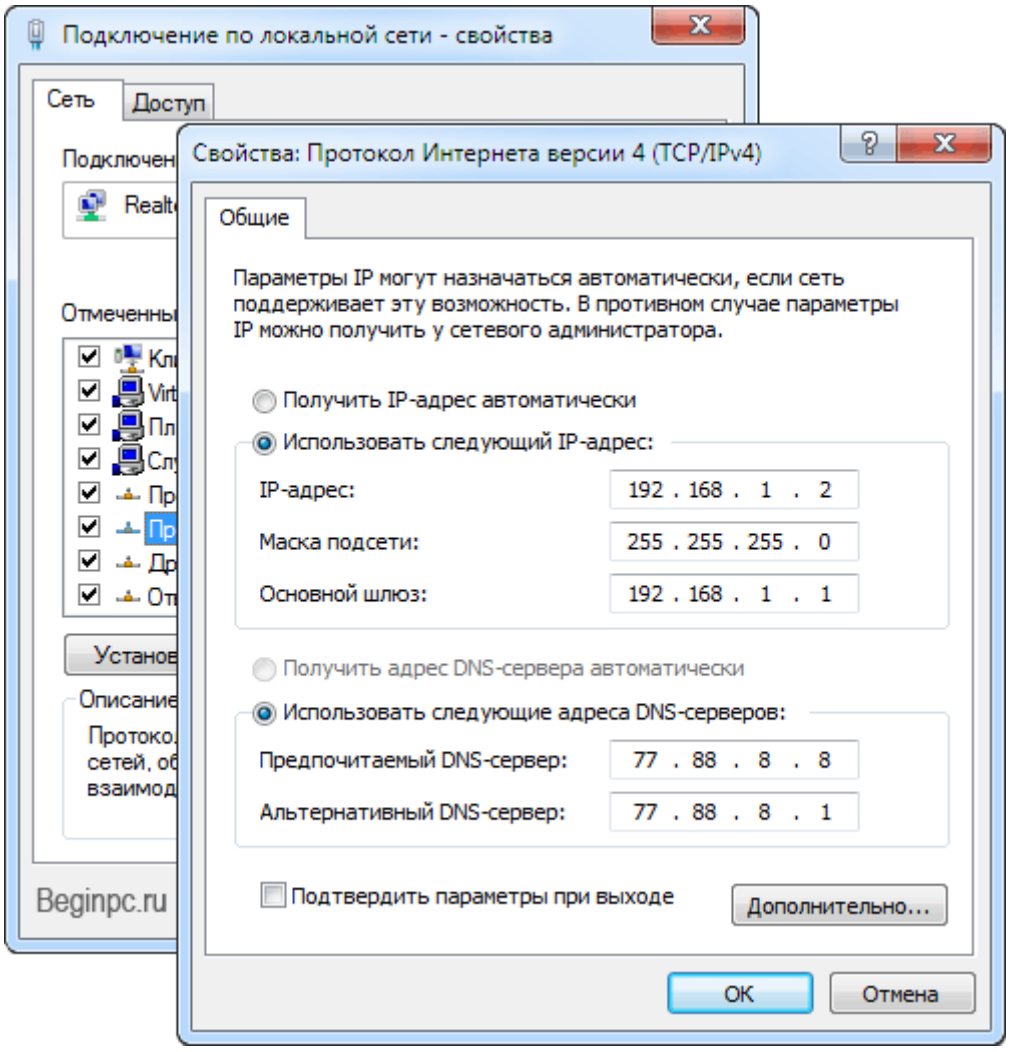
Чтобы его узнать, локальный компьютер обращается к системе DNS (Domain Name System — система доменных имен), которая сопоставляет понятные человеку символьные имена сайтов и серверов (доменные имена, такие как lib.ru, mail.ru, geekbrains.ru, habr.com) и понятные компьютеру цифровые адреса.

DNS — распределенная система серверов. Адрес одного из DNS-серверов можно увидеть в настройках TCP/IP вашей операционной системы. Для работы в сети Интернет операционная система должна знать адрес хотя бы одного DNS-сервера, но может быть указано и несколько адресов (DNS-серверы, как правило работают парами: если один не ответил, то запрос будет направлен второму).

В настройках может быть указан либо DNS-сервер провайдера, либо публичные открытые DNS-серверы, предоставляемые такими компаниями, как Google, Cloudflare, Яндекс. Эти адреса могут быть предоставлены провайдером автоматически, либо выставлены вручную. К примеру, на картинке ниже указаны два DNS-сервера:

- первичный (основной) DNS-сервер: 77.88.8.8;
- вторичный (резервный) DNS-сервер: 77.88.8.1.

В данном случае это DNS-серверы от Яндекс.DNS, адреса которых были прописаны вручную. Другие варианты — от Google (8.8.8.8 и 8.8.4.4), от Cloud Flare (1.1.1.1 и 1.0.0.1).

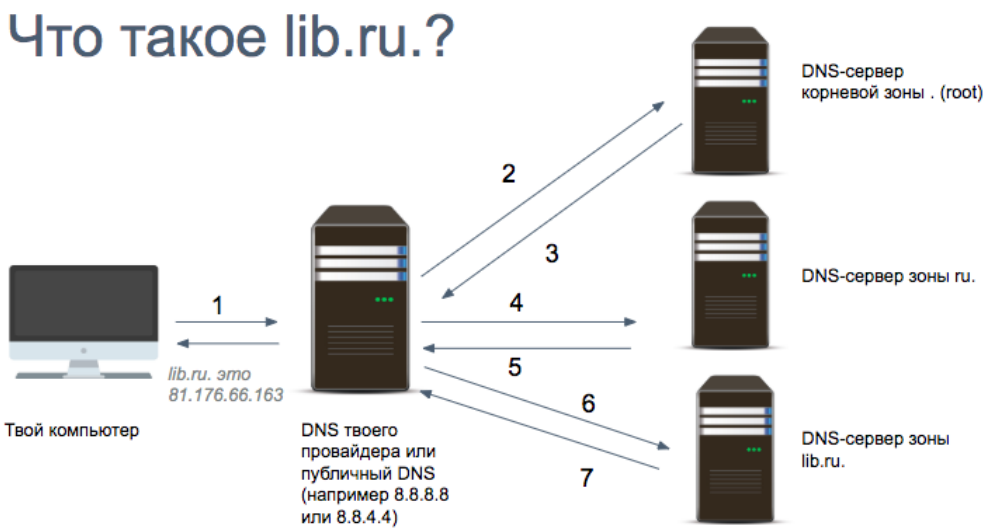


Итак, компьютеру известен адрес DNS-сервера. Чтобы вы могли обратиться к серверу lib.ru, ваш компьютер сначала обращается к DNS-серверу с запросом, что такое lib.ru.

Если DNS-сервер знает его адрес (например, если аналогичный запрос поступал раньше, и ответ закешировался, потому DNS-серверы, к которым обращаются непосредственно компьютеры конечных пользователей, также называются кеширующими), он сразу отправит в ответ нужный адрес, но как быть, если к этому сайту за недавнее время обратились впервые? Тогда сервер провайдера (или публичный открытый DNS-сервер) спрашивает дальше.

Сначала спрашивается сервер корневой зоны (root; она обозначается одним символом — «.» — точкой). В ответ корневой сервер присылает адреса серверов, которые знают, что такое ru. (в записях доменных имен в конце доменного имени присутствует указание на корневой домен; так как домены ru, org и com принадлежат корневой зоне, то они обозначаются как ru. и org. и com.)

Узнав адрес DNS-сервера, отвечающего за ru., мы теперь можем у него узнать сервер, который отвечает за lib.ru, и уже у последнего — что такое lib.ru. Сервер DNS нам сообщит, что для того, чтобы сделать запрос к серверу lib.ru. нужно обратиться к машине с IP-адресом 81.176.66.163.



Поиск адреса для сайта lib.ru

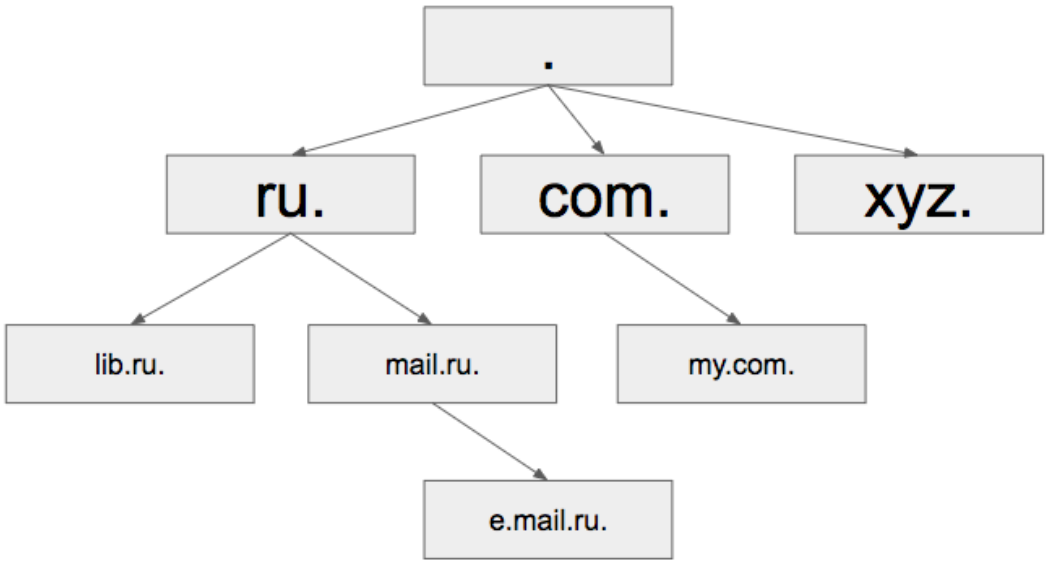


Иллюстрация иерархии доменных зон

Теперь, когда мы знаем цифровой адрес (IP-адрес) сервера, на котором находится страница сайта, мы можем запросить страницу, и больше нам ничего знать не нужно?

Нет. Страница большая, а в сети существует ограничение на максимальный размер пакета. Как быть, если размер страницы окажется больше?

Страницу придется поделить на фрагменты, их нужно будет передать и заново собрать в той же последовательности, в которой они были отправлены. Это уже отдельный процесс, который нужно инициировать и управлять им.

Кроме того, напомним, что Интернет — это сеть сетей. Каналов передачи информации очень много, а возможных маршрутов в сети еще больше. Встает вопрос, куда дальше отправить информацию чтобы она дошла до получателя. Эти задачи тоже решаются с помощью специальных протоколов.

Как видите, даже такое небольшое событие, как запрос страницы, складывается из многих действий: мы отправляем запрос, веб-сервер находит страницу на сервере, разбивает ее на фрагменты и отправляет нам по частям. Если какой-то из пакетов будет потерян, веб-сервер заново отправит утерянные фрагменты. И мы наконец сможем посмотреть нашу страницу в браузере.

Теперь вы понимаете, насколько это сложный и интересный процесс. Вместе мы разберемся во всех подробностях, как это работает.

IP-адреса и доменные имена

Имена нужны всему. Люди привыкли обращаться друг к другу по именам. Даже если мы используем позывные при общении по рации или телефонные номера для звонков по телефону, в конечном итоге всё равно мы обращаемся друг к другу по имени, имя и фамилию мы записываем в адресной книге.

Нужны имена и компьютерам. Но если наши человеческие имена состоят из букв, из звуков, которые мы произносим, компьютеры работают с числами и только с числами. Неудивительно, что адреса, используемые компьютерами, — численные.

Наибольшее распространение получили IP-адреса версии 4. IP расшифровывается как Internet Protocol — межсетевой протокол. Сейчас существуют протоколы версии 4 (IPv4, о котором мы будем говорить по большей части времени) и 6 (IPv6, которого мы тоже коснемся).

Примеры IPv4-адресов:

- 8.8.8.8;
- 192.168.1.10;
- 81.176.66.163;
- 77.88.8.1;
- 10.0.0.2.

Возможно, какие-то из этих адресов вы даже встречали при работе с компьютером и сетями.

Для простоты можно считать, что IPv4-адрес — это четыре десятичных числа, разделенных точками. Но на самом деле каждый компонент (октет) такого адреса может принимать значение в диапазоне от 0 до 255. То есть адреса 1.2.3.400 или 1.1.1.256 уже не являются валидными.

Почему так? Размер IPv4-адреса — 4 байта, или 32 бита. Каждый байт может принимать значения от 0 до 255:

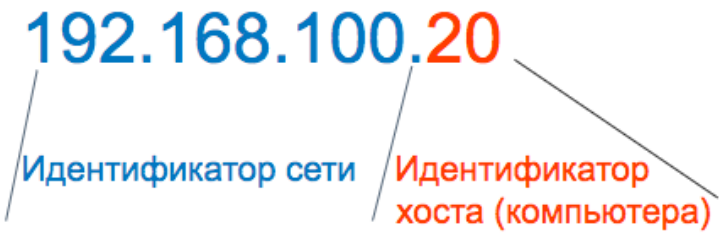
- 8.8.4.4 = x08080404;
- 5.255.255.5=x05FFFF05;
- 10.1.11.17 = x0A010B11.

В примере выше разные биты IP-адреса обозначены разными цветами.

Это похоже на телефонные номера, которые имеют единую структуру:

Телефонный номер содержит код страны, код города или мобильного оператора, код выхода на АТС и номер абонента, относящегося к данной АТС.

Структура IP-адреса в чём-то похожа: он содержит часть, идентифицирующую сеть, в которой находится абонент, и адрес хоста (компьютера) относительно сети, в которой он находится.



Чтобы определить, какая часть IP-адреса идентифицирует сеть, а какая — хост, используется битовая маска:

Накладывая маску на адрес, мы можем вычленить адрес сети и адрес хоста в этой сети. Внешне маска похожа на IPv4-адрес и также состоит из 4 октетов. Самые простые маски включают 255 и 0 (более сложные случаи мы разберем на следующих уроках). Левая часть такой маски содержит октет/октеты 255, и указывает, что на соответствующих позициях октеты IPv4 идентифицируют сеть. Правая часть такой маски содержит октет/октеты 0, и указывает, что на соответствующих позициях октеты IPv4-адреса идентифицируют хост.

Существуют особого вида IPv4-адреса, идентифицирующие сети. Биты, служащие для идентификации хоста, в таких IPv4-адресах установлены в ноль. Для определения адреса сети используется маска, и если маска содержит только октеты 255 и 0, то адрес сети будет в хостовых октетах также содержать 0.

У сетей разного размера разные маски. Рассмотрим примеры IPv4-адресов и масок разной длины.

Адрес: 192.168.1.100
Маска: 255.255.255.0
Сеть: 192.168.1.0

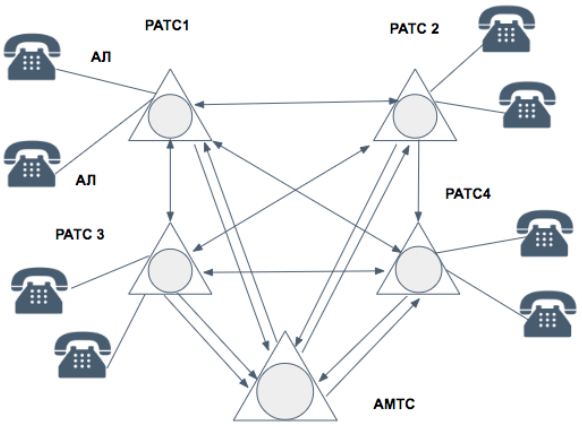
Адрес: 172.16.10.40
Маска: 255.255.0.0
Сеть: 172.16.0.0

Адрес: 10.1.2.3
Маска: 255.0.0.0
Сеть: 10.0.0.0

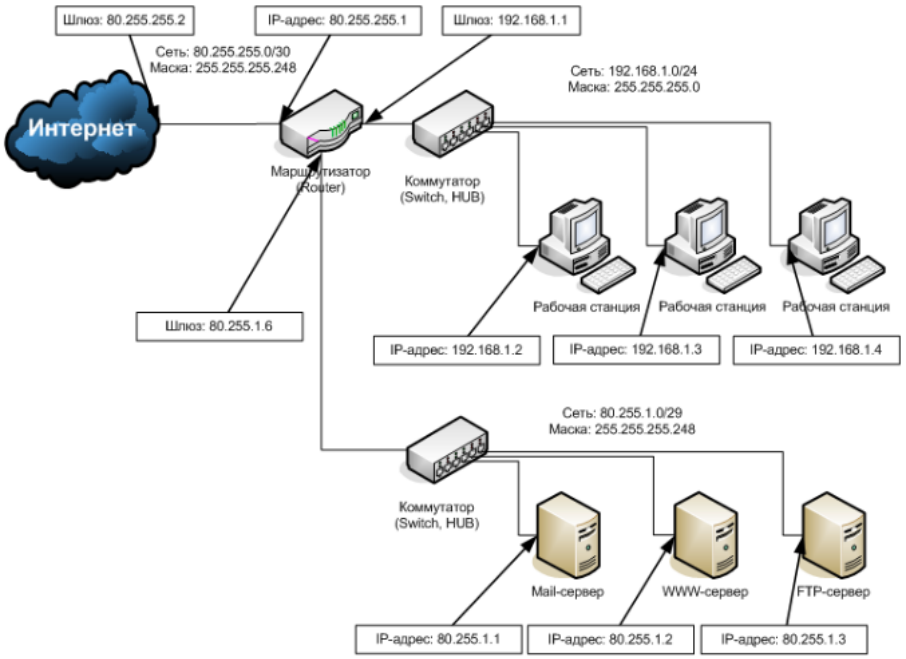
Маска всегда отсекает от сетевой части адрес хоста, оставляя адрес сети.

Зачем нужна такая иерархия — адрес сети, адрес хоста? Интернет — это сеть сетей. И чтобы два компьютера (хоста) из разных сетей могли связаться, сначала нужно «выйти» на другую сеть, и передать пакет далее.

Наверняка вы слышали об АТС и плане нумерации. При телефонном звонке мы выходим на нашу АТС, далее она связывается с АТС, номер которой включён в телефонный номер, чтобы та переключила на необходимого абонента.



В телекоммуникационных же сетях найти путь до адресата и довести до него отправленные пакеты помогают коммутаторы и маршрутизаторы.



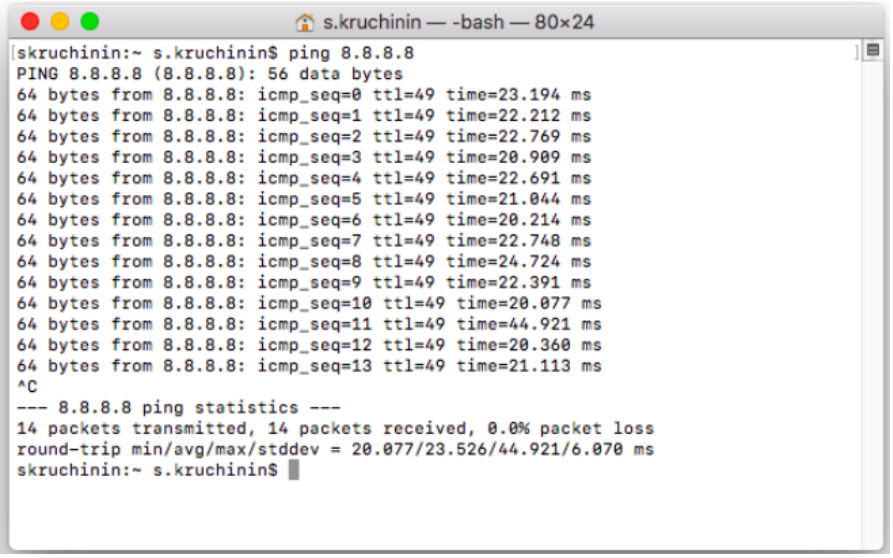
Для проверки связи используют терминал операционной системы и команду **ping**. Для этого сначала надо открыть терминал:

- в Windows нажать **Windows+R** и набрать **cmd;**
- в Linux нажать **Ctrl+Alt+**T в оконном режиме, либо **Ctrl+Alt+F1** или **Alt+F1** в текстовом режиме;
- в Mac OS набрать **Cmd+Проблел** и начать набирать **terminal.app**.

Команда **ping** будет работать во всех операционных системах.

Проверим:

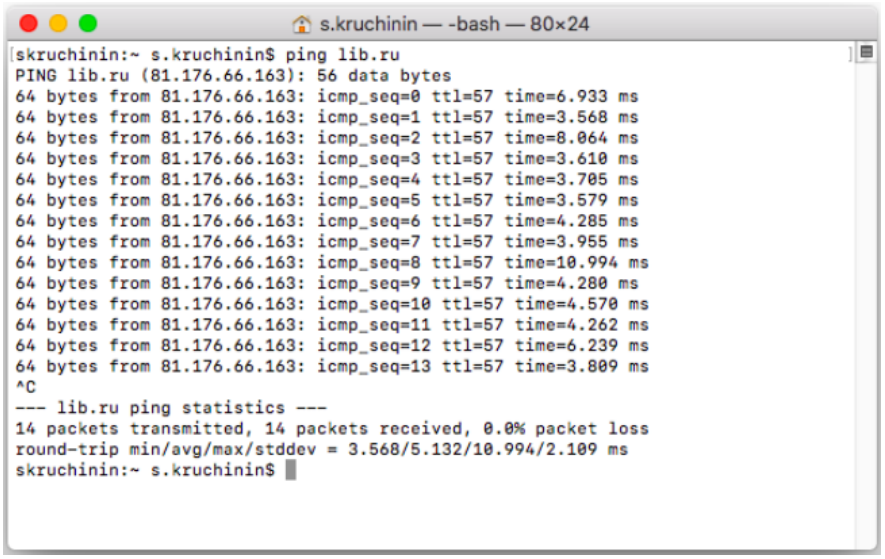
```
ping 8.8.8.8
```



Для остановки в Linux и Mac OS можно нажать **Ctrl+C**.

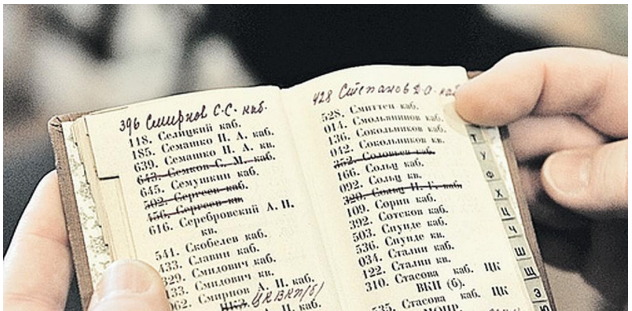
Работает **ping** и при использовании доменных имен:

```
ping lib.ru
```



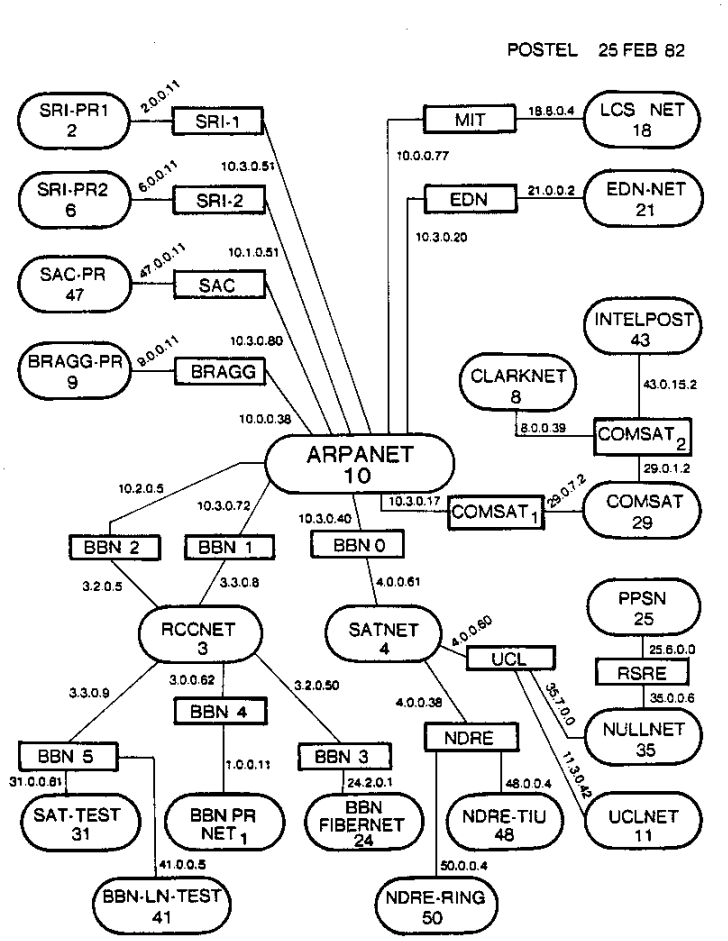
Если хост отвечает, мы будем видеть время отклика.

Числовые адреса сложно запомнить. Чтобы не держать телефонные номера в памяти, мы



записываем их в телефонные книги вместе с именами их владельцев. Сейчас адресная книга есть в любом мобильном телефоне.

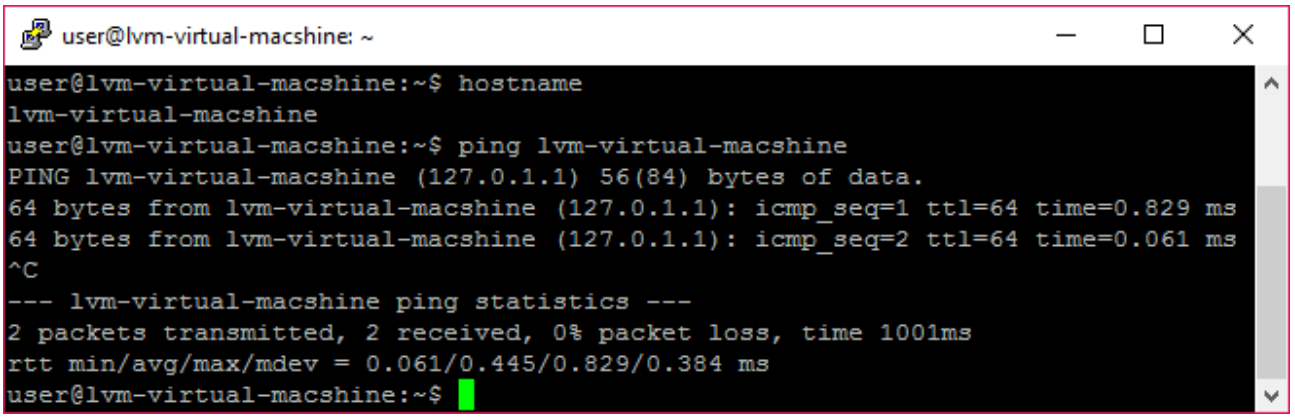
Когда появились первые предшественники сетей, из которых и развился современный Интернет, у компьютеров в этих сетях были только числовые адреса, но когда их стало слишком много, понадобилось дать компьютерам имена.



Тогда придумали псевдонимы — имена хостов.

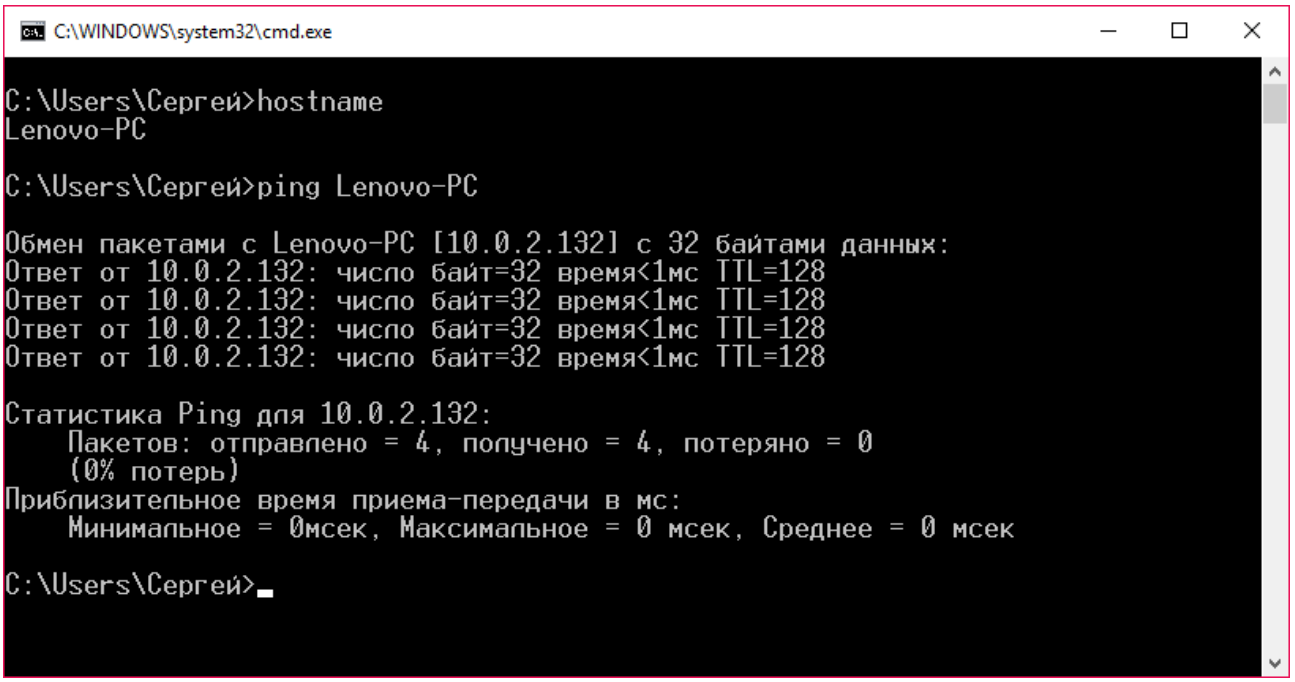
Протокол IPv4 появился в 1981 году и стал стандартом для ARPANET, первой глобальной компьютерной сети и предшественницы Интернета. Маленькая сеть на иллюстрации не сравнится в масштабах с существующей, но даже то количество компьютеров, которое есть на схеме, достаточно велико, чтобы оперировать вручную IP-адресами (или иными адресами сетевого уровня) было неудобно. Поэтому компьютерам стали давать имена. Символьное имя хоста (hostname) можно сопоставить с IP-адресом. В командной строке Windows и Linux hostname покажет имя вашего компьютера. Его можно использовать вместо IP-адреса. В GNU/Linux:

```
user@host: ~ $ hostname
user@host: ~ $ ping lvm-virtual-macshine
```



В Windows:

```
C:\>hostname
Lenovo-PC
C:\>ping Lenovo-PC
```

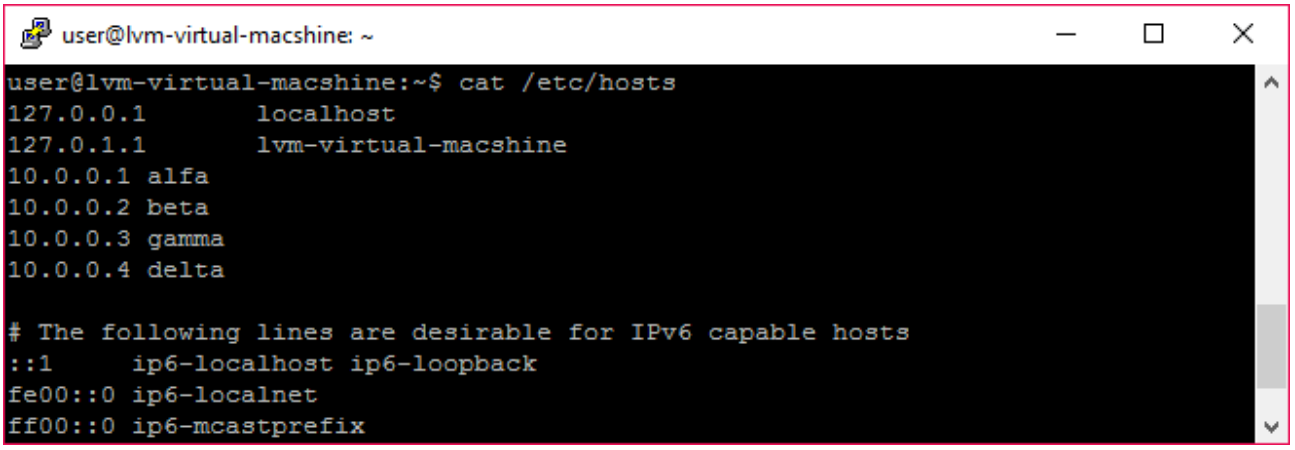


Компьютеры соединяются, используя IP-адреса. Когда вместо IP-адреса мы пишем имя хоста, операционная система обращается к DNS для преобразования имени хоста в IP-адрес, что мы и видим на рисунках: ответ приходит от указанного IP-адреса.

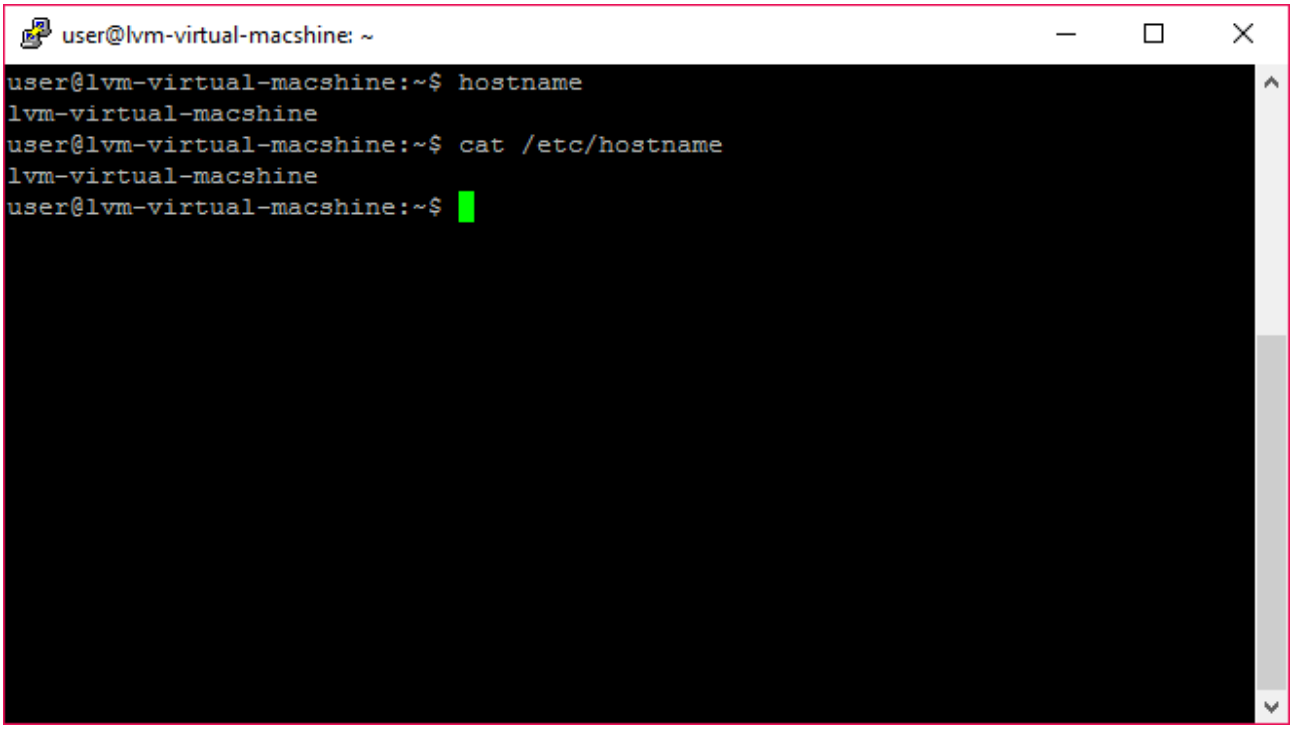
Выше можно увидеть, как работает команда **ping**: вы пишете адрес, с которым хотите проверить связь, и компьютер отправляет по очереди специальные пронумерованные сообщения (ICMP-пакеты; ICMP — Internet Control Message Protocol, протокол межсетевых управляющих сообщений), а проверяемый компьютер отправляет их назад. Если сообщение дошло до компьютера (не было потеряно) и вернулось, то отображаются данные об ответе и затраченном времени.

Современные компьютеры узнают адрес у системы DNS. Но как быть, если доменного имени нет в системе DNS (например, вы разрабатываете веб-сервис локально, а нужно использовать доменные имена)? До появления DNS уже распространились UNIX-подобные системы (по большей части UNIX BSD), а при подходе unix-way конфигурационные файлы хранятся в файлах (в директории /etc). Поэтому на всех компьютерах сети появился файл /etc/hosts, который содержал локальную таблицу

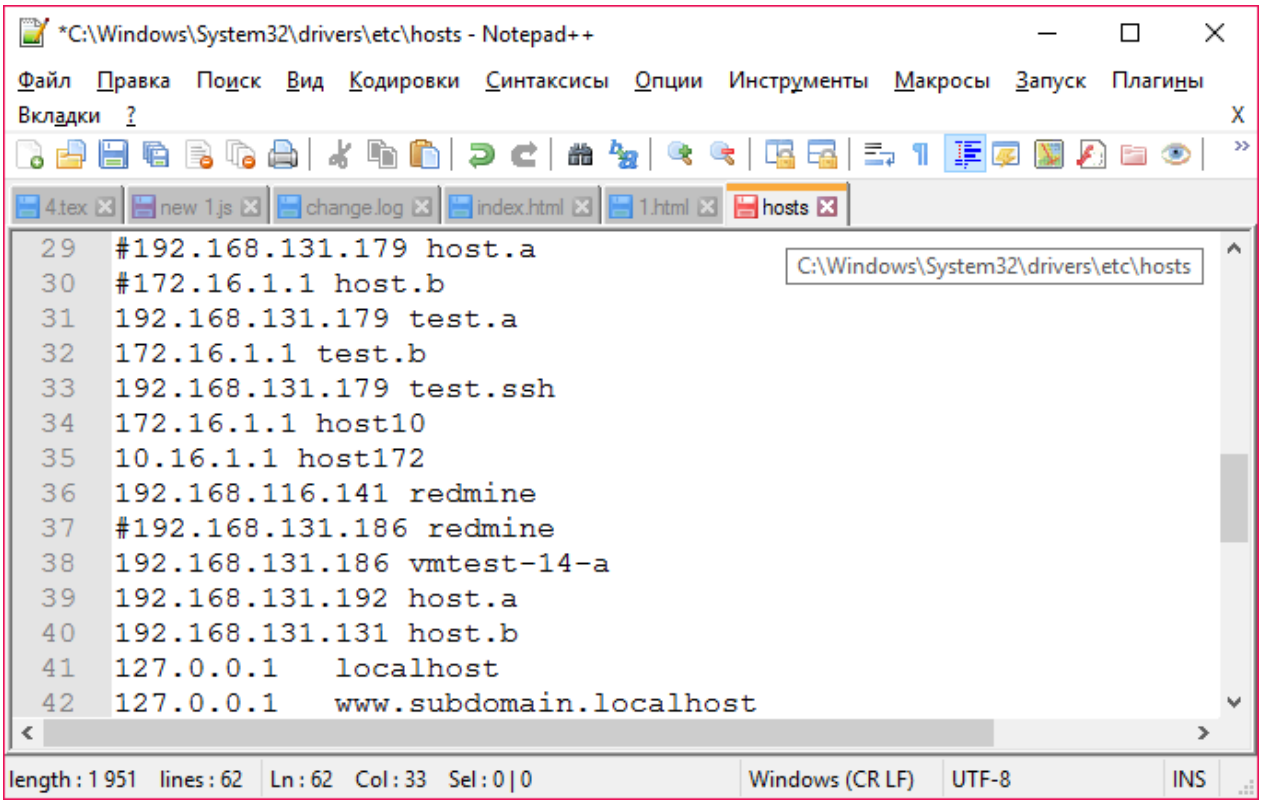
соответствия «IP-адрес — имя хоста». Когда в сети появлялся новый хост (новый компьютер), необходимо было изменять файл /etc/hosts на всех машинах (команда **cat** в Linux позволяет вывести содержимое файла в терминал).



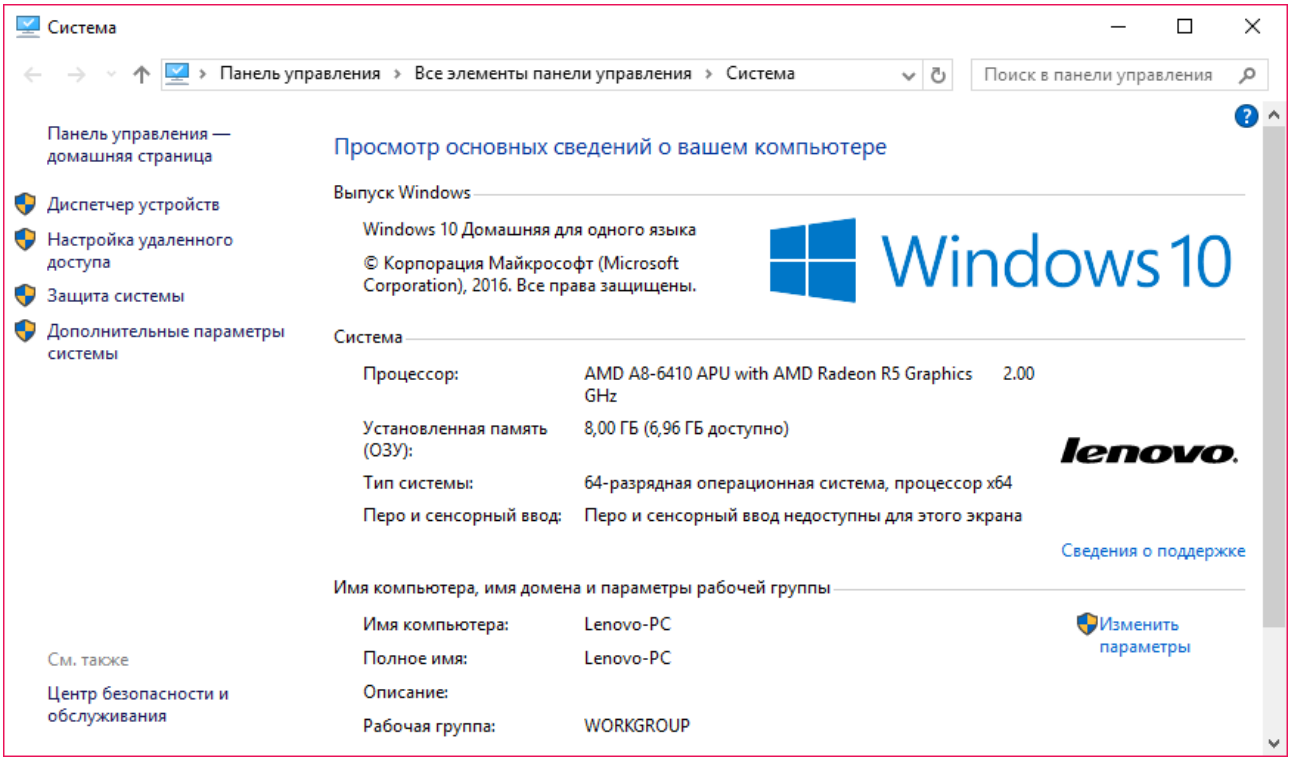
Имя текущей машины также хранилось в файле /etc/hostname:



Список соответствий «IP-адрес — имя хоста» в Windows хранится аналогично. Обычно это файл C:\Windows\System32\drivers\etc\hosts.



А имя компьютера можно посмотреть и изменить в свойствах **Мой компьютер/Этот компьютер** или в панели управления.



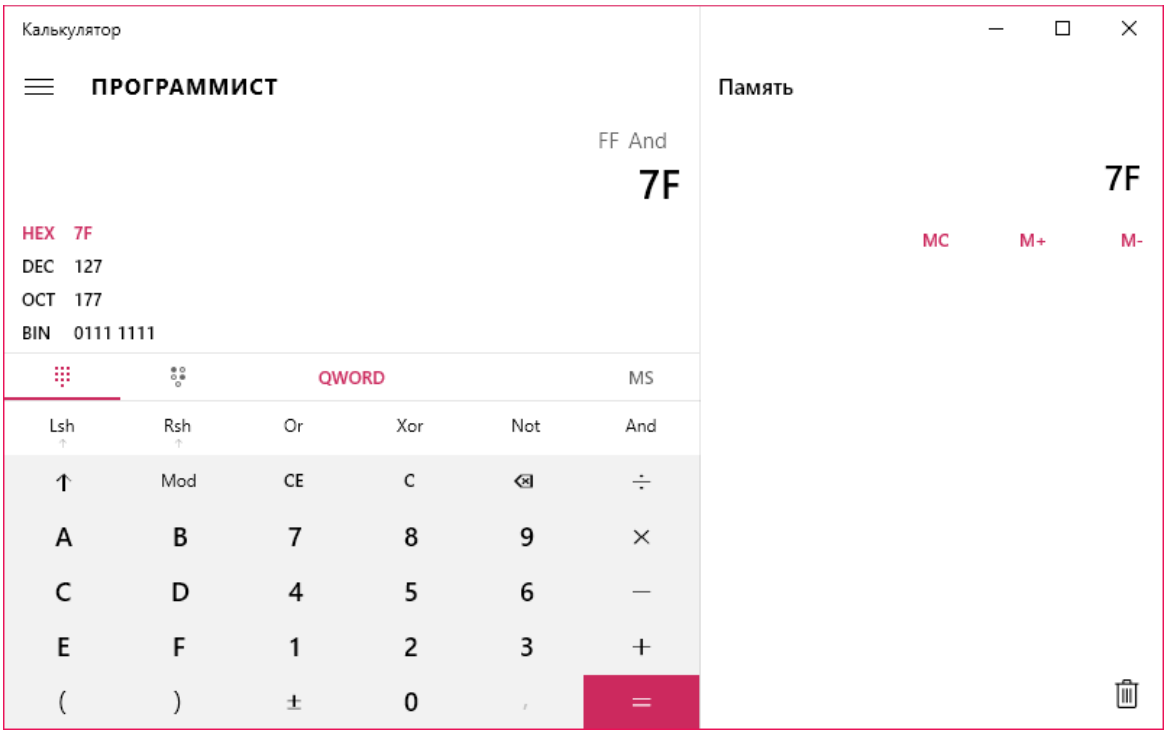
Такой способ обмена информацией в именах хостов (с помощью ручной правки файла /etc/hosts или копирования его вручную с машины на машину) оказался неудобным. А когда сеть разрослась, это стало физически невозможно. Поэтому и понадобилась система доменных имен.

Доменные имена похожи на имена хостов, но построены по иерархическому принципу. Были созданы несколько зон верхнего уровня: .com — для коммерческих сайтов, .org — для некоммерческих, .edu — для образовательных, .mil — для военных. Организации получали домены в этих зонах, например, .mit.edu — зона Массачусетского технологического института, .caltech.edu — доменная зона Калифорнийского технологического института. При этом администратор каждой зоны уже сам выделяет поддомены для соответствующих сервисов и служб. Так веб-сайт Массачусетского технологического института находится на домене web.mit.edu, а сайт Калифорнийского технологического института — на домене www.caltech.edu.

Теперь мы имеем представление, как происходит идентификация компьютеров в сети.

Что нужно знать, чтобы двигаться дальше

Как вы помните, компьютеры работают не с символьными именами, а только с числами — более того, только в двоичной системе. Если вы еще не знакомы с такими вещами, как двоичная и шестнадцатеричная система счисления, перевод из одной системы в другую, логические операции И (AND), ИЛИ (OR), НЕ (NOT), ИСКЛЮЧАЮЩЕЕ ИЛИ (XOR), крайне рекомендуется изучить, это необходимо знать каждому программисту.



Как вы помните, IPv4-адреса состоят из четырёх октетов — десятичных чисел от 0 до 255. Посмотрим, как выглядит произвольный адрес — например, 64.233.164.139 — в шестнадцатеричной и двоичной системе счисления.

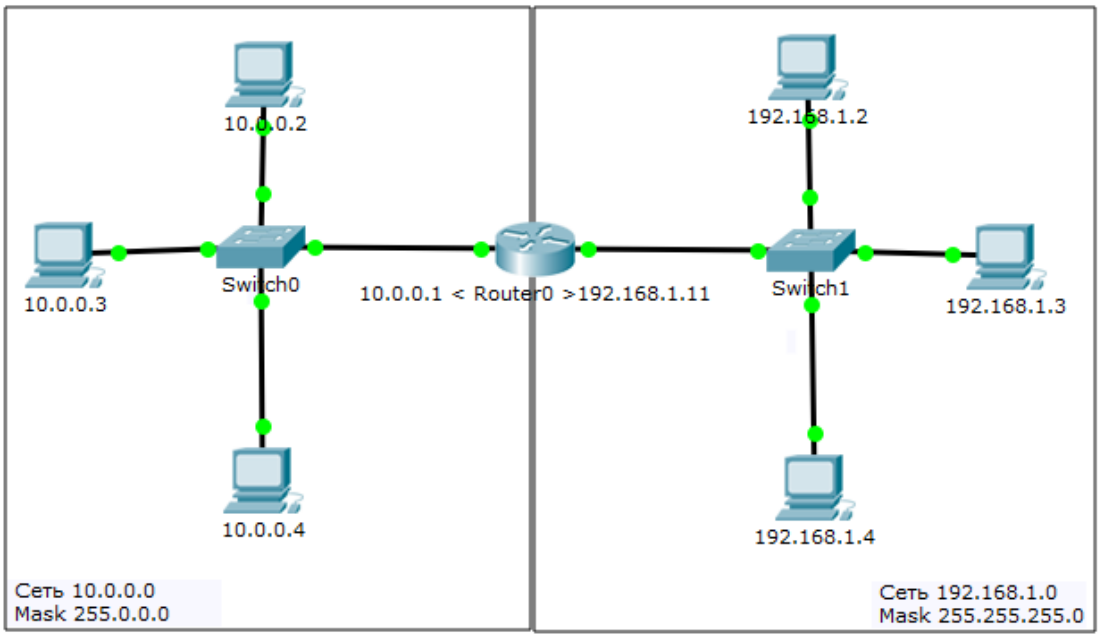
В десятичной системе счисления	64.	233.	164.	139
В шестнадцатеричной системе счисления	40	E9	A4	8B
В двоичной системе счисления	0100 0000	1110 1001	1010 0100	1000 1011

Вы можете посмотреть, как выглядит в других системах счисления любой другой IP-адрес, при

помощи стандартного калькулятора Windows или аналогичного: для этого нужно перейти в режим «Программист».

Таким образом, адрес 64.233.164.139 в шестнадцатеричной системе счисления выглядит как 40.E9.A4.8b и в двоичной системе счисления как 0100 0000.1110 1001.1010 0100.1000 1011. (Точки и пробелы вставлены только для удобства записи.)

Также напомним, что IP-адрес содержит часть, идентифицирующую сеть, и часть, идентифицирующую хост. У компьютеров, находящихся в одной сети, IP-адреса будут начинаться одинаково, в разных — по-разному.



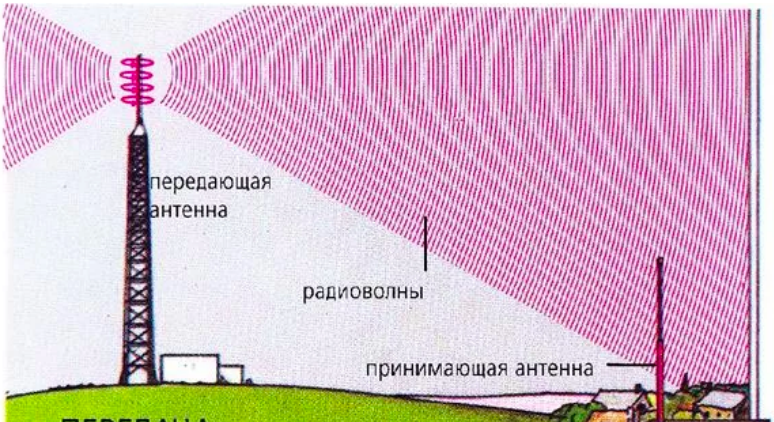
На картинке мы видим две сети. Компьютеры объединяются в сеть при помощи коммутатора (свитча). Одна сеть содержит IP-адреса, начинающиеся с 10.0.0., вторая — с 192.168.1. Маршрутизатор (роутер) объединяет обе сети и обладает двумя IP-адресами (бывают случаи, когда это не так, но мы рассматриваем самый простой пример). Обратите внимание на маску. Если мы возьмем маску сети и адрес, мы можем проверить, принадлежит он этой сети или нет. Если в маске октет 255, значит октет оставляем, если 0 — заменяем на 0. В результате мы получаем адрес сети. (На самом деле система сложнее, это простой случай.)

Информацией между компьютерами можно обмениваться, если и они, и их IP-адреса принадлежат к одной сети.

Основные определения

Чтобы разобраться, как работают сети и из чего они состоят, мы начнем с основных определений. Заучивать эти определения не надо, важнее понимать их.

Виды связи



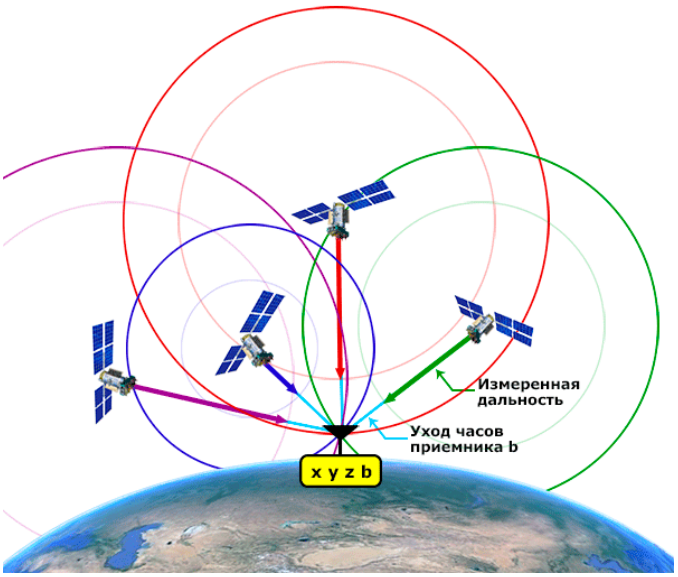
Simplex — односторонняя связь, при которой информация передается только в одном направлении. Сигнал передается от

радиопередающей станции множеству абонентов.

Примеры:

- телевидение или радиовещание;
- передача сигнала от спутников GPS.

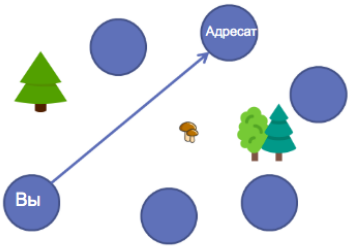
И если с передающей антенной телевидения или радио проблем не возникает, существует распространенное заблуждение насчет навигационной системы GPS (Global Positioning System): якобы GPS-приемник отправляет сигнал на спутник. Ее путают с сетями GSM, в которых мобильный телефон действительно не только принимает сигнал от трех базовых станций, но и отправляет ответ. В случае же с GPS 32 спутника излучают сигнал. Каждый спутник сообщает точное время на спутнике (сообщенное атомными часами, идущими чуть медленнее, чем наши земные часы, чтобы компенсировать эффекты общей и специальной теории относительности, обусловленные собственным движением спутника относительно Земли и отличающейся гравитацией по сравнению с GPS-приемником) и свое положение. Принимая сигнал одновременно от 4 спутников, GPS-приемник решает систему из 4 уравнений, что позволяет определить точные координаты GPS-приемника.



Half-duplex (или полудуплекс) — двусторонняя связь, при которой устройство в каждый конкретный момент может либо передавать, либо принимать информацию. Полудуплексная передача схожа с симплексной, но она позволяет транслировать информацию в оба направления. В любой момент момент передача происходит только в одном направлении. Оба абонента взаимодействуют по очереди. Пример — разговор по рации. Когда несколько абонентов переговариваются на одном канале, вы слышите разговор других абонентов, но чтобы перейти из режима «прием» в режим «передача», необходимо нажать кнопку или использовать тангенту переключения приема-передачи. Ранние Ethernet работали в полудуплексе.

Full-duplex, или просто **duplex** — двусторонняя связь, в которой оба устройства могут вести передачу одновременно. Примером такой связи может быть телефонный разговор, где вы можете говорить и слушать одновременно, без переключения режима. Современные стандарты Ethernet, как правило, работают в full-duplex.

Методы передачи и адресации



Unicast (Юникаст), или однонаправленная (односторонняя) передача данных, подразумевает передачу пакетов единственному адресату. Это самый часто используемый вариант передачи данных.

Пример — общение по рации:

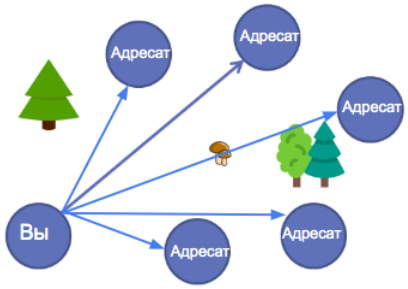
```
Первый, я второй, как слышно? Прием.
```

Несмотря на то, что по рации нас слышат все, обращаемся мы к одному конкретному абоненту с позывным «Первый».

Еще пример использования (в терминале):

```
ping 192.168.1.1
```

(работает с большинством Wi-Fi-маршрутизаторов). У вас только один узел с адресом 192.168.1.1, при этом «слышать» этот пакет технически могут все хосты в вашей локальной сети, но адресован он конкретному получателю.



Broadcast (бродкаст) — широковещательная передача данных всем адресатам в данной сети.

Примером использования при работе по радиосвязи могут быть циркулярные сообщения, которые не только слышны всем, но и должны быть приняты и обработаны каждым узлом, слышащим такое сообщение.

```
Всем всем всем. Я первый. Переходим на резервную частоту. Конец связи.
```

Еще пример использования (в терминале):

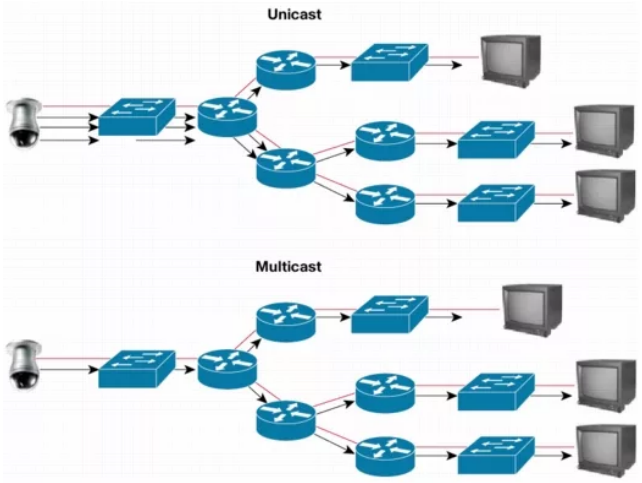
```
ping 192.168.1.255
```

Или (в GNU/Linux):

```
ping -b 192.168.1.255
```

Мы пингуем широковещательный (бродкастовый) адрес и тем самым адресуем наше сообщение всем участником локальной сети, а не кому-то конкретному. На него отвечают все хосты в данной сети. Бродкастовый адрес содержит все битовые единицы в хостовой части адреса.

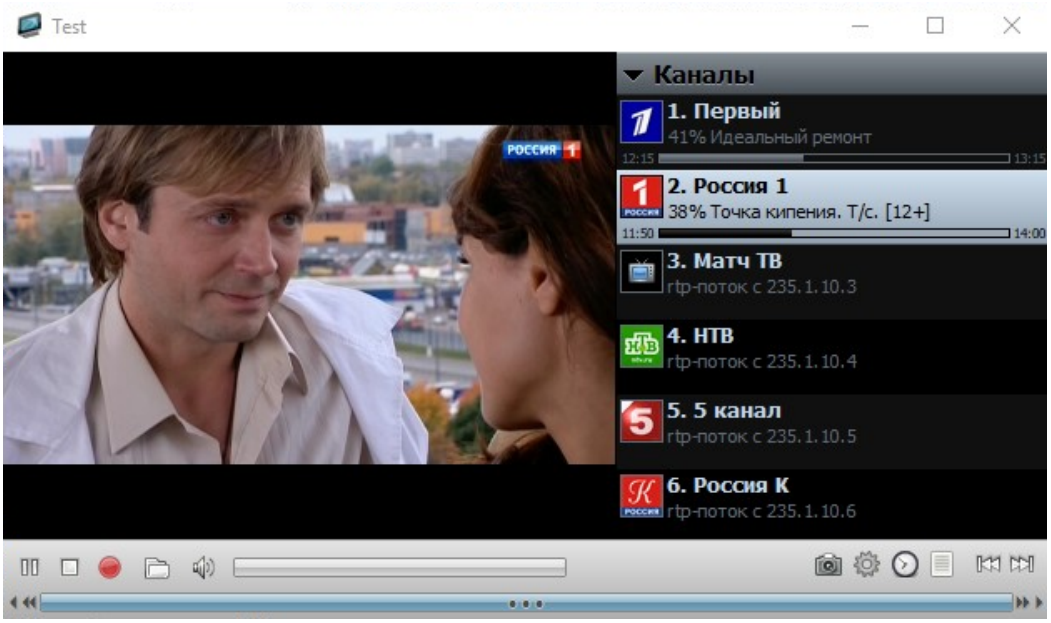
Для вас самое главное — выучить отличие unicast от broadcast. Но также существуют способы адресации multicast и anycast. Мы не будем их использовать на занятиях, но можете также почитать о них здесь либо пропустить и перейти к чтению раздела **«Виды коммутаций»**

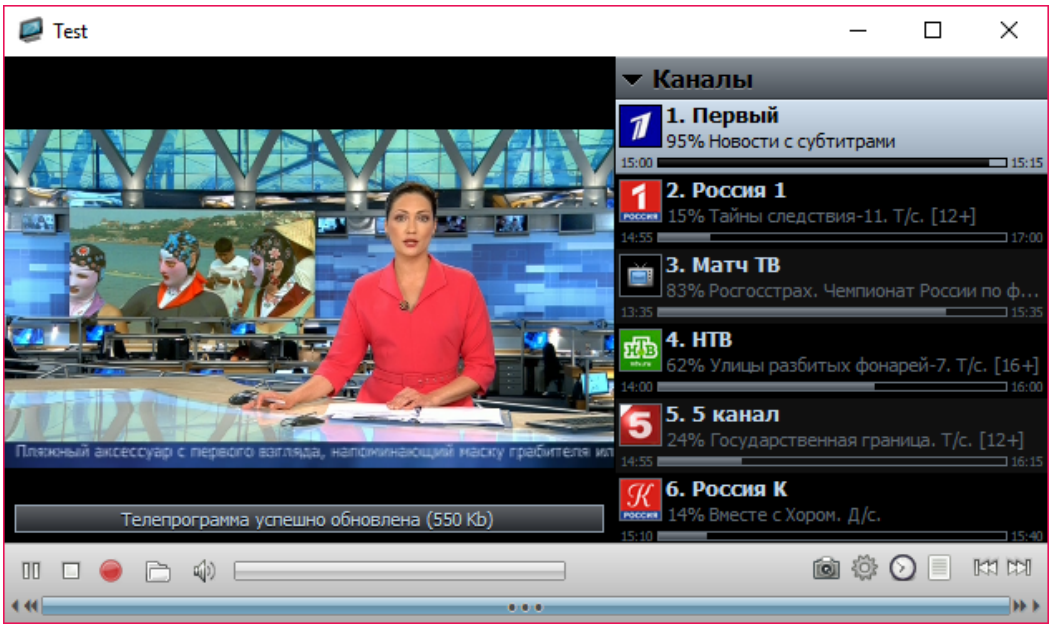


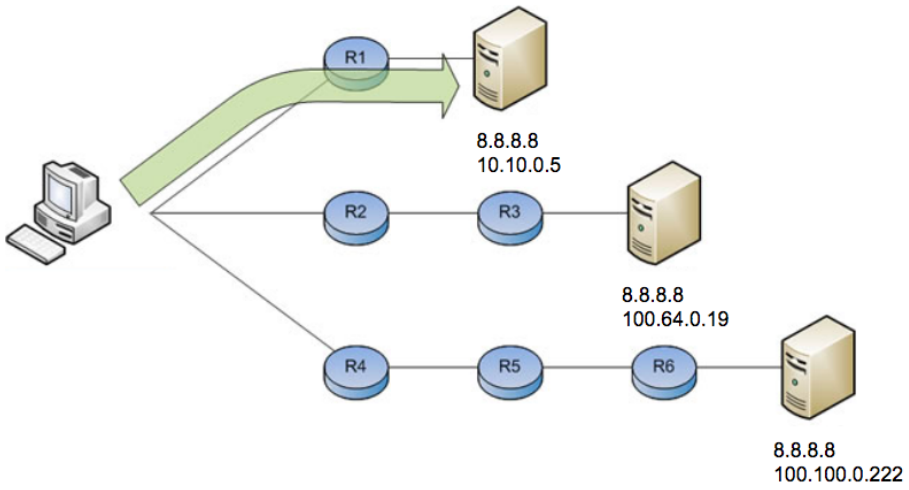
Multicast — мультивещание, многоадресное вещание. Используется в протоколах маршрутизации и в потоковом вещании. Используются специальные multicast IP-адреса, которые назначаются не отдельным хостам, а группам. В отличие от unicast мультикастное вещание позволяет вместо множества потоков информации для каждого из абонентов, дублирующих друг друга использовать только один поток, который будет разветвляться на конечных маршрутизаторах и доводиться уже до каждого абонента отдельно (см. рисунок, вверху — unicast, внизу то же самое, но при использовании multicast. Обратите внимание, что при мультикастном вещании снижается нагрузка на канал).

Сервер транслирует информацию группе получателей. Если клиент хочет получать контент, он с помощью протокола IGMP (Internet Group Management Protocol) подписывается на группу и начинает получать потоковый трафик. Несмотря на то, что сообщение отправляется на multicast-адрес, у компьютера остаются те же IP-адреса, что были и до подписки. Отличие от бродкаста в том, что мультикаст — это не все адреса в сети, а лишь некоторая группа получателей. Это те адреса, которые подписались на мультикаст-рассылку (записались в мультикаст-группу), и они могут быть в разных сетях.

Пример:







Anycast — еще один механизм сетевой адресации. В этом случае при запросе на anycast-адрес сообщение получает один получатель из anycast-группы (несколько хостов, использующих один и тот же IP-адрес). В отличие от multicast-адресов anycast-адреса не образуют особого диапазона. Anycast-адрес является обычным адресом хоста, и теоретически любой IP-адрес может оказаться эникастным. Кроме anycast-адреса, каждый хост, как правило, имеет еще и юникастный адрес для обращения и работы с каждым конкретным физическим сервером.

Примеры использования Anycast:

- корневые DNS-серверы;
- публичные кеширующие DNS-серверы (например, Google DNS);
- система QRATOR использует Anycast для защиты от DDOS-атак (используется Хабром и многими крупными сайтами). Сначала запрос попадает на anycast-адрес, используемый для анализа. Если запрос не является попыткой атаки, то он перенаправляется на адрес защищаемого сайта. https://qrator.net/ru/solutions/ddos/how-qrator-works;
- HTTP CDN (Content Delivery Network);
- шлюзы тоннелей 6to4 для инкапсуляции IPv6-пакетов в IPv4. Они отзываются на 192.88.99.1/32, применяемый в качестве ретранслятора при инкапсуляции IPv6 в IPv4 (6to4). Такие шлюзы поддерживаются многими компаниями, и при использовании 6to4 пакет IPv6,

инкапсулированный в Ipv4, проследует до ближайшего хоста с этим IP, который распакует пакет и отправит его дальше по IPv6-маршрутизации.

Пример:

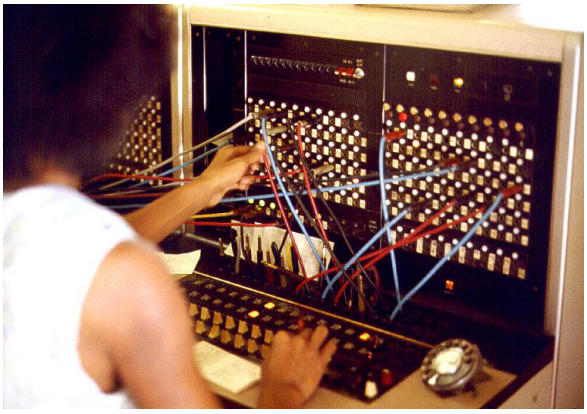
```
ping 8.8.4.4
```

Пингуем вторичный DNS-сервер 8.8.4.4 от компании Google. Именно благодаря anycast очень большое число абонентов могут использовать услуги этого сервиса.

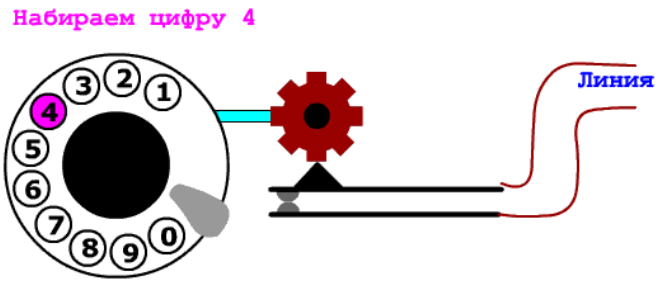
Виды коммутации

Коммутация — процесс передачи информации между узлами (интерфейсами узла) согласно таблице коммутации. Разберем виды коммутации.

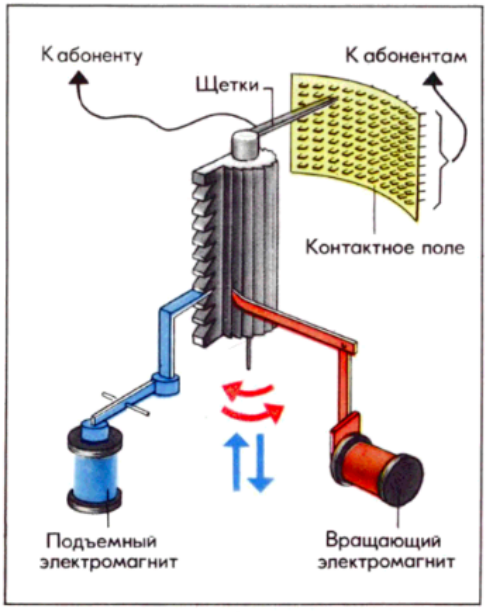
Ручная коммутация. Использовалась на ранней стадии развития телефонии: «Барышня, соедините». При этом роль коммутатора играла сама «барышня», переключая провода на панели коммутации.



Механическая коммутация. Позволяет заменить барышень электромеханическими приборами, упрощая жизнь большинству людей.

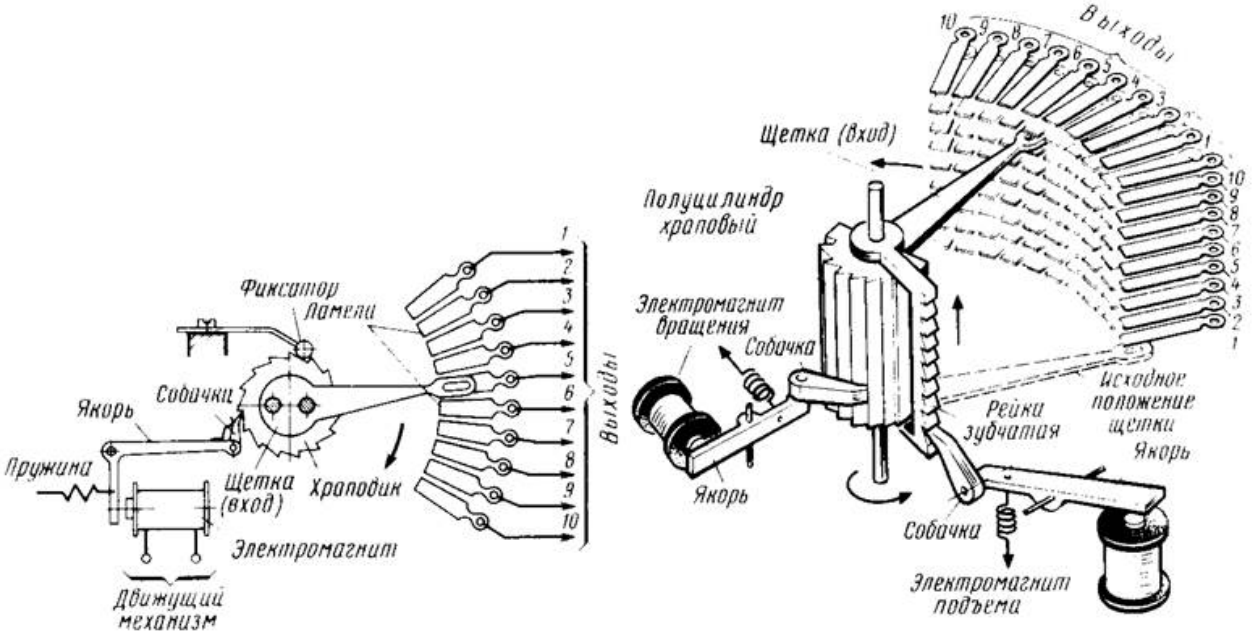


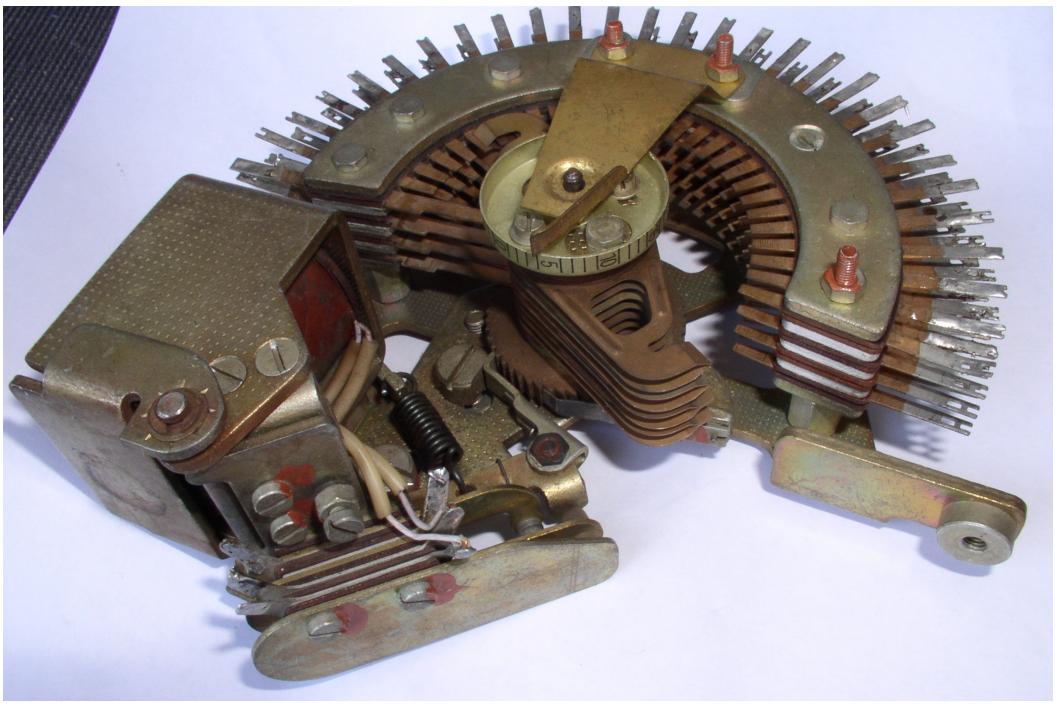
Принцип работы импульсного дискового номеронабирателя см. здесь.



Декадно-шаговый искатель

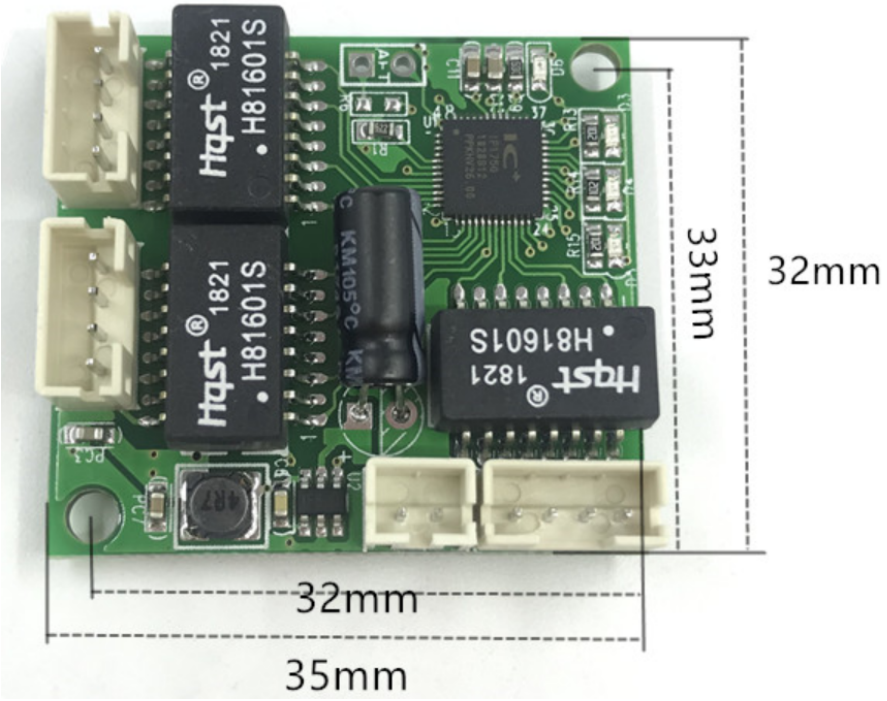
При электромеханической коммутации мы вращаем диск номеронабирателя. В зависимости от выбранной цифры механическое реле генерирует серию импульсов, (1 — 1 «щелчок», 2 — 2 «щелчка», ... 0 — 10 «щелчков») которую на АТС интерпретирует декадно-шаговый искатель и механически коммутирует линию на контактном поле. Декадно-шаговый искатель позволяет обслуживать до 100 абонентов, при большем числе абонентов используются каскады декадно-шаговых искателей.



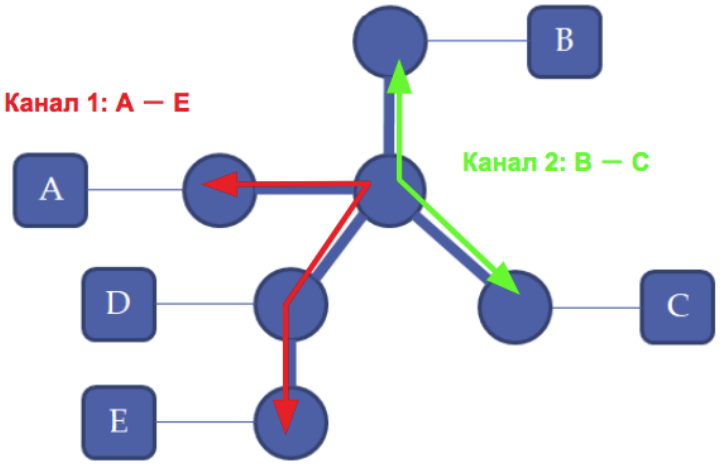


Декадно-шаговый искатель

Автоматическая коммутация. Дальнейшее развитие техники и использование цифровой телефонии позволило добиться коммутации без движущихся частей. Использование цифровых сигнальных процессоров и программируемых логических интегральных схем, а в более сложных устройствах — специализированных ЭВМ, позволяет выполнять коммутацию полностью автоматически.



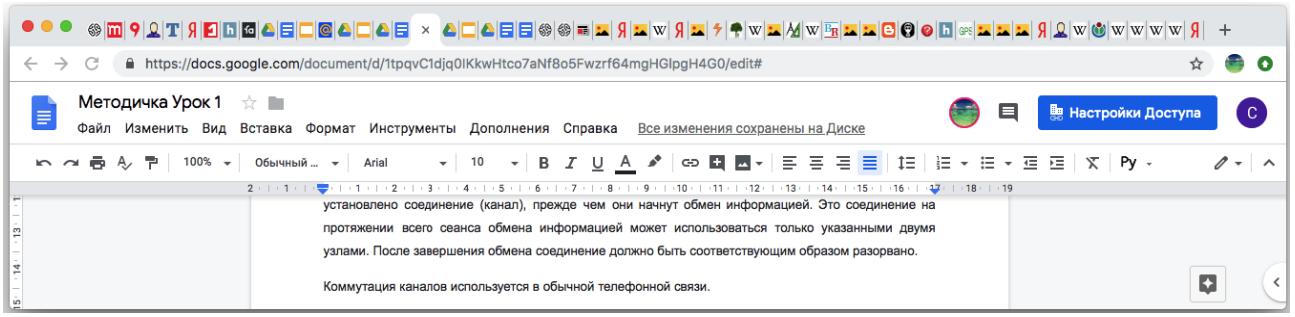
Коммутация каналов. Все вышеперечисленные случаи описывают коммутацию каналов.



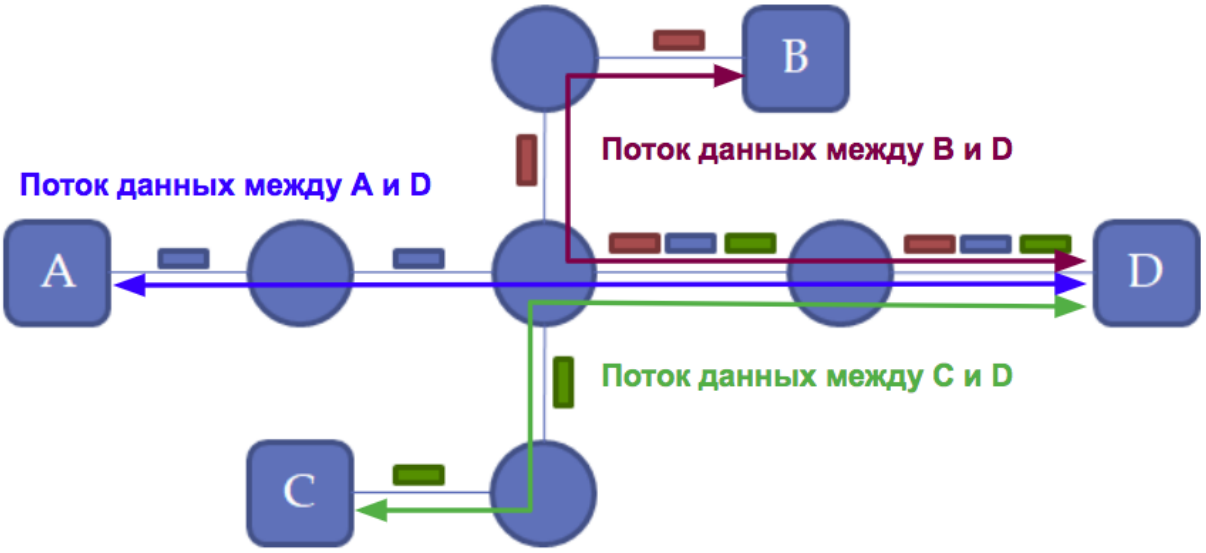
Коммутация каналов — вид коммутации, при которой между двумя узлами сети должно быть установлено соединение (канал), прежде чем они начнут обмен информацией. Это соединение на протяжении всего сеанса обмена информацией может использоваться только указанными двумя узлами. После завершения обмена соединение должно быть соответствующим образом разорвано.

Коммутация каналов используется в обычной телефонной связи.

На картинке мы видим, что узел E общается с узлом A, а узел C с узлом B. При необходимости связаться с узлом D узлу C придется сначала разорвать связь с узлом B. А если узел B пожелает пообщаться с узлом A, то необходимо, чтобы кроме узла B канал освободил также и узел A. В современном мире требуется получение информации сразу и одновременно от многих источников. Представьте, каково было бы использовать в старые времена Internet Explorer со множеством вкладок, если бы использовалась коммутация каналов.



Коммутация пакетов — при такой коммутации информация от каждого устройства делится на небольшие пакеты, и данные передаются по одним и тем же физическим каналам.



Как видно из рисунка, узел D может одновременно общаться с узлами A, B и C, не разрывая связь с каждым из них. Таким образом, можно одновременно слушать потоковую музыку, закачивать несколько разных файлов с разных серверов и общаться в мессенджере. Стек TCP/IP ориентируется на пакетную коммутацию.

Классификация сетей

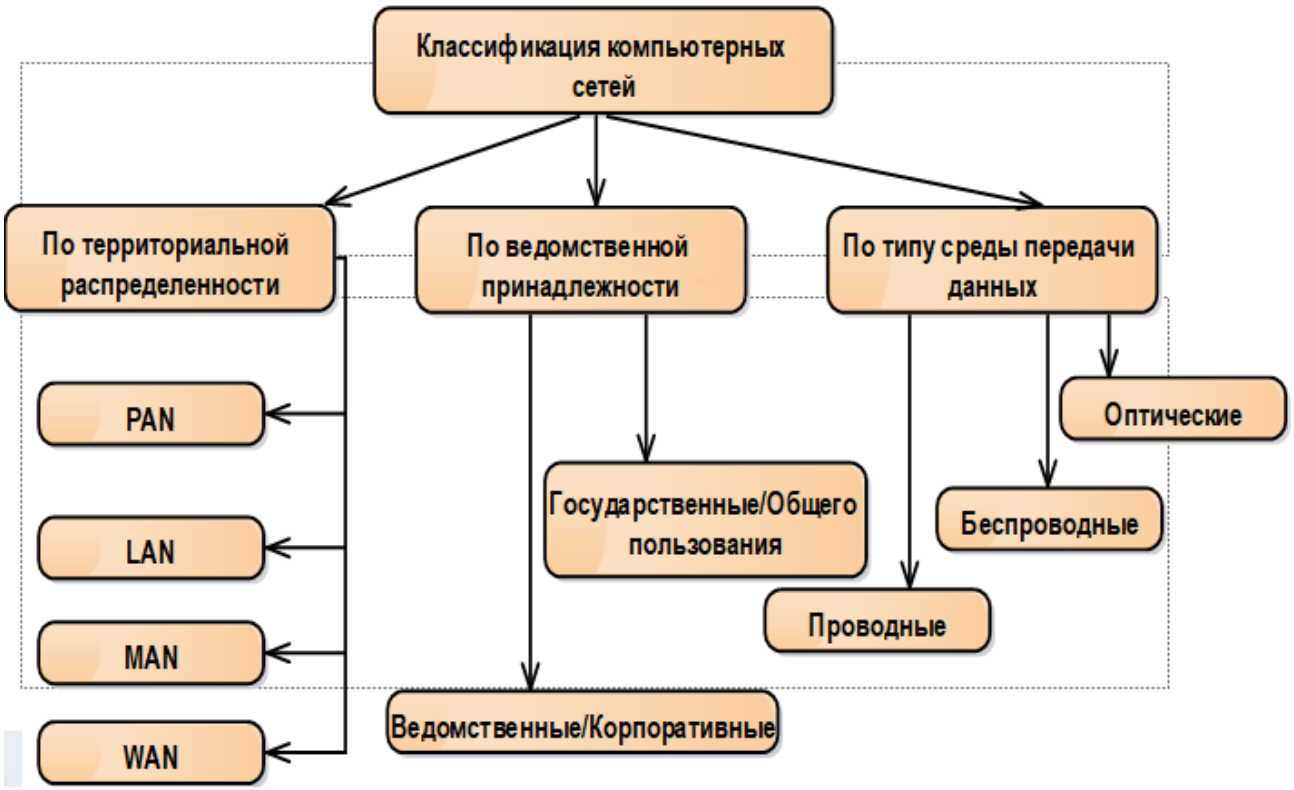
Легко вспомнить, что есть радиосети, телевизионные и компьютерные сети, беспроводные (многие пользуются Wi-Fi) и проводные.

Среди сетей есть государственные и ведомственные (используемые военными, спецслужбами), корпоративные (сети предприятий), частные сети (например, домашняя сеть ноутбуков и мобильных устройств). Кроме того, есть сети маленькие (домашняя сеть), относительно большие (сеть провайдера), и Интернет (и его аналоги, такие как Кванмен в КНДР).

Разберем способы классификации сетей (заучивать это не нужно).

Классификация сетей по типу предоставляемых сервисов. На сегодняшний день параллельно продолжают существовать:

1. Сети фиксированной связи или телефонные сети общего пользования (ТФоП или PSTN).
2. Сети передачи данных (теле- и радиосети).
3. Компьютерные сети.
4. Мобильные сети.



По территориальной принадлежности:

1. **PAN (Personal Area Network)** — персональная сеть, предназначенная для взаимодействия различных устройств, принадлежащих одному владельцу. Пример — стандарт IEEE 802.15 WPAN / Bluetooth. WPAN — Wireless PAN. Стек Bluetooth широко применяется для соединения нескольких устройств, таких как смартфоны, SmartTV и т. д.
2. **LAN (Local Area Network)** — локальные сети, имеющие замкнутую инфраструктуру до выхода на поставщиков услуг. Термин «LAN» может описывать и маленькую офисную сеть, и сеть большого завода, занимающего несколько сотен гектаров. Это сети закрытого типа, доступ к ним разрешен только ограниченному кругу пользователей, для которых работа в такой сети

непосредственно связана с их профессиональной деятельностью. Как правило, используются немаршрутизируемые серые адреса (10.0.0.0/172.16.0.0/192.168.0.0/100.64.0.0). К таковым относятся большинство домашних сетей (устройства в квартире, подключенные к беспроводному маршрутизатору с помощью Wi-Fi IEEE 802.11 или Ethernet — IEEE 802.3) и корпоративных сетей.

3. **MAN (Metropolia Area Network)** — компьютерная сеть, которая соединяет пользователей с компьютерными ресурсами в географической области (регионе), большей, чем та, которая покрыта локальной сетью (ЛВС), но меньшей, чем область, охваченная глобальной сетью (WAN). Проще говоря, это городские сети. Для реализации MAN разрабатывались стандарты IEEE 802.6 MAN на основе оптических сетей FDDI (не получил широкого распространения и считается устаревшим) и IEEE 802.16 WMAN (Wireless MAN) и WiMAX с поддержкой ячеистой (MESH) технологии. В настоящее время к сетям MAN можно отнести провайдерские сети, использующие Ethernet (проводной и оптический) и GPON (оптический).
4. **WAN (wide area network)** — телекоммуникационная или компьютерная сеть, охватывающая большие территории. К ним относится Интернет.

Сетевые модели

OSI/ISO	TCP/IP (DOD)
7. Прикладной уровень	4. Уровень приложений
6. Уровень представления	
5. Сеансовый уровень	
4. Транспортный уровень	3. Транспортный уровень
3. Сетевой уровень	2. Сетевой уровень
2. Канальный уровень	1. Уровень сетевых интерфейсов
1. Физический уровень	

Уровни модели OSI/ISO и их соответствие модели TCP/IP

Существуют две популярные многоуровневые сетевые модели: OSI/ISO (ЭМВОС) и TCP/IP (DOD).

Идея многоуровневой модели и стека как ее реализации заключается в том, что логика работы сетевых механизмов изолируется на каждом конкретном уровне таким образом, что каждый уровень на одной машине работает с аналогичным на другой машине, не задумываясь о реализации нижестоящих уровней. Каждый уровень имеет интерфейс на уровень ниже, обращаясь к которому, он реализует собственные возможности. Соответственно, при необходимости организации надежной

доставки приложение открывает TCP-сокет, указывает IP-адрес и порт сервера, и в случае успешного подключения имеет надежный способ передачи в обе стороны, который использует для реализации прикладного протокола. Соответственно, на прикладном уровне можно не беспокоиться о тройном рукопожатии, управлении окном, ретрансмиссиях и дупликациях. Напротив, на транспортном уровне, например при реализации протокола TCP, не имеет значения, какой прикладной протокол используется выше, какие протоколы будут использоваться ниже. TCP хорошо работает и через Ethernet, и через Wi-Fi, если взять канальный уровень.

Сетевая модель OSI (Open Systems Interconnection Basic Reference Model) — базовая эталонная модель взаимодействия открытых систем, сокр. ЭМВОС; 1978 г) — абстрактная сетевая модель для коммуникаций и разработки сетевых протоколов. Это академическая попыткой создания универсальной модели взаимодействия систем. На практике модель потерпела поражение перед стеком TCP/IP, ставшим практической реализацией стека сетевых технологий. Тем не менее, модель OSI/ISO иногда бывает удобна для описаний, и нижние 4 уровня применяются для описания сетевых устройств.

OSI/ISO расшифровывается как Open Systems Interconnection Basic Reference model (OSI model). Модель является стандартом ISO (ISO/IEC 7498-1). Есть также отечественное соответствие данному стандарту, ЭМВОС — эталонная модель взаимодействия открытых систем, и отечественный стандарт ГОСТ Р ИСО/МЭК 7498-1-99). Разработана в 1978 году и состоит из 7 уровней, которые частично соотносятся с действительно применяемыми технологиями.

Данная модель не вполне соответствует используемому стеку технологий TCP/IP, но часто применяется:

- в преподавании сетевых технологий (в учебниках, таких как «Компьютерные сети» Э. Таненбаума и В. Олифера);
- государственным заказчиком и государственными структурами/ведомствами;
- нижние уровни (L1–L4) используются производителями сетевого оборудования и сетевыми инженерами (например Cisco);
- не всегда модель TCP/IP удобно использовать, когда речь идет о протоколах шифрования;
- работодатели на собеседовании могут задавать вопросы о модели OSI/ISO.

Несмотря на то, что стек TCP/IP — стандарт, используемый в современных сетях, модель OSI/ISO также необходимо знать, хотя бы на уровне названия и назначения семи уровней модели OSI/ISO (ЭМВОС).

OSI/ISO	**Задачи**	**Данные или PDU**
7. Прикладной уровень	Доступ приложений к сетевым службам	Данные
6. Уровень представления	Представление и кодирование данных	Данные

5. Сеансовый уровень	Установка и управление сеансом связи, завершение и возобновление сеанса связи	Данные
4. Транспортный уровень	Надежное соединение точка — точка	Сегмент или дейтаграмма
3. Сетевой уровень	Определение пути и логическая адресация	Пакет (реже дейтаграмма)
2. Канальный уровень	Физическая адресация и целостность передаваемых данных через физическую среду	Кадр (фрейм)
1. Физический уровень	Кодирование сигнала и передача данных через физическую среду	Биты

Рассмотрим назначение уровней:

- Прикладной уровень предоставляет интерфейс к сетевым услугам для прикладного программного обеспечения. Сам же он запрашивает интерфейс у уровня представления.
- Уровень представления предназначен для перекодирования и преобразования форматов данных. Запрашивает интерфейс у сеансового уровня.
- Сеансовый уровень необходим для установки и управления сеансами связи. Запрашивает интерфейс у транспортного уровня.
- Транспортный уровень предназначен для передачи данных от отправителя к получателю, и предоставляет возможность управления надежностью передачи. Запрашивает интерфейс у сетевого уровня.
- Сетевой уровень предназначен для глобальной логической адресации узлов и осуществления маршрутизации. Сетевой уровень запрашивает интерфейс у канального уровня.
- Канальный уровень предназначен для взаимодействия устройств через физическую среду передачи, принимая нужные кадры или фреймы (так именуются блоки данных вместе с заголовками на канальном уровне), отбрасывая ненужные и контролируя целостность.
- Физический уровень представлен физическими способами передачи информации. Витая пара (две пары, четыре пары), коаксиал, оптоволокно, радиоканал и т. д.

Протоколы TCP/IP не имеют однозначного сопоставления с моделью OSI/ISO.

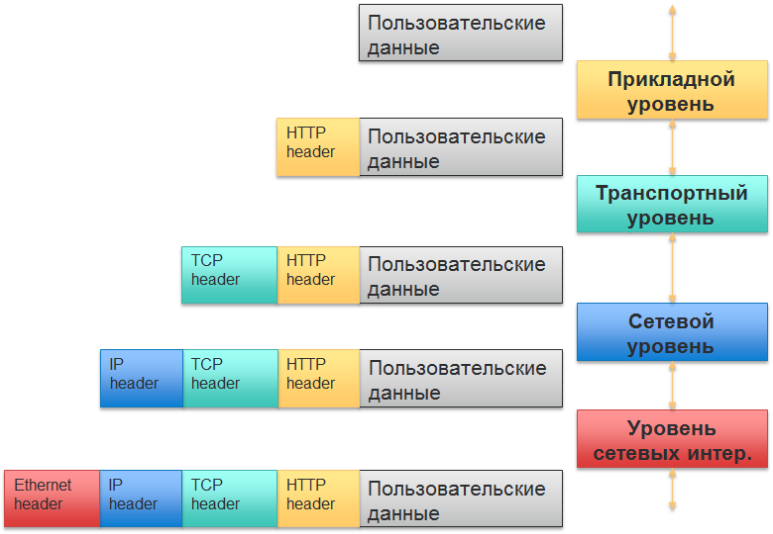
Иногда протоколы маршрутизации RIP и BGP относятся к прикладному уровню, но чаще к сетевому. Протокол DNS относится к прикладному, но иногда его относят к сеансовому уровню. Протоколы SSL и TLS чаще всего относятся к уровню представления, но иногда и к сеансовому; транспортный протокол TCP также обладает сеансовыми чертами. Протокол ARP относят либо к сетевому, либо к канальному уровню (на самом деле он занимает промежуточное положение). Консенсус есть относительно протоколов канального уровня (IEEE 802.3 Ethernet, IEEE 802.11 Wi-Fi, ITU G.992.1 ADSL, ITU G 991.1 HDSL), основных протоколов сетевого уровня (IP, ICMP, IGMP), протоколов транспортного уровня (TCP, UDP). Большинство протоколов относятся к прикладному уровню

(FTP/FTPS, SFTP/SSH, HTTP/HTTPS, SMTP, POP3, IMAP4 и т. д.). Тунеллирующие протоколы часто относят к сеансовому уровню, хотя их местоположение в стеке OSI/ISO не может быть строго определенным из-за дублирования уровней модели OSI/ISO внутри тоннеля и вовне (PPTP, L2TP, OpenVPN).

Схема расположения протоколов в модели OSI/ISO может быть такой:

Вторая модель, которая будет нам более полезна, — TCP/IP. Модель DOD, или TCP/IP — модель сетевого взаимодействия, разработанная Министерством обороны США и названная по имени Министерства (DOD — Department of Defence), практической реализацией которой является стек протоколов TCP/IP.

Сетевой протокол — набор правил и действий (очередности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включенными в сеть устройствами.



Эта модель появилась до модели OSI и независимо от нее при разработке ARPANET и быстро завоевала популярность благодаря своей простоте. Модель OSI оказалась громоздкой и слишком долго разрабатывалась, поэтому основной сейчас является TCP/IP. Декомпозиция подразумевает интерфейсное и межпротокольное взаимодействие. Оно организуется с помощью инструмента инкапсуляции.

PDU (Protocol Data Unit), или протокольный блок данных — название блока данных независимо от используемого уровня модели OSI.

Инкапсуляция — механизм языка программирования, ограничивающий доступ к составляющим объект компонентам (методам и свойствам), делает их приватными, то есть доступными только внутри объекта.

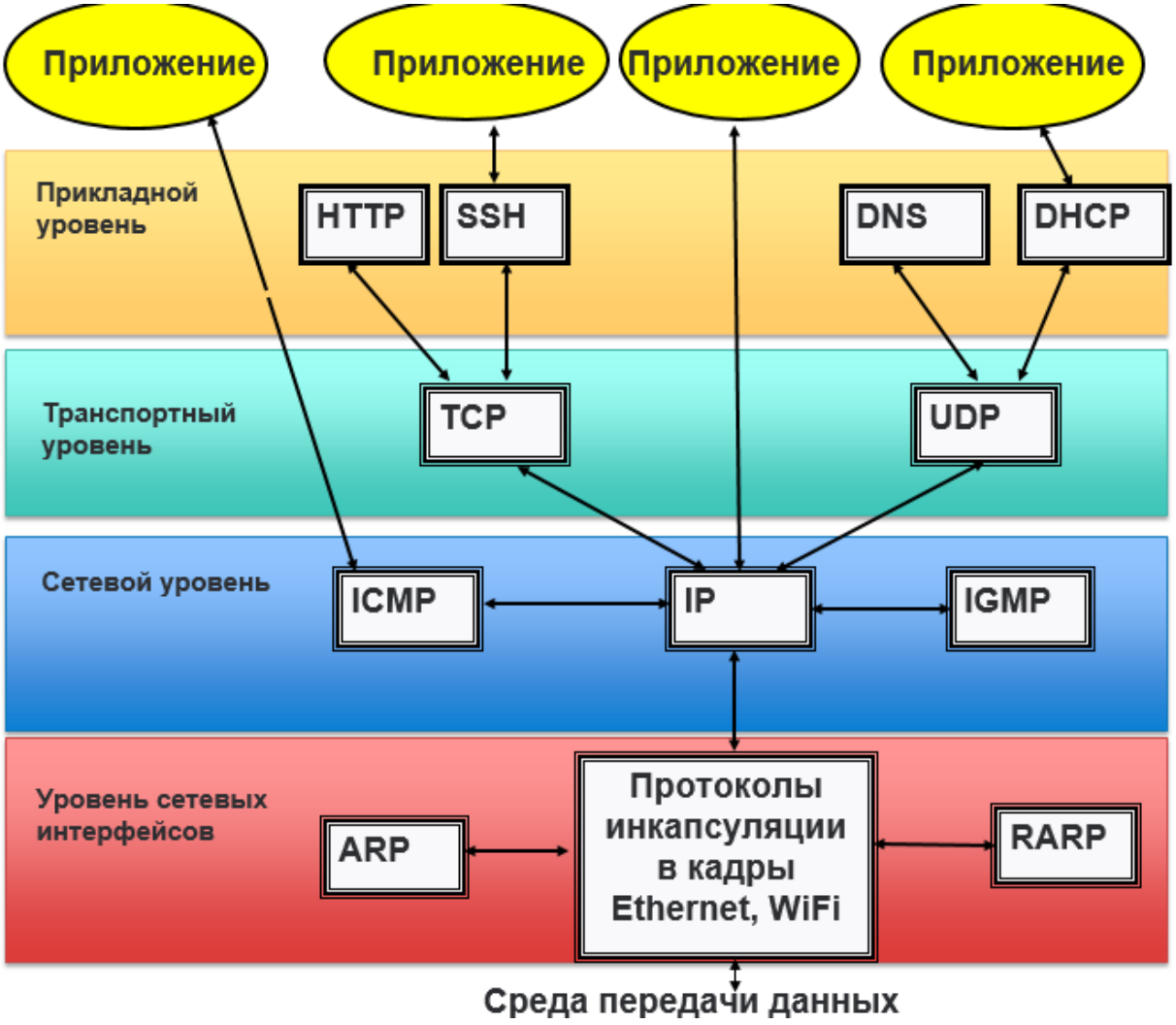
В сетевых технологиях инкапсуляция означает помещение PDU-протокола вышестоящего уровня в поле данных PDU-протокола нижестоящего уровня, «обертывание» его более низкоуровневым заголовком, упаковка его в PDU нижестоящего уровня. Это можно сравнить с матрешкой: меньшую матрешку (более высокий уровень) вкладывают в оболочку более крупной матрешки (более низкий уровень). Обратный процесс называется декапсуляцией. Подобный механизм позволяет сосредоточиться на протоколах одного уровня, не затрагивая механизмы работы других протоколов. Так разработчики и инженеры сетевого оборудования могут не зависеть от способов взаимодействия прикладных протоколов (используемых приложениями). И наоборот, прикладные протоколы смогут работать через самые разные протоколы нижних уровней (например, как через Wi-Fi, так и через Ethernet). Таким образом обеспечивается прозрачность сетевой передачи.

Стек TCP/IP

Разберем подробнее стек TCP/IP.

Стек протоколов TCP/IP — набор сетевых протоколов передачи данных, используемых в сетях, включая Интернет. Название TCP/IP состоит из названий двух наиважнейших протоколов семейства — Transmission Control Protocol (TCP) и Internet Protocol (IP), которые были разработаны и описаны первыми в данном стандарте. Также изредка упоминается как модель DOD в связи с историческим происхождением от сети ARPANET в 1970-х годах (под управлением Department of Defence, Министерства обороны США).

Уровни протоколов TCP/IP расположены по принципу стека (англ. stack — стопка) — это означает, что протокол, располагающийся на уровне выше, работает «поверх» нижнего, используя механизмы инкапсуляции. Например, протокол TCP работает поверх протокола IP.



В самом низу находится уровень сетевых интерфейсов, объединяющий физический и канальный уровень модели OSI. Пример: интерфейс Ethernet, описывающий передачу данных по коаксиальному кабелю или витой паре. Протоколы этих уровней обычно реализуются на аппаратном уровне, например в сетевой карте компьютера.

Выше идет межсетевой уровень, соответствующий сетевому уровню модели OSI и часто называемый сетевым. На этом уровне работает протокол IP (Internet Protocol), описывающий структуру сети и

доставку пакетов. На межсетевом уровне решается вопрос идентификации хостов и маршрутизации. Основной протокол — IP (Internet Protocol). Данные снабжаются заголовком, где указываются IP-адреса получателя и отправителя. На сетевом уровне обычно говорят о IP-пакетах или IP-дейтаграммах (реже). Это синонимы. Также к сетевому уровню относятся вспомогательные протоколы: ICMP (для передачи сообщений о сетевых проблемах), IGMP (для управления группами широковещательных рассылок, например, для потокового вещания IPTV), ARP (для преобразования IP-адресов в MAC-адреса), RARP (расширение протокола ARP для назначения машине IP-адреса по ее MAC-адресу; был вытеснен протоколом DHCP), протоколы маршрутизации. При этом ICMP вкладывается в IP-пакет. Некоторые протоколы маршрутизации технически могут быть выполнены как протоколы прикладного уровня (использовать интерфейс транспортного уровня TCP, UDP) и при этом входить в механизм сетевого уровня. Протокол ARP работает сразу поверх канального уровня и служит для поиска MAC-адресов для искомых IP-адресов своей сети (или шлюза). Протоколы ARP и RARP занимают промежуточное положение между двумя нижними уровнями, иногда их относят к уровню сетевых интерфейсов (канальному уровню).

Еще выше — транспортный уровень, где находятся протокол TCP (Transmission Control Protocol), использующийся для передачи данных с гарантированной доставкой, и протокол UDP (User Datagram Protocol) — для передачи данных с негарантированной доставкой. Эти протоколы обычно реализуются на уровне операционной системы. На транспортном уровне решается вопрос идентификации приложений и надежной/ненадежной доставки. И TCP, и UDP в заголовке содержат двухбайтный порт получателя и порт отправителя. Существует два набора портов: TCP-порты (в системе именуются STREAM) и UDP-порты (DGRAM). Именно по номеру порта операционная система понимает, какому из работающих приложений должны быть доставлены полученные системой данные.

На самом верху находится множество протоколов прикладного уровня, выполняющих конкретные задачи. Обычно они программируются в отдельных приложениях. Большинство протоколов, используемых приложениями, — прикладного уровня: HTTP, FTP, SSH/SFTP, SMTP, POP3, IMAP4, XMPP, а также DNS, DHCP, NTP, SNMP. По большей части любое приложение может реализовать свой протокол прикладного уровня. Прикладные протоколы используют тот или иной протокол транспортного уровня в зависимости от решаемых задач. Если нужна надежная передача файлов большого объема (архив, приложения) либо механизм сессии (как ssh), используется TCP. Если нужна отправка коротких сообщений или мультимедиа-потока, не критичного к потерям пакетов, — UDP.

IP — протокол, лежащий в основе Интернета, его название так и расшифровывается: Internet Protocol.

В настоящее время используются следующие две версии протокола IP:

- IPv6 — активно внедряемая в работу с 2010 года (текущая версия спецификации опубликована в декабре 1998). IP-адрес имеет разрядность 128 бит и записывается в виде восьми 16-битных полей, с использованием шестнадцатеричной системы счисления и с

возможностью сокращения двух и более последовательных нулевых полей до двух двоеточий подряд. Пример: 2001:db8:42::1337:cafe.

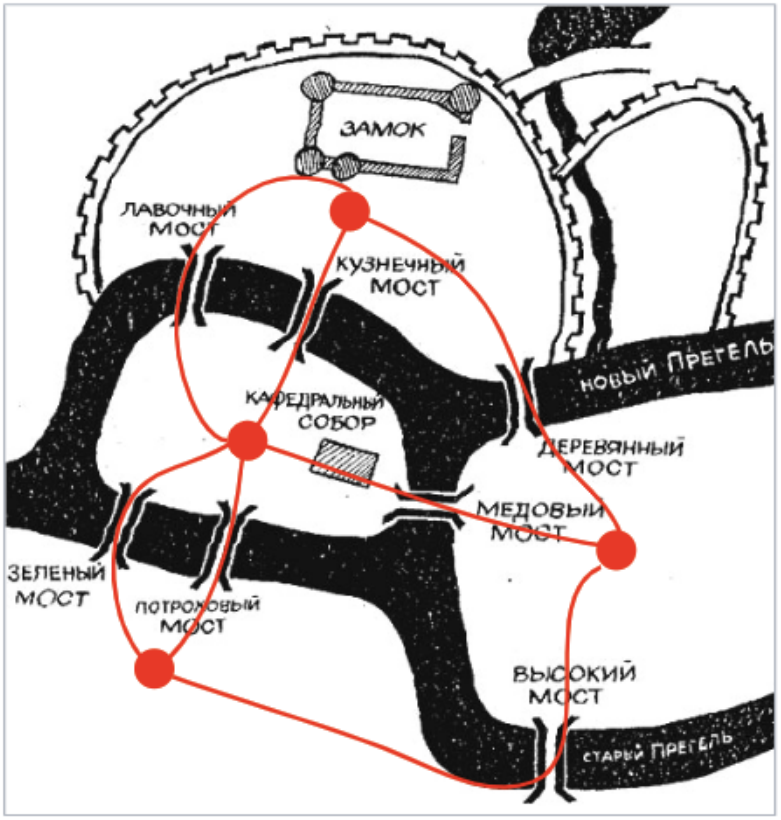
- IPv4 — «классическая» версия (1981 г.). IP-адрес имеет разрядность 32 бита и записывается в виде четырех десятичных чисел в диапазоне 0…255 через точку. Пример: 192.0.2.34.

Каждый узел может напрямую связаться только с узлами своей сети (например, подключенными к тому же сегменту Ethernet). Для определения узлов используется адрес сети — часть IP-адреса, определяемая маской сети. Связь с узлами других сетей осуществляется через промежуточные узлы — маршрутизаторы.

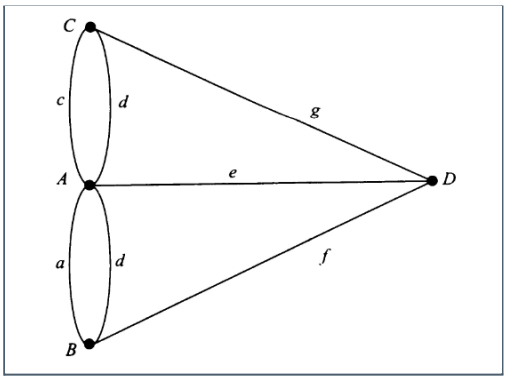
Взаимодействие стека TCP/IP и модели OSI/ISO можно изложить следующим образом. Стек TCP/IP это набор инкапсулируемых протоколов, используемых в современной сети Интернет. При этом 4 уровней стека TCP/IP не всегда достаточно для описания существующих протоколов (так, TLS не помещается в модель TCP/IP занимая место между 3 и 4 уровнем, а MPLS — между 2 и 3). Модель OSI/ISO не описывает действующую последовательность выполнения задач (например, TCP имеет черты не только 4 уровня, но и 5, TLS занимает 5 и 6 уровень, протокол HTTP, будучи протоколом 7 уровня, выполняет также и задачи 5 и 6 уровня и т. д). Но модель OSI/ISO описывает задачи телекоммуникаций — другое дело, что выполняться они могут не в последовательности уровней OSI/ISO и на разных уровнях модели TCP/IP.

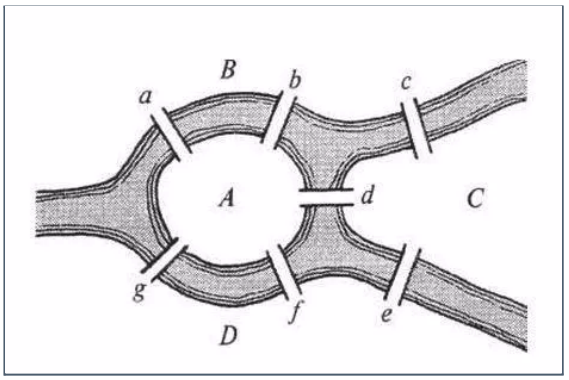
Графы и топологии

Прежде чем изучить сетевую топологию, нужно немножко математики, а именно основы теории графов. Концепцию графов сформулировал Леонард Эйлер в попытке решить задачу о кёнигсбергских мостах.



Жителей Кёнигсберга занимала одна задача: как пройти по всем городским мостам через реку Прегель, не проходя ни по одному из них дважды. Леонард Эйлер доказал, что такого маршрута не существует. Если посчитать сушу за вершины и провести соединяющие точки линии там, где можно пройти через мосты, мы получим такую схему:



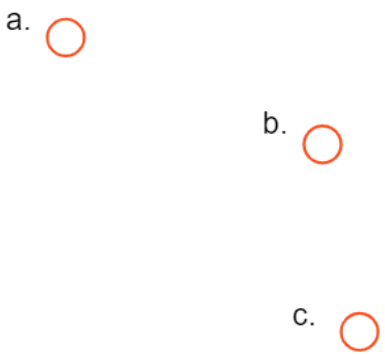


Такая схема — пример графа (строго говоря, одного из обобщений графа — мультиграфа, где вершины могут быть соединены кратными — «дублирующими» — ребрами).

Граф определяется как множество объектов (вершин) и связей (ребер) между ними.

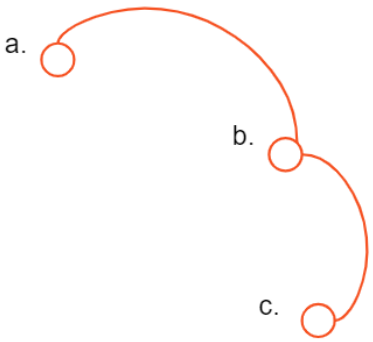
Вершина — некоторый объект. Это может быть человек, компьютер, таблица в базе данных и т. д.

Граф может состоять только из вершин. Такой граф называется **пустым графом**, нуль-графом или нулевым графом.

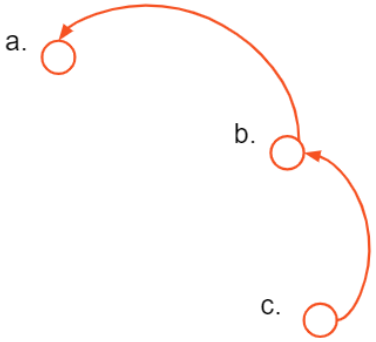


Пустой граф из трех вершин

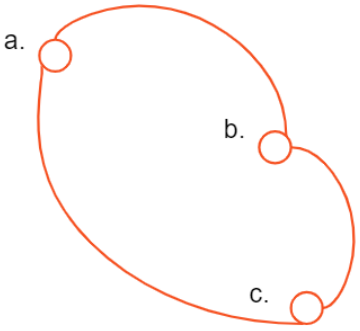
Ребро — связь между вершинами. Рёбра могут быть неориентированными и ориентированными (проходимыми в одном направлении; такие рёбра называются дугами).



Граф с неориентированными ребрами



Граф с ориентированными ребрами (дугами)

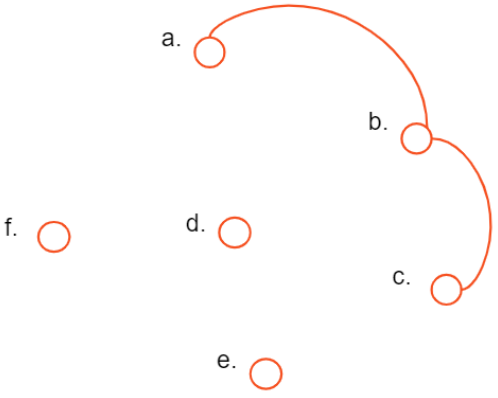


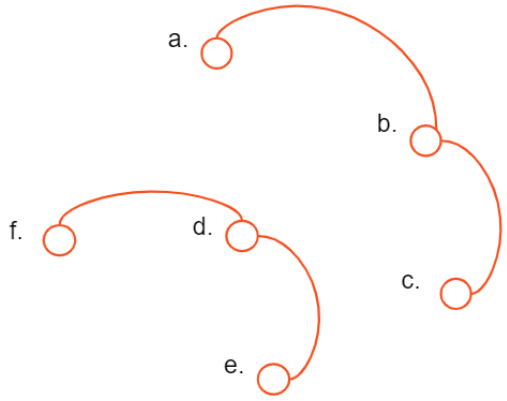
Цикл — последовательность вершин, начинающаяся и заканчивающаяся в одной и той же вершине, каждые две последовательные вершины в которой смежны.

<< Пример цикла

Изолированная вершина — вершина, не имеющая ребер.

Пример изолированных вершин >>





Несвязный граф — граф, в котором имеются, как минимум две области, не связанные между собой ребрами.

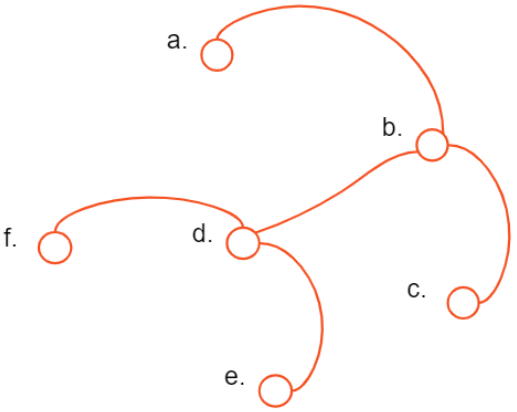
Для неориентированного графа это означает, что есть две точки, такие, что из одной в другую попасть нельзя.

<< Пример несвязного графа

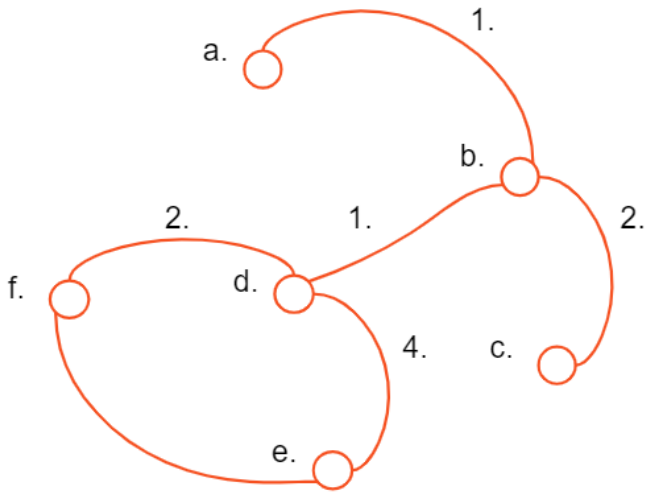
Связный граф — граф, в котором все вершины связаны.

Для неориентированного графа из любой вершины можно попасть в любую другую.

Дерево — связный граф, не содержащий циклов.



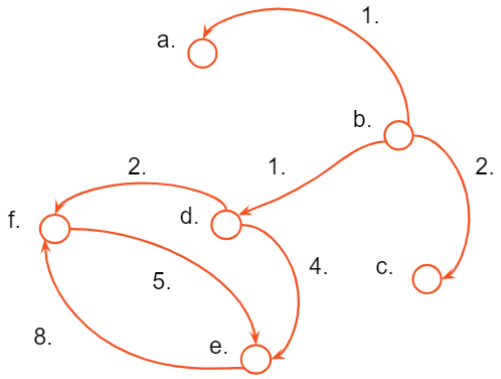
Пример связного графа >>
(также является и деревом)



<< Пример связного графа с циклами (деревом не является).

Также данный пример является примером взвешенного графа.

Взвешенный граф — граф, каждому ребру которого поставлено в соответствие некое значение (вес ребра).

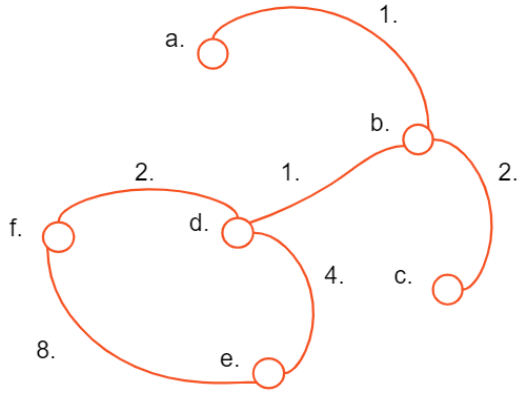


Ориентированный граф (орграф) — граф или мультиграф, ребрам которого присвоено направление. Направленные рёбра именуются также **дугами**, а в некоторых источниках и просто ребрами.

Примерами ориентированных графов являются картинки, иллюстрирующие виды связи (симплекс, дуплекс, полудуплекс, юниксат, бродкаст).

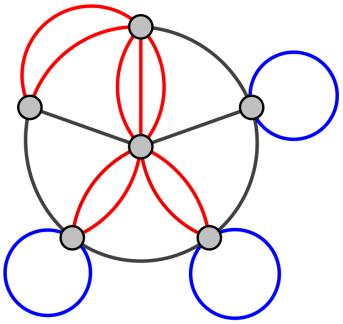
<< Пример ориентированного взвешенного графа (у каждой дуги есть вес)

Неориентированный граф — граф, ни одному ребру которого не присвоено направление. Также может называться неорграфом.



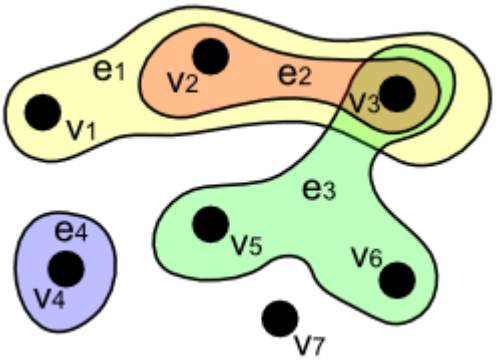
Пример неориентированного взвешенного графа >>

Мультиграф — граф, в котором разрешается присутствие кратных ребер (их также называют «параллельными»), то есть ребер, имеющих те же самые конечные вершины. Таким образом, две вершины могут быть соединены более чем одним ребром (тем самым мультиграфы отличаются от гиперграфов, в которых каждое ребро может соединять любое число вершин, а не в точности две).



Задача о кёнигсбергских мостах и топологии с дублирующими каналами являются мультиграфами.

Гиперграф — обобщение графа, в котором каждым ребром могут соединяться не только две вершины, но и любые подмножества вершин. Такое ребро называется гиперребром.



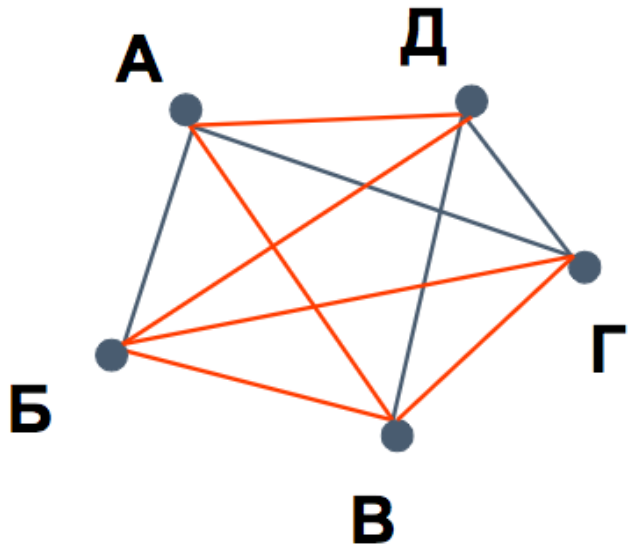
Гиперграфы применяются, в частности, при моделировании электрических цепей.

Топология «шина» как раз является гиперграфом. Соединяющий провод (шина) является гиперребром.

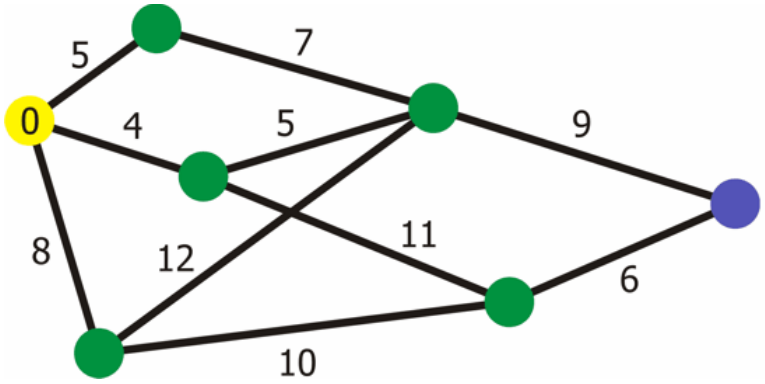
<< Пример гиперграфа:
Вершины v_1, v_2, v_3 объединены гиперребром e_1, v_3, v_5,v_6 — гиперребром e_3.

Графы позволяют решать задачи и головоломки, например такие: Артемий, Бонифаций, Василий, Георгий и Дмитрий обменялись рукопожатиями. Каждый пожал руку другому только по одному разу. Сколько рукопожатий было?

Изобразив людей в виде вершин, а рукопожатия в виде ребер и подсчитав число ребер мы легко получим правильный ответ: 10



Алгоритмы на графах имеют значительное применение в вычислительной технике. Алгоритм Дейкстры, в частности, используется в алгоритмах маршрутизации.



Смотреть анимацию по ссылке:
https://drive.google.com/file/d/1bUgx0qB8ZVHpYeN8cdxHD-qccHlav8Z-/view.

Топологии

Сетевая топология — это конфигурация графа, вершинам которого соответствуют конечные узлы сети (компьютеры) и коммуникационное оборудование (коммутаторы, маршрутизаторы), а ребрам — физические или информационные связи между вершинами.

Сетевая топология может быть:

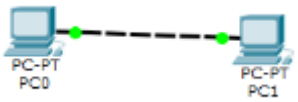
1. Физической — описывает реальное расположение и связи между узлами сети.
2. Логической — описывает прохождение сигнала в рамках физической топологии.
3. Информационной — описывает направление потоков информации, передаваемых по сети.
4. Управления обменом — принцип передачи права на пользование сетью.

Под топологией (компоновкой, конфигурацией, структурой) компьютерной сети обычно понимается физическое расположение компьютеров сети друг относительно друга и способ их соединения линиями связи. Важно отметить, что понятие топологии относится прежде всего к локальным сетям, в которых структуру связей можно легко проследить. В глобальных сетях структура связей обычно скрыта от пользователей и не слишком важна, так как каждый сеанс связи может проходить по собственному пути.

Топология определяет требования к оборудованию, тип используемого кабеля, допустимые и наиболее удобные методы управления обменом, надежность работы, возможности расширения сети. И хотя выбирать топологию пользователю сети приходится нечасто, знать об особенностях основных топологий, их достоинствах и недостатках надо.

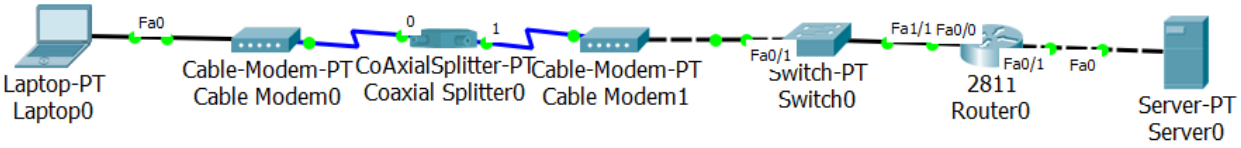
Базовые топологии сети:

Точка-точка/point-to-point — самая простая топология с выделенной линией связи между двумя конечными точками — например, два компьютера, соединенные кабелем непосредственно.

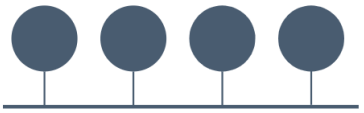


Сейчас сеть такой топологии образуется при туннельных соединениях. Также с точки зрения 7 уровня модели OSI/ISO соединение между клиентом и сервером — тоже точка-точка.

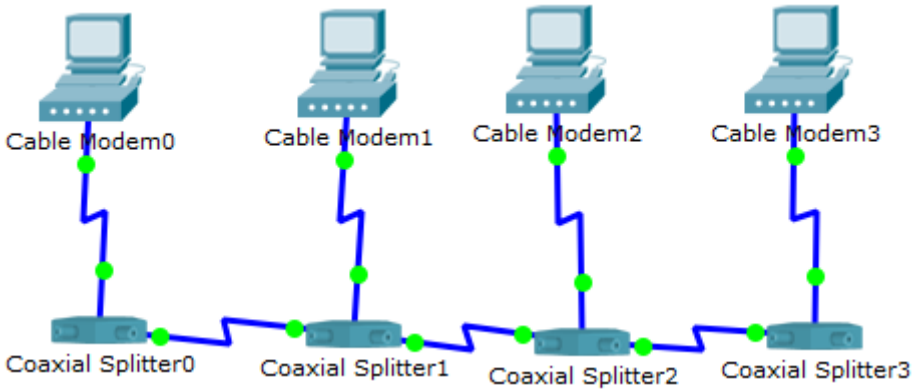
Цепь/circuit или линейная топология — сетевая топология, в которой все узлы сети подключены последовательно друг через друга.

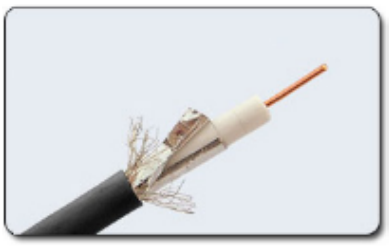


Цепью на логическом уровне можно считать цепочку из коммутаторов и маршрутизаторов. Как линейная топология выглядит трассировка маршрута с помощью команд tracert/traceroute.



Шина/bus — все компьютеры параллельно подключаются к одной линии связи. Информация от одного компьютера сразу же передается всем остальным. Каждый узел слышит остальные, и требуется решить две задачи: определить, нужно ли обработать входящее сообщение или отбросить его (если оно адресовано не нам), и обнаружить или предотвратить коллизии (попытку одновременной передачи сразу двумя или более узлами).



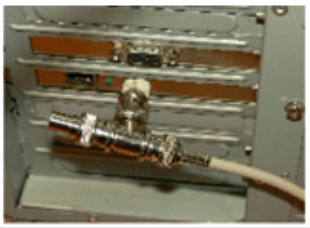


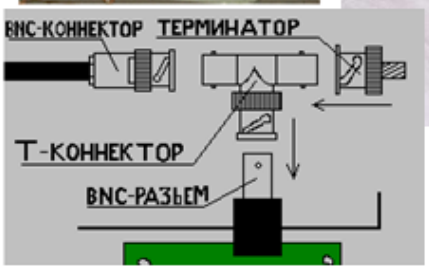
Коаксиальный кабель

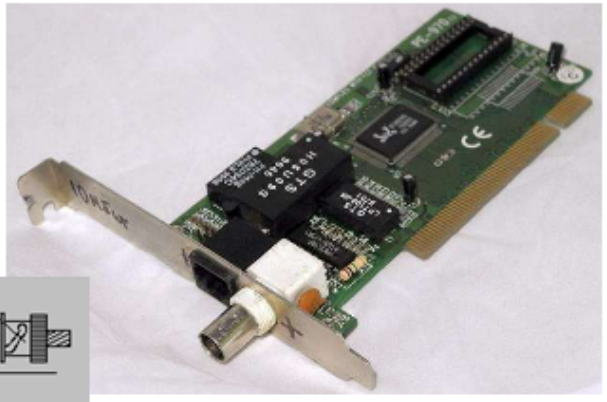
Т-коннектор BNC

Т-коннектор BNC

Терминатор - терминирующий резистор 50 Ом





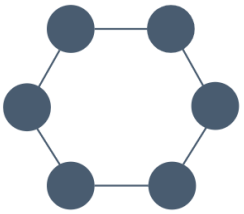


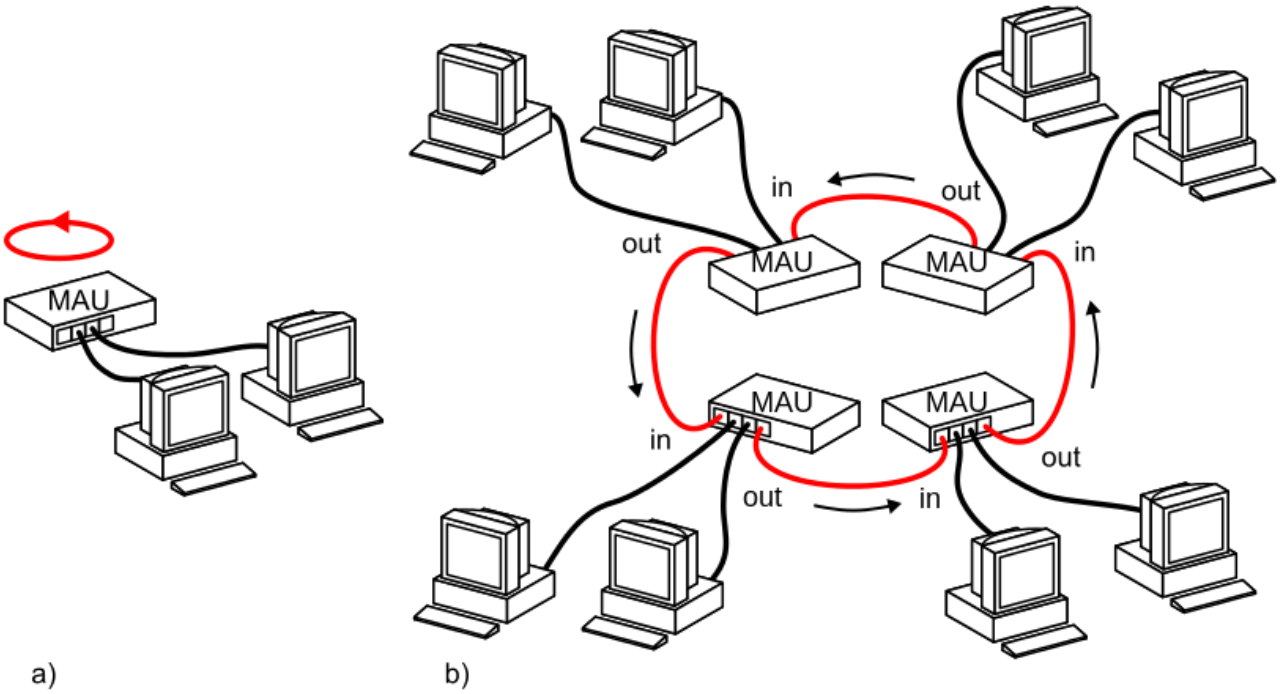
Сетевая карта с BNC-разъемом

В прошлом топология «шина» применялась для организации сети Ethernet по коаксиальному кабелю. Но и сейчас топология «шина» применяется в компьютерной технике: USB-шина, PCI-шина. Также шиной будет топология сети при использовании PLC (Power Line Communications).

Звезда/star — к одному центральному узлу присоединяются остальные периферийные компьютеры, причем каждый из них использует отдельную линию связи. Информация от периферийного компьютера передается только центральному узлу, от центрального узла — одному или нескольким периферийным компьютерам.

Кольцо/ring — компьютеры последовательно объединены в кольцо. Передача информации в кольце всегда производится только в одном направлении. Каждый из компьютеров передает информацию только одному — следующему в цепочке, а получает информацию — от предыдущего в цепочке.





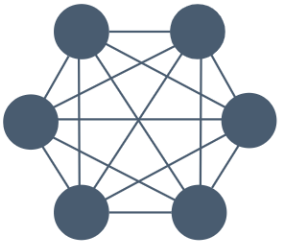
Источник изображения: By Andrew28913, https://en.wikipedia.org/wiki/File:Token_ring.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=33943276

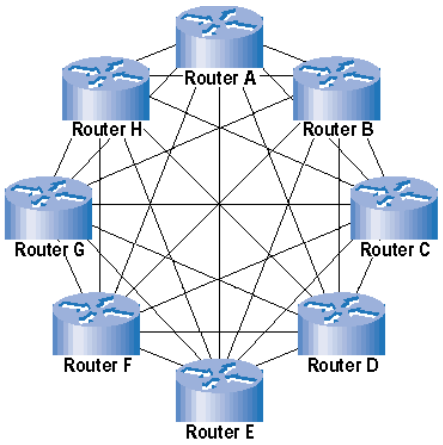
Пример кольцевой топологии на основе технологии Token Ring. Компьютеры подключаются к MAU — Media Access Unit. На рисунке A два компьютера объединены одним MAU. На рисунке B 8 компьютеров объединены по два благодаря четырем MAU. В данном случае каждый MAU имеет 4 порта: in, два для подключения компьютеров и out. Информация передается по кругу только в одном направлении — от in к out. MAU по очереди передают токен; тот MAU, у которого в данный момент токен, может передавать, остальные только слушать.

Сейчас кольцевая топология может быть использована при объединении коммутатров или маршрутизаторов в кольцо.

Mesh (ячеистая топология) — сетевая топология компьютерной сети, построенная на принципе ячеек, в которой рабочие станции сети соединяются друг с другом и способны принимать на себя роль коммутатора для остальных участников. Данная организация сети является достаточно сложной в настройке, однако при такой топологии реализуется высокая отказоустойчивость. Как правило, узлы соединяются по принципу «каждый с каждым». Обычно используется беспроводное оборудование. Часто применяется на массовых мероприятиях, в военном деле, в спутниковой связи. Применяются особые алгоритмы маршрутизации (ad-hoc-маршрутизации), такие как AODV или OLSR.

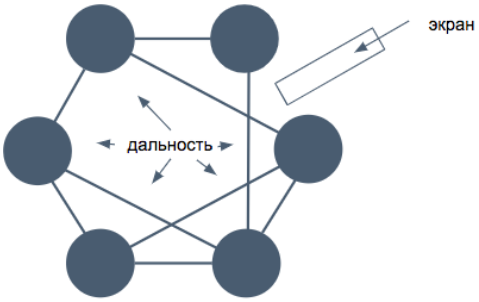
Полносвязная/fully connected mesh topology — сеть, в которой каждый компьютер непосредственно связан со всеми остальными. Однако этот вариант громоздкий и неэффективный, потому что каждый компьютер в сети должен иметь большое количество коммуникационных портов, достаточное для связи с остальными.

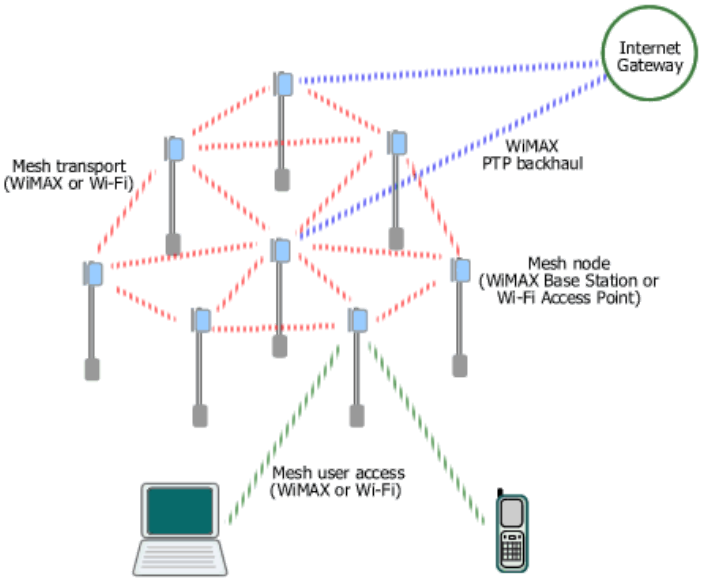




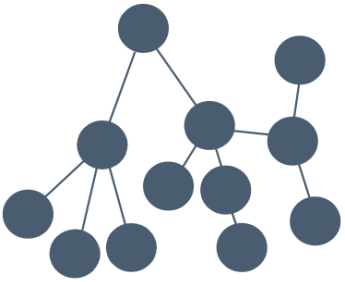
Полносвязные MESH-сети есть в беспроводной ячеистой топологии, например, когда каждый радиоузел «видит» все остальные.

Неполносвязная (ячеистая)/mesh — топология, аналогичная полносвязной, но в которой не все компьютеры соединены с каждым из остальных. Неполносвязных топологий существует несколько. В них, в отличие от полносвязных, может применяться передача данных не напрямую между компьютерами, а через дополнительные узлы.



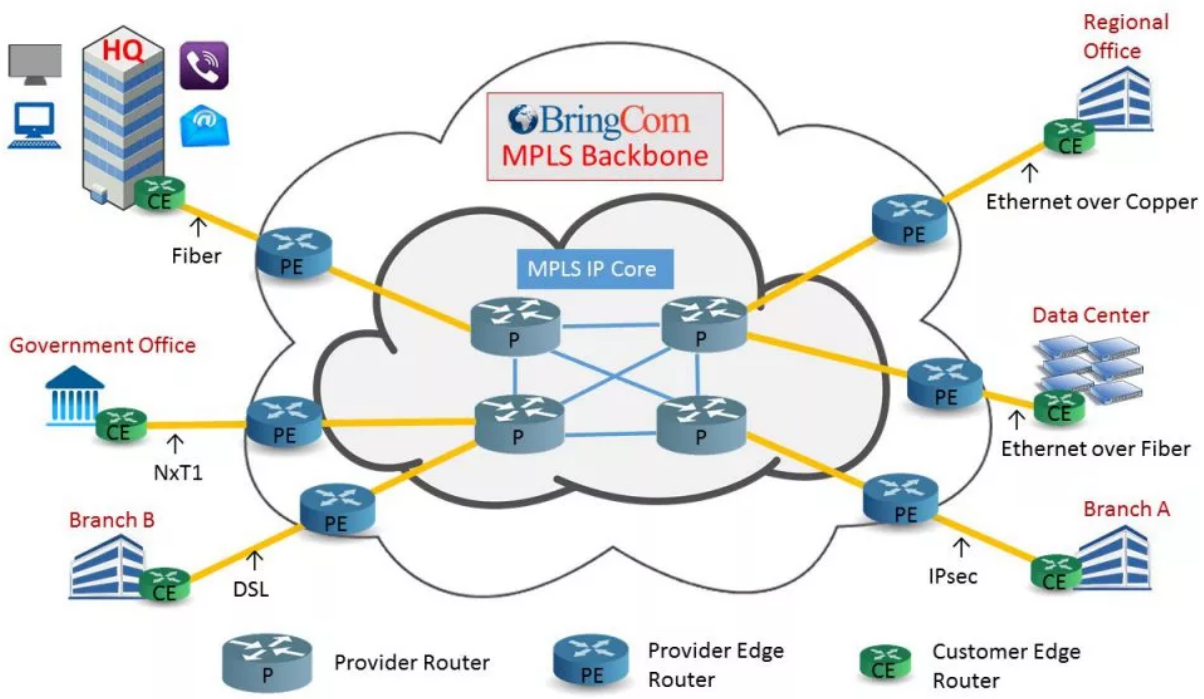


Дерево (иерархическая звезда) — сетевая топология, в которой каждый узел более высокого уровня связан с узлами более низкого уровня звездообразной связью, образуя комбинацию звезд.



Название «дерево» пришло из теории графов. Первый узел дерева принято называть корнем, следующие узлы высокого уровня — родительскими, а узлы более низкого уровня — дочерними. Таким образом, каждый дочерний узел, который имеет связь с более низкими узлами, является для этих узлов родительским.

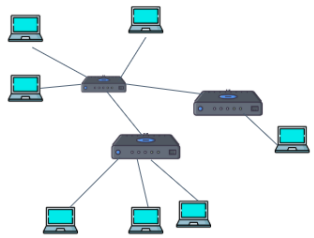
Смешанная — топология, преобладающая в крупных сетях с произвольными связями между компьютерами. В таких сетях можно выделить отдельные произвольно связанные фрагменты (подсети), имеющие типовую топологию. Поэтому их называют сетями со смешанной топологией.



Практическое задание

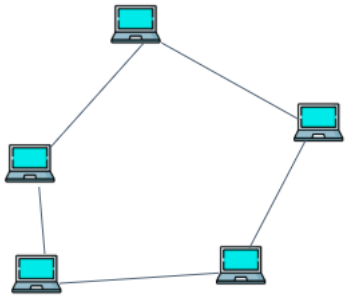
Презентация для выполнения практического задания:
https://docs.google.com/presentation/d/1Kb27UQkxtJwDW9g5rXNLzssQSDQ-Pf1ryRD4nhRT-KM/edit#slide=id.g354fdad279_0_1.

Можно скачать презентацию в формате PPTX (Урок 1.pptx) и открыть в PowerPoint или скопировать версию в Google Docs и работать с ней. Необходимо определить и записать правильные ответы.



Топология А:

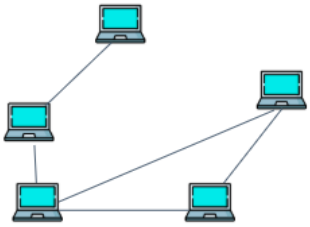
- смешанная;
- звезда;
- дерево.



Топология Б:

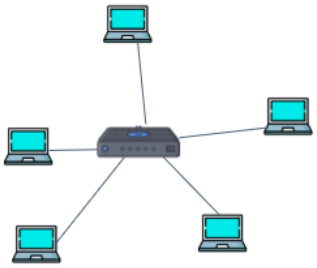
- шина;
- линейная;
- кольцо.

Топология В:



- линейная;
- смешанная;
- MESH.

Топология Г:



- шина;
- звезда;
- кольцо.

Отметьте правильные ответы (например жирным шрифтом) в своей копии презентации. Неправильные варианты можно убрать или зачеркнуть. Сдать выполненное задание можно в формате PPTX или PDF. В нем должны быть четыре топологии с указанием верного названия каждой из них.

Дополнительные материалы

1. https://proglib.io/p/graph-theory/.
2. Unucast, Multicast and Anycast.

3. https://ru.wikipedia.org/wiki/Ячеистая_топология.

Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. https://en.wikipedia.org/wiki/Token_ring.
2. https://mykonspekts.ru/1-17035.html.
3. http://survincity.ru/2015/01/radiodisciplina-pravila-radioobmena/.
4. https://habr.com/ru/company/ivideon/blog/230117/.
5. http://www.gps-profi.ru/what-is-gps.php.
6. http://www.avprog.narod.ru/phone/rdta.html.
7. http://bookre.org/reader?file=652828&pg=29.
8. https://poznayka.org/s68119t1.html.
9. https://proglib.io/p/graph-theory/.
10. https://ru.wikipedia.org/wiki/Глоссарий_теории_графов.
11. https://ru.wikipedia.org/wiki/Гиперграф.
12. https://ru.wikipedia.org/wiki/Мультиграф.
13. https://ru.wikipedia.org/wiki/Цикл_(теория_графов).