Network: Часть 2. Введение в сети

Published: 26.08.2026 15:10 | Updated: 26.08.2026 15:10

Что такое компьютерные сети и зачем они нужны

Компьютерная сеть — это система, в которой несколько устройств обмениваются данными через каналы связи по определённым правилам, называемым протоколами. В основе любой сети лежат три ключевых элемента: **узлы (hosts)** — компьютеры, серверы, смартфоны и другие устройства, **среда передачи** — кабель, оптоволокно, радиоканал (Wi-Fi, сотовая связь) и **протоколы** — правила, по которым устройства «общаются» друг с другом.

Зачем нужны сети

Сети пронизывают все сферы современной жизни. Можно выделить несколько ключевых применений:

Доступ к информации. Сети открывают доступ к веб-сайтам, базам данных, облачным сервисам. Когда вы открываете сайт в браузере, ваш компьютер обращается к удалённому серверу, и тот возвращает запрошенную страницу.
Общение. Электронная почта, мессенджеры, видеозвонки — всё это обмен пакетами данных между клиентом и сервером (или напрямую между устройствами). Сообщение в Telegram или видеовызов в Zoom — это тысячи пакетов, проходящих через сеть за доли секунды.
Электронная коммерция. Онлайн-магазины, банковские системы, платёжные шлюзы — все они используют сеть для передачи чувствительных данных. Именно поэтому в таких системах критически важна безопасность: шифрование, аутентификация, защита от перехвата.
Развлечения. Стриминговые сервисы (YouTube, Netflix), онлайн-игры — все они предъявляют высокие требования к пропускной способности и низкой задержке. Для просмотра видео в 4K нужна стабильная высокая скорость, для онлайн-игр — минимальная задержка, чтобы действия игрока отображались на экране без заметного запаздывания.
Интернет вещей (IoT). Умные устройства, датчики, бытовая техника — всё это

постоянно обменивается данными через сеть. По оценкам, к середине 2020-х годов количество подключённых IoT-устройств исчисляется десятками миллиардов.

Типы компьютерных сетей

Сети различаются по масштабу, назначению и технологиям.

Широкополосный доступ (access networks) — сети, через которые пользователь подключается к интернету: домашний проводной интернет, мобильный интернет.
Беспроводные и мобильные сети — Wi-Fi, сотовые сети 4G, 5G. Современные беспроводные технологии, такие как Wi-Fi 6/6E и 5G, обеспечивают пропускную способность и задержки, которые ещё несколько лет назад были доступны только проводным соединениям. Устройства подключаются без проводов, что даёт мобильность и гибкость.
CDN (сети доставки контента) — используются для ускорения доставки данных. Контент (видео с YouTube, статические файлы сайтов) кэшируется на серверах, расположенных географически ближе к пользователю. Это снижает задержку и разгружает магистральные каналы.
Корпоративные сети — сети внутри компаний и организаций: офисные сети, внутренние сервисы, VPN для удалённых сотрудников. Современные корпоративные сети всё чаще строятся как гибридные — сочетание собственной инфраструктуры и облачных сервисов.
Транзитные сети — сети операторов связи, которые соединяют другие сети между собой. Это «магистрали» интернета, по которым проходит основной трафик между разными регионами и континентами.

Масштабы сетей

Сети классифицируются по размеру и охвату:

Тип	Расшифровка	Пример
PAN	Personal Area Network	Bluetooth между смартфоном и наушниками
LAN	Local Area Network	Домашняя или офисная сеть
MAN	Metropolitan Area Network	Сеть в пределах города
WAN	Wide Area Network	Глобальная сеть, интернет

Интернет — это **глобальная сеть (WAN)** , состоящая из множества объединённых сетей разных типов и масштабов.

Примеры сетей

Интернет — самая большая сеть, соединяющая миллионы отдельных сетей. Он работает на основе стека IP-протоколов и не имеет единого владельца или центрального управления.
Мобильные сети — используют базовые станции (вышки сотовой связи) и обеспечивают связь через радиоканал. Современные сети 5G обеспечивают не только высокую скорость, но и сверхнизкую задержку, что открывает возможности для промышленной автоматизации, удалённой хирургии и автономного транспорта.
Wi-Fi — локальная беспроводная сеть, работающая в пределах помещения или небольшой территории. Технологии Wi-Fi 6 и Wi-Fi 7 значительно повысили пропускную способность и эффективность работы в условиях плотной застройки (много устройств в одном помещении).

Сетевые протоколы

Что такое протокол

Протокол — это набор правил, определяющих формат сообщений, порядок обмена и действия при отправке и получении данных. Протокол — это «язык общения» между устройствами. У протокола есть структура сообщений, порядок обмена, механизмы обработки ошибок и (в некоторых случаях) состояния.
Важно понимать: протокол работает **между одноимёнными уровнями разных устройств**. TCP на вашем компьютере «общается» с TCP на сервере, хотя физически данные проходят через множество промежуточных узлов.

Зачем нужны протоколы

Без протоколов устройства не смогут понять друг друга — данные останутся бессмысленным набором битов. Протоколы определяют, как интерпретировать эти биты, как обнаруживать ошибки, как запрашивать повторную отправку потерянных данных.

Разделение протоколов на уровни

Сложные системы делят на уровни (layering). Каждый уровень решает свою задачу и взаимодействует только с соседними уровнями через строго определённые интерфейсы. Пример стека: приложение (HTTP) → транспорт (TCP) → сеть (IP) → канальный уровень → физический уровень.

Соединения и надёжность

Одни протоколы устанавливают соединение и гарантируют доставку (например, TCP). Другие работают без установления соединения и не гарантируют доставку (UDP). Выбор между ними — это компромисс между надёжностью и скоростью/эффективностью.

Модели сетей

OSI — теоретическая модель

OSI (Open Systems Interconnection) — эталонная модель, разработанная Международной организацией по стандартизации (ISO). Она содержит **7 уровней**:

1. **Физический** — передача битов по физической среде (кабель, радиоволны, оптика).
2. **Канальный** — управление доступом к среде, адресация в пределах локальной сети (MAC-адреса).
3. **Сетевой** — маршрутизация пакетов между сетями (IP-адреса).
4. **Транспортный** — надёжная доставка данных между процессами (TCP/UDP, порты).
5. **Сеансовый** — управление диалогом между приложениями.
6. **Уровень представления** — преобразование данных (кодирование, шифрование, сжатие).
7. **Прикладной** — взаимодействие с пользовательскими приложениями (HTTP, FTP, SMTP).

OSI — это детальный «архитектурный чертёж» сети. Однако на практике она используется преимущественно как теоретическая и учебная модель, поскольку её полная реализация слишком сложна.

TCP/IP — практическая модель интернета

TCP/IP — это модель, которая реально используется в интернете. Она содержит **4 уровня**:

- **Прикладной** (HTTP, DNS, FTP, SMTP) — объединяет функции уровней 5–7 модели OSI.
- **Транспортный** (TCP, UDP).
- **Интернет** (IP, ICMP) — соответствует сетевому уровню OSI.
- **Сетевой доступ** (Ethernet, Wi-Fi) — объединяет физический и канальный уровни OSI.

TCP/IP — это «реально построенный дом», в отличие от OSI, которая осталась «подробным чертежом». Именно TCP/IP стек используется в реальных системах и устройствах по всему миру.

Службы (Services) vs Протоколы (Protocols)

Это один из самых важных концептов, который необходимо чётко различать.

- **Протокол (Protocol)** — это набор правил, определяющий, как два одноимённых уровня на разных устройствах взаимодействуют друг с другом. Это конкретная реализация: формат сообщений, порядок обмена, обработка ошибок.
- **Служба (Service)** — это абстракция, определяющая, что данный уровень предоставляет уровню, расположенному выше в том же устройстве. Это интерфейс, через который вышележащий уровень может пользоваться функциональностью нижележащего, не вникая в детали реализации.

Простая формула для запоминания:

> **Протокол = реализация (как работает двигатель)**
> **Служба = интерфейс/абстракция (как ты нажимаешь педаль газа)**

Ключевое следствие: уровень может полностью изменить свой внутренний протокол (например, перейти с одного алгоритма маршрутизации на другой), но если он сохранит ту же службу (интерфейс) для вышележащего уровня, то вышележащий уровень даже не заметит изменений. Это и есть основа модульности и независимости уровней.

Стандартизация

Чтобы сети разных производителей работали вместе, нужны открытые стандарты. Ключевые организации по стандартизации:

- **IEEE** (Institute of Electrical and Electronics Engineers) — разрабатывает стандарты для физического и канального уровней (Ethernet, Wi-Fi).
- **IETF** (Internet Engineering Task Force) — отвечает за протоколы интернета. Все спецификации публикуются в виде **RFC** (Request for Comments) документов, доступных каждому.
- **ISO** (International Organization for Standardization) — разработала модель OSI и множество других стандартов.

Открытость стандартов — ключевая причина, почему интернет смог вырасти из небольшой исследовательской сети в глобальную инфраструктуру. Любой производитель может реализовать протокол по спецификации RFC, и его устройства будут совместимы с устройствами других вендоров.

Социальные аспекты сетей

Отмечу, что сеть — это не только технология, но и социальная система. Ключевые вопросы:

Приватность — кто имеет доступ к вашим данным, как они собираются и используются.
Безопасность — защита от взломов, кражи данных, кибератак. С ростом числа подключённых устройств (IoT) поверхность атак расширяется, и безопасность становится ещё более критичной.
Цензура и нейтральность сети — вопрос о том, должны ли провайдеры иметь право ограничивать или приоритезировать трафик. Нейтральность сети означает, что все пакеты обрабатываются одинаково, независимо от их источника, назначения или содержимого.

Единицы измерения

В сетях используются:

- **Бит (bit)** — минимальная единица информации.
- **Байт (byte)** — 8 бит.

- Скорости передачи: **Kbps** (килобит в секунду), **Mbps** (мегабит в секунду), **Gbps** (гигабит в секунду).

Важно не путать мегабиты (Мбит/с) с мегабайтами (Мбайт/с) — 100 Mbps = 12,5 Мбайт/с.

Архитектура интернета как «сеть сетей»

Интернет — это не одна монолитная сеть, а **множество независимых сетей, соединённых между собой через маршрутизаторы**. Эти сети могут быть домашними LAN, корпоративными сетями, сетями интернет-провайдеров (ISP). Связь между сетями обеспечивается через:

- **Маршрутизаторы (routers)** — устройства, которые получают пакет, анализируют IP-адрес назначения и принимают решение, куда отправить пакет дальше.
- **Магистральные сети (backbone)** — высокоскоростные каналы, соединяющие крупные узлы.
- **Точки обмена трафиком (IXP)** — места, где разные провайдеры соединяют свои сети для обмена трафиком.

Структура интернета иерархична (упрощённо):

```
Конечные системы (хосты)
    ↓
Сети доступа
    ↓
Провайдеры (ISP)
    ↓
Региональные сети
    ↓
Магистральные сети
```

Пакетная передача (packet switching)

Пакетная передача — основной принцип работы интернета. Данные передаются не как единый непрерывный поток, а как **отдельные независимые пакеты**.

Как это работает

1. Сообщение разбивается на небольшие пакеты.
2. Каждый пакет передаётся отдельно.
3. Пакеты могут идти разными маршрутами через сеть.
4. На приёмной стороне пакеты собираются в исходное сообщение.

Свойства пакетной передачи

- Нет фиксированного выделенного канала — ресурсы используются эффективно.
- Возможны задержки и потери пакетов.
- Пакеты могут приходить не по порядку.
- Сеть может динамически адаптироваться к нагрузке и отказам.

Circuit switching vs Packet switching

Коммутация каналов (circuit switching) — устанавливается фиксированный канал на всё время сеанса связи, ресурсы резервируются. Пример: традиционные телефонные сети. Минус — неэффективно при простоях, так как канал занят даже когда данные не передаются.
Пакетная коммутация (packet switching) — канал не резервируется, пакеты динамически используют доступные ресурсы сети. Именно этот подход используется в интернете. Плюсы: высокая эффективность, гибкость, масштабируемость. Именно благодаря эффективности и гибкости интернет выбрал пакетную коммутацию.

Проблемы пакетной передачи и их решение

Пакетная сеть должна решать несколько фундаментальных проблем:

1. **Потери пакетов** — пакеты могут теряться в сети из-за переполнения буферов или ошибок передачи.
2. **Дублирование** — возможна повторная доставка одного и того же пакета.
3. **Перестановка порядка** — пакеты могут приходить не в том порядке, в котором были отправлены.
4. **Задержки (latency)** — время доставки может сильно варьироваться.

Эти проблемы не решаются на уровне IP (сетевой уровень). Вместо этого они решаются на более высоких уровнях — например, TCP на транспортном уровне обеспечивает надёжность, повторную передачу потерянных пакетов и восстановление порядка.

Задержки в сети

Когда данные проходят через сеть, возникает задержка, которая складывается из нескольких компонентов:

Processing delay — время обработки пакета узлом (маршрутизатором или хостом): проверка заголовка, поиск в таблице маршрутизации, принятие решения о пересылке.
Queuing delay — время ожидания пакета в очереди перед отправкой. Если очередь переполнена, пакеты отбрасываются — это одна из основных причин потерь.
Transmission delay — время, необходимое для «заливки» всех битов пакета в канал. Зависит от размера пакета и пропускной способности канала.
Propagation delay — время распространения сигнала по физической среде. Зависит от расстояния и скорости распространения сигнала (для оптоволокна — около 2/3 скорости света в вакууме).

В сумме эти четыре компонента определяют **общее время доставки пакета** от отправителя к получателю.

Пропускная способность и потери

Пропускная способность (bandwidth) — теоретическая максимальная скорость, которую канал может обеспечить. Например, канал 100 Mbps.
Фактическая скорость (throughput) — реальная скорость передачи данных, которая зависит от загруженности сети, потерь пакетов, задержек и особенностей протоколов. Throughput почти всегда ниже bandwidth.
Потери пакетов (packet loss) — пакеты могут теряться из-за переполнения буферов маршрутизаторов, ошибок передачи или перегрузки сети. Потери — нормальное явление в сетях, и надёжные протоколы (TCP) умеют с ними справляться.

Очереди и буферизация

Каждый маршрутизатор имеет буфер (очередь), где пакеты хранятся перед отправкой. Буферы нужны для сглаживания колебаний трафика: если пакеты приходят быстрее, чем маршрутизатор может их отправить, они встают в очередь.
Если очередь переполнена, новые пакеты отбрасываются. Это одна из основных причин **packet loss** и роста **задержки** (когда очередь длинная, пакеты ждут дольше).

Сетевые топологии

Сети могут быть организованы по-разному:

- **Звезда** — все устройства подключены к одному центральному узлу.
- **Кольцо** — устройства соединены последовательно в замкнутое кольцо.
- **Mesh (сетка)** — устройства соединены множеством связей, обеспечивая избыточность.
- **Дерево** — иерархическая структура.

В реальности интернет — это сложная комбинация mesh-топологии на магистральном уровне и иерархических структур на уровнях доступа.

Адресация в сети

Для доставки данных нужны адреса, и на разных уровнях используются разные типы адресов:

- **MAC-адрес** — физический адрес сетевого интерфейса (канальный уровень). Используется для доставки в пределах одной локальной сети.
- **IP-адрес** — логический адрес устройства (сетевой уровень). Используется для глобальной маршрутизации между сетями.

Это два разных пространства адресов, и они решают разные задачи. IP-адрес может меняться при перемещении устройства между сетями, MAC-адрес остаётся неизменным (привязан к оборудованию).

Фрагментация пакетов

Каждый тип сети имеет ограничение на максимальный размер передаваемого блока данных — **MTU** (Maximum Transmission Unit). Если пакет, полученный от вышележащего уровня, превышает MTU текущего канала, он разбивается на несколько более мелких частей (фрагментов). На приёмной стороне фрагменты собираются обратно в исходный пакет.
Фрагментация — это компромисс: она позволяет передавать большие пакеты через сети с маленьким MTU, но создаёт дополнительную нагрузку на маршрутизаторы и увеличивает риск потерь (если потерян один фрагмент, весь пакет считается потерянным).

Мультиплексирование и демультиплексирование

Мультиплексирование (multiplexing) — несколько потоков данных используют один и тот же канал связи. Интернет использует **статистическое мультиплексирование**: ресурсы канала распределяются динамически между пакетами разных потоков по мере их поступления.
Демультиплексирование (demultiplexing) — на стороне приёма система определяет, какому процессу или приложению передать полученные данные. Например, TCP использует **порты** для этой цели: входящий пакет содержит номер порта, и операционная система направляет данные соответствующему приложению.

Роль портов и сокетов

Порт — это числовой идентификатор конкретного приложения или сервиса на устройстве. Стандартные порты:

- 80 — HTTP
- 443 — HTTPS
- 22 — SSH
- 53 — DNS

Комбинация **IP-адрес + порт + протокол** однозначно идентифицирует конкретный сетевой сервис на конкретном устройстве.
Сокет (socket) — это программная абстракция, точка соединения между приложением и сетью. Сокет определяется как IP-адрес + порт + протокол. Через сокет приложение отправляет и получает данные.

Инкапсуляция как универсальный механизм

Инкапсуляция (encapsulation) — ключевой механизм передачи данных в уровневой архитектуре. На каждом уровне данные «оборачиваются»: вышележащий уровень передаёт данные нижележащему, а тот добавляет свой заголовок (и иногда — концевик).
Процесс выглядит так:

```
[HTTP данные]
   ↓
[TCP заголовок + данные]
   ↓
[IP заголовок + TCP + данные]
   ↓
[Ethernet заголовок + IP + TCP + данные] — кадр
   ↓
битовый поток в среде передачи
```

На стороне получателя процесс обратный — **декапсуляция**: каждый уровень снимает свой заголовок и передаёт данные вышележащему уровню.
Каждый уровень работает независимо, не зная деталей других уровней. Это фундамент масштабируемости и модульности сетевой архитектуры.

Роль маршрутизаторов

Маршрутизатор работает на **сетевом уровне (IP)** . Он не интересуется содержимым данных (не важно, HTTP это, видео или что-то ещё) — он принимает решение исключительно на основе IP-адреса назначения.

Как маршрутизатор обрабатывает пакет

1. Получает пакет из одного интерфейса.
2. Считывает IP-адрес назначения из заголовка.
3. Смотрит **таблицу маршрутизации** (содержит информацию о доступных сетях и путях к ним).
4. Выбирает следующий узел (next hop) для этого пакета.
5. Отправляет пакет через соответствующий интерфейс.

Концепция «best effort»

Интернет работает по принципу **best effort** — «с максимальным усилием». Это означает:

- Сеть делает всё возможное для доставки каждого пакета.
- Но **не гарантирует** доставку.
- Не гарантирует порядок прибытия пакетов.
- Не гарантирует время доставки.

Надёжность, если она нужна, обеспечивается на более высоких уровнях — например, TCP. Такой подход позволяет сети оставаться простой, быстрой и масштабируемой. Сложность выносится на конечные узлы (хосты), а не встраивается в ядро сети.

End-to-End принцип

End-to-End принцип — один из фундаментальных архитектурных принципов интернета. Он гласит: **сложные функции должны реализовываться на концах сети (на хостах), а не внутри неё**.

Что это значит на практике

- Сеть (IP-уровень) — простая и быстрая. Она только доставляет пакеты.
- Вся сложная логика (надёжность, контроль потока, шифрование, аутентификация) — на хостах.

Например: TCP реализует надёжную доставку поверх ненадёжного IP. Сеть не занимается исправлением ошибок — это делают конечные устройства.

Этот принцип позволил интернету эволюционировать: можно добавлять новые протоколы и сервисы на хостах, не меняя инфраструктуру сети. Однако современные требования (видеоконференции, облачные вычисления, IoT, промышленная автоматизация) создают давление на эту модель, и в некоторых случаях сеть всё же должна предоставлять определённые гарантии качества (QoS).

Ключевые принципы архитектуры интернета

Если сказать коротко, интернет построен на сочетании трёх фундаментальных идей:

1. **Пакетная передача (packet switching)** — данные передаются отдельными пакетами, а не непрерывным потоком. Это обеспечивает эффективное использование ресурсов и устойчивость к отказам.
2. **Уровневая архитектура (layering)** — сложная задача разбита на независимые уровни, каждый решает свою задачу и взаимодействует с соседними через строго определённые интерфейсы.
3. **End-to-End принцип** — сложность вынесена на конечные узлы, сеть остаётся простой и быстрой.

Эти три идеи вместе дают **масштабируемость**, **устойчивость** и **гибкость** — именно те качества, которые позволили интернету вырасти из небольшой исследовательской сети в глобальную инфраструктуру, объединяющую миллиарды устройств.

Открытая архитектура и независимость уровней

Интернет построен как **открытая, расширяемая система**. Это означает:

- Можно добавлять новые протоколы на любом уровне, не меняя остальные.
- Можно заменять технологии на нижних уровнях (например, переходить с медного кабеля на оптоволокно или с Ethernet на Wi-Fi), и верхние уровни при этом продолжают работать.
- Изменения на одном уровне не требуют изменений на других.

Именно открытость стандартов и независимость уровней обеспечили совместимость систем разных производителей и позволили интернету развиваться без «сломанной совместимости».

Баланс между простотой и функциональностью

Архитектура интернета — это сознательный компромисс:

- **Сеть (IP)** — простая, быстрая, минималистичная.
- **Сложность** — на концах (TCP, приложения, шифрование, аутентификация).

Этот баланс даёт максимальную гибкость: сеть не нужно перестраивать для поддержки новых приложений. Новые сервисы просто реализуются на хостах, используя существующую сетевую инфраструктуру. Именно поэтому интернет смог поддержать и веб, и видео-стриминг, и облачные вычисления, и интернет вещей — без фундаментальных изменений в ядре сети.

Итог

Статья закладывает фундамент для всего дальнейшего изучения компьютерных сетей. Ключевые концепции, которые формируют «мышление сетевого инженера»:

- **Интернет = сеть сетей**, соединённых маршрутизаторами.
- Данные передаются **пакетами**, а не непрерывным потоком.
- Сеть работает по принципу **best effort** — без гарантий.
- Сложность вынесена на край сети (**end-to-end принцип**).
- Уровни изолируют ответственность и упрощают систему (**layering**).
- Протоколы работают **между одноимёнными уровнями** разных устройств.
- **Инкапсуляция** — ключевой механизм передачи данных.
- Маршрутизаторы принимают решения **на основе IP-адресов**.
- Задержки складываются из **processing, queuing, transmission и propagation**.
- **Порты** позволяют одному устройству обслуживать множество приложений.

Эти принципы — не просто теоретические знания. Они лежат в основе анализа трафика (Wireshark), диагностики сетевых проблем, настройки серверов и оптимизации производительности. Понимание того, как пакет проходит от клиента до сервера через несколько сетей, почему возможны потери и задержки, почему интернет не гарантирует доставку — это основа для осознанной работы с любыми сетевыми технологиями.