Модель TCP/IP — это стек протоколов, которые задают правила передачи данных по Сети. Так как бал здесь правят протоколы TCP и IP, в честь них и назвали всю модель.

**TCP (Transmission Control Protocol)** отвечает за обмен данными. Он управляет их отправкой и следит за тем, чтобы они дошли до получателя в целости. У TCP есть свои гарантии, что всё пройдёт успешно, — о них чуть позже.

**IP (Internet Protocol)** отвечает за адресацию. Его задача — связывать друг с другом устройства и нарезать данные на пакеты для удобной отправки. Чтобы протокол мог быстро найти дорогу от одного компьютера к другому, придумали IP-адреса — уникальные идентификаторы, которые есть у каждого устройства в Сети.

Эти два протокола работают в связке: IP строит маршрут, а TCP контролирует, чтобы всё передавалось правильно. Условно их можно сравнить с тандемом Шерлока и Ватсона в бессмертной классике Дойля: первый решает проблему, а второй следит, чтобы тот не наделал глупостей и случайно не упал в Рейхенбахский водопад.

**Минутка истории.** Оба протокола, и TCP, и IP, на самом деле стары как мир. Их придумала ещё в семидесятых группа разработчиков под началом «отца интернета» Винта Сёрфа. То есть все наши смартфоны и умные часы общаются по правилам, которые заложили ещё во времена расцвета диско и космической миссии «Союз — Аполлон».

Но протоколами TCP и IP модель не ограничивается — например, есть ещё HTTP, FTP, UDP и сотни других. Все они заточены под определённые задачи. Так, HTTP помогает браузеру формировать запросы к серверу, FTP — скачивать файлы, а UDP — быстро передавать видео, музыку и игры, но с возможными потерями по пути.

**Как работает TCP/IP**

Настало время посмотреть, как модель TCP/IP работает в жизни. Допустим, вы заходите в интернет, чтобы почитать [статью о том, как нейросети рисуют котиков](https://skillbox.ru/media/code/govorim-o-generativnosostyazatelnykh-neyrosetyakh/?utm_source=media&utm_medium=link&utm_campaign=all_all_media_links_links_articles_all_all_skillbox). Запускаете браузер, открываете сайт Skillbox Media, нажимаете на заголовок статьи. А дальше начинается магия.

Как только вы кликнули на ссылку, браузер отправляет запрос на сервер, где лежит страница. Для этого он создаёт HTTP-запрос, в котором описывает всё, что сервер должен ему передать: «Я слышал, у тебя есть статья про генеративные нейронки. Дай, пожалуйста». Сервер отвечает: «Нет проблем» — и отправляет браузеру HTTP-ответ с нужными данными. И так они будут общаться до тех пор, пока вы не закроете сайт.

**Из каких уровней состоит TCP/IP**

Пора окунуться в детали и узнать, как модель TCP/IP устроена изнутри. Глобально она делится на четыре уровня:

* канальный уровень — отвечает за взаимодействие по сетевому оборудованию, например по Ethernet-кабелю или Wi-Fi;
* межсетевой уровень — помогает отдельным сетям общаться друг с другом;
* транспортный уровень — отвечает за передачу данных между устройствами, например, по протоколам TCP и UDP;
* прикладной уровень — помогает приложениям общаться друг с другом с помощью интерфейсов или API.

На каждом уровне есть свои протоколы, которые обеспечивают надёжность передачи данных между компьютерами в Сети. Всего модель TCP/IP поддерживает сотни разных протоколов.

**Канальный уровень**

**Для чего нужен:** устанавливать физическое соединение между устройствами в локальной сети с помощью радиоволн и проводов.

**Примеры протоколов:** [Ethernet](https://ru.wikipedia.org/wiki/Ethernet), [Wi-Fi](https://ru.wikipedia.org/wiki/Wi-Fi" \t "_blank), [Bluetooth](https://ru.wikipedia.org/wiki/Bluetooth).

**Как работает:** данные делятся на небольшие кусочки (фреймы) и передаются между устройствами. Каждый фрейм содержит часть передаваемой информации и служебные данные.

Чтобы понять, куда отправлять фреймы, используют адресацию канального уровня — MAC-адреса. Это уникальные физические адреса устройств — по ним протоколы канального уровня определяют отправителей и получателей.

Ещё одна важная задача канального уровня — проверять, что данные передаются безошибочно. Для этого протоколы используют свои средства проверки:

* Если возникла ошибка, устройство отправляет фрейм обратно, а второе устройство передаёт его ещё раз.
* Если всё прошло удачно, то фрейм передаётся на следующий уровень для обработки.

**Межсетевой уровень**

**Для чего нужен:** строить маршруты между устройствами по всему интернету — этот процесс называется маршрутизацией.

**Примеры протоколов:** [IP](https://ru.wikipedia.org/wiki/IP), [ICMP](https://ru.wikipedia.org/wiki/ICMP), [ARP](https://ru.wikipedia.org/wiki/ARP).

**Как работает:** IP-протокол вычисляет местонахождение устройств по их IP-адресам, а также строит до них кратчайшие пути и делит данные на пакеты (или, как говорят на юге России, — кульки :)).

Чтобы определить, где находится получатель и как построить путь к нему, IP обращается к системе DNS — она знает IP-адреса всех устройств в интернете.

Когда адрес получен, передаваемый файл разбивается на небольшие части — пакеты. Они содержат фрагменты данных и служебную информацию, например IP-адреса отправителя и получателя.

После этого начинается передача пакетов по маршрутизаторам и коммутаторам. Но процессом отправки занимается уже следующий уровень — транспортный.

**Транспортный уровень**

**Для чего нужен:** передавать данные по маршруту, построенному на предыдущем уровне.

**Примеры протоколов:** [TCP](https://ru.wikipedia.org/wiki/Transmission_Control_Protocol), [UDP](https://ru.wikipedia.org/wiki/UDP).

**Как работает:** устанавливает надёжное соединение между устройствами, а затем следит за передачей данных по нему и исправляет ошибки.

Главных протокола здесь два:

* TCP (Transmission Control Protocol) — гарантирует передачу всех данных без потерь. Полезен при отправке текстовых файлов.
* UDP (User Datagram Protocol) — не гарантирует передачу данных без потерь, но обеспечивает хорошую скорость. Полезен при просмотре видео или прослушивании музыки в интернете.

TCP решает всё медленно, но надёжно. UDP — быстро, но не факт, что качественно

И у нас остаётся последний уровень — прикладной.

**Прикладной уровень**

**Для чего нужен:** настраивать связи между приложениями — например, между браузером и серверным софтом.

**Примеры протоколов:** [HTTP](https://skillbox.ru/media/code/chto-takoe-http-i-zachem-on-nuzhen/?utm_source=media&utm_medium=link&utm_campaign=all_all_media_links_links_articles_all_all_skillbox), [FTP](https://ru.wikipedia.org/wiki/FTP), [SMTP](https://ru.wikipedia.org/wiki/SMTP).

**Как работает:** использует различные протоколы и сервисы, которые помогают приложениям обмениваться данными по интернету.

На прикладном уровне хранятся протоколы для всего, что нужно человеку: отправки имейлов, веб-браузинга, передачи файлов и удалённого доступа. Вот некоторые из них:

* HTTP (Hypertext Transfer Protocol) — самый популярный протокол для передачи данных по интернету.
* FTP (File Transfer Protocol) — ещё один известный протокол, заточенный под передачу файлов.
* SMTP (Simple Mail Transfer Protocol) — протокол для отправки электронных писем.

Этот уровень нужен, чтобы упростить пользователям передачу данных по интернету. Именно с его помощью программисты и обычные пользователи взаимодействуют с моделью TCP/IP.