**Иерархическая модель взаимодействия открытых систем (OSI)**

семиуровневая иерархическая модель, разработанная [Международным комитетом по стандартизации](http://your-hosting.ru/terms/i/iso/) [ISO](http://your-hosting.ru/terms/i/iso/) для определения спецификации и связи сетевых [протоколов](http://your-hosting.ru/terms/p/protocol/).

* (phisical layer) - первый (физический) уровень - определяющий связь на уровне аппаратуры.
* (data link layer) - уровень канала передачи данных - отвечает за прием и передачу [пакетов](http://your-hosting.ru/terms/p/packet/), сервис на уровне дэйтаграмм, локальную адресацию и контроль ошибок.
* (network layer) - сетевой уровень - отвечает за адресацию и маршрутизацию при межсетевом обмене.
* (transport layer) - транспортный уровень - обеспечивает доставку между конечными точками.
* (session layer) - сеансовый уровень - отвечает за установку, поддержку и разрыв соединения.
* (representation layer) - представительский уровень - определяет способы шифрования, кодирования и, иногда, сжатия данных.
* (application layer) - прикладной уровень - определяет способы передачи информации между приложениями.

**Протоколы передачи данных**

* IP
* UDP
* FTP
* DNS
* HTTP
* NTP
* SSH





Название данного протокола - **Intrenet Protocol** - отражает его суть: он должен передавать пакеты между сетями. В каждой очередной сети, лежащей на пути перемещения пакета, протокол IP вызывает средства транспортировки, принятые в этой сети, чтобы с их помощью передать этот пакет на маршрутизатор, ведущий к следующей сети, или непосредственно на узел-получатель.

Протокол IP относится к протоколам без установления соединений. Перед IP не ставится задача надежной доставки сообщений от отправителя к получателю. Протокол IP обрабатывает каждый IP-пакет как независимую единицу, не имеющую связи ни с какими другими IP-пакетами. В протоколе IP нет механизмов, обычно применяемых для увеличения достоверности конечных данных: отсутствует квитирование - обмен подтверждениями между отправителем и получателем, нет процедуры упорядочивания, повторных передач или других подобных функций. Если во время продвижения пакета произошла какая-либо ошибка, то протокол IP по своей инициативе ничего не предпринимает для исправления этой ошибки. Например, если на промежуточном маршрутизаторе пакет был отброшен по причине истечения времени жизни или из-за ошибки в контрольной сумме, то модуль IP не пытается заново послать испорченный или потерянный пакет. Все вопросы обеспечения надежности доставки данных по составной сети в стеке TCP/IP решает протокол TCP, работающий непосредственно над протоколом IP. Именно TCP организует повторную передачу пакетов, когда в этом возникает необходимость. IP-пакет состоит из заголовка и поля данных.

Механизм **TCP** предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым (в отличие от UDP) целостность передаваемых данных и уведомление отправителя о результатах передачи.

**UDP** использует простую модель передачи, без явных «рукопожатий» для обеспечения надёжности, упорядочивания или целостности данных. Датаграммы могут прийти не по порядку, дублироваться или вовсе исчезнуть без следа, но гарантируется, что если они придут, то в целостном состоянии. UDP подразумевает, что проверка ошибок и исправление либо не нужны, либо должны исполняться в приложении. Чувствительные ко времени приложения часто используют UDP, так как предпочтительнее сбросить пакеты, чем ждать задержавшиеся пакеты, что может оказаться невозможным в системах реального времени. При необходимости исправления ошибок на сетевом уровне интерфейса приложение может задействовать TCP или SCTP, разработанные для этой цели.

Сокет — программный интерфейса для обеспечения обмена данными между процессами. Процессы при таком обмене могут исполняться как на одной ЭВМ, так и на различных ЭВМ, связанных между собой сетью. Сокет — абстрактный объект, представляющий конечную точку соединения.

Сокеты бывают клиентские и серверные. Клиентские можно сравнить с конечными аппаратами телефонной сети, а серверные — с коммутаторами. Клиентское приложение (например, браузер) использует только клиентские сокеты, а серверное (например, веб-сервер, которому браузер посылает запросы) — как клиентские, так и серверные сокеты. Для взаимодействия между машинами с помощью стека протоколов TCP/IP используются адреса и порты. Адрес представляет собой 32-битную структуру для протокола IPv4, 128-битную для IPv6. Номер порта — целое число в диапазоне от 0 до 65535 (для протокола TCP).

Эта пара определяет сокет («гнездо», соответствующее адресу и порту).