47 Протоколы транспортного уровня. Протоколы TCP и UDP. Принципы работы TCP.

Транспортный уровень:
http://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D1%8B%D0%B9_%D1%83%D1%80%D0%BE%D0%B2%D0%B5%D0%BD%D1%8C

Назначение транспортного уровня: http://datanets.ru/naznachenie-transportnogo-urovnia.html

Сетевая модель OSI:
http://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI

Основные функции TCP: http://kunegin.com/ref1/net_prot/tcpprot.htm

Транспортный уровень (англ. *Transport layer*) — 4-й уровень сетевой модели OSI, предназначен для доставки данных. При этом не важно, какие данные передаются, откуда и куда, то есть, он предоставляет сам механизм передачи. Блоки данных он разделяет на фрагменты, размер которых зависит от протокола, короткие объединяет в один, а длинные разбивает. Протоколы этого уровня предназначены для взаимодействия типа точка-точка. Пример: TCP, UDP.

В функции протоколов транспортного уровня TCP и UDP входит исполнение роли связующего звена между прилегающими к ним прикладным уровнем и уровнем межсетевого взаимодействия. От прикладного протокола транспортный уровень принимает задание на передачу данных с тем или иным качеством, а после выполнения рапортует об этом. Нижележащий уровень межсетевого взаимодействия протоколы TCP и UDP рассматривают как своего рода инструмент, не очень надежный, но способный перемещать пакет в свободном и рискованном путешествии по составной сети.

UDP

UDP (англ. *User Datagram Protocol* — протокол пользовательских датаграмм) — один из ключевых элементов Transmission Control Protocol/Internet Protocol, набора сетевых протоколов для Интернета. С UDP компьютерные приложения могут посылать сообщения (в данном случае называемые датаграммами) другим хостам по IP-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных.

UDP использует простую модель передачи, без неявных «рукопожатий» для обеспечения надёжности, упорядочивания или целостности данных. Таким образом, UDP предоставляет ненадёжный сервис, и датаграммы могут прийти не по порядку, дублироваться или вовсе исчезнуть без следа. 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.

[Дейтаграмма (англ. *datagram*), также **датаграмма** — блок информации, посланный как пакет сетевого уровня через передающую среду без предварительного установления соединения и создания виртуального канала. Датаграмма представляет собой единицу информации в протоколе (*protocol data unit*, PDU) для обмена информацией на сетевом (в случае , IP-датаграммы) и транспортном (в случае , UDP-датаграммы) уровнях эталонной модели OSI.**]**

Природа UDP как протокола без сохранения состояния также полезна для серверов, отвечающих на небольшие запросы от огромного числа клиентов, например DNS и потоковые мультимедийные приложения вроде IPTV, IP Телефония (VoIP - сокр. от Voice IP), протоколы туннелирования IP и многие онлайн-игры.

Основные особенности:

- *Ненадёжный* — когда сообщение посылается, неизвестно, достигнет ли оно своего назначения — оно может потеряться по пути. Нет таких понятий, как подтверждение, повторная передача, тайм-аут.

- *Неупорядоченность* — если два сообщения отправлены одному получателю, то порядок их достижения цели не может быть предугадан.

- *Легковесность* — никакого упорядочивания сообщений, никакого отслеживания соединений и т. д. Это небольшой транспортный уровень, разработанный на IP.

- *Датаграммы* — пакеты посылаются по отдельности и проверяются на целостность только если они прибыли. Пакеты имеют определенные границы, которые соблюдаются после получения, то есть операция чтения на сокете-получателе выдаст сообщение таким, каким оно было изначально послано.

- *Нет контроля перегрузок* — UDP сам по себе не избегает перегрузок. Для приложений с большой пропускной способностью возможно вызвать коллапс перегрузок, если только они не реализуют меры контроля на прикладном уровне.

TCP

TCP (англ. *Transmission Control Protocol*, протокол управления передачей) — один из основных протоколов передачи данных Интернета, предназначенный для управления передачей данных в сетях и подсетях TCP/IP.

Выполняет функции протокола транспортного уровня в стеке протоколов IP.

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.

В отличие от традиционной альтернативы — UDP, который может сразу же начать передачу пакетов, TCP устанавливает соединения, которые должны быть созданы перед передачей данных. TCP соединение можно разделить на 3 стадии:

- Установка соединения
- Передача данных
- Завершение соединения

Принцип работы:

Протокол TCP предоставляет транспортные услуги, отличающиеся от услуг UDP. Вместо ненадежной доставки датаграмм без установления соединений, он обеспечивает гарантированную доставку с установлением соединений в виде байтовых потоков.

Протокол TCP используется в тех случаях, когда требуется надежная доставка сообщений. Он освобождает прикладные процессы от необходимости использовать таймауты и повторные передачи для обеспечения надежности. Наиболее типичными прикладными процессами, использующими TCP, являются FTP (File Transfer Protocol - протокол передачи файлов) и TELNET. Кроме того, TCP используют система X-Window, rcp (remote copy - удаленное копирование) и другие "r-команды". Большие возможности TCP даются не бесплатно. Реализация TCP требует большой производительности процессора и большой пропускной способности сети. Внутренняя структура модуля TCP гораздо сложнее структуры модуля UDP.

Прикладные процессы взаимодействуют с модулем TCP через порты. Для отдельных приложений выделяются общеизвестные номера портов. Например, сервер TELNET использует порт номер 23. Клиент TELNET может получать услуги от сервера, если установит соединение с TCP-портом 23 на его машине.

Когда прикладной процесс начинает использовать TCP, то модуль TCP на машине клиента и модуль TCP на машине сервера начинают общаться. Эти два оконечных модуля TCP поддерживают информацию о состоянии соединения, называемого виртуальным каналом. Этот виртуальный канал потребляет ресурсы обоих оконечных модулей TCP. Канал является дуплексным; данные могут одновременно передаваться в обоих направлениях. Один прикладной процесс пишет данные в TCP-порт, они проходят по сети, и другой приклад ной процесс читает их из своего TCP-порта.

Протокол TCP разбивает поток байт на пакеты; он не сохраняет границ между записями. Например, если один прикладной процесс делает 5 записей в TCP-порт, то прикладной процесс на другом конце виртуального канала может выполнить 10 чтений для того, чтобы получить все данные. Но этот же процесс может получить все данные сразу, сделав только одну операцию чтения. Не существует зависимости между числом и размером записываемых сообщений с одной стороны и числом и размером считываемых сообщений с другой стороны.

Протокол TCP требует, чтобы все отправленные данные были подтверждены принявшей их стороной. Он использует таймауты и повторные передачи для обеспечения надежной доставки. Отправителю разрешается передавать некоторое количество данных, не дожидаясь подтверждения приема ранее отправленных данных. Таким образом, между отправленными и подтвержденными данными существует окно уже отправленных, но еще неподтвержденных данных. Количество байт, которые можно передавать без подтверждения, называется размером окна. Как правило, размер окна устанавливается в стартовых файлах сетевого программного обеспечения. Так как TCP-канал является дуплексным, то подтверждения для данных, идущих в одном направлении, могут передаваться вместе с данными, идущими в противоположном направлении. Приемники на обеих сторонах виртуального канала выполняют управление потоком передаваемых данных для того, чтобы не допускать переполнения буферов.

Основные особенности:

- *Надёжность* — TCP управляет подтверждением, повторной передачей и тайм-аутом сообщений. Производятся многочисленные попытки доставить сообщение. Если оно потеряется на пути, сервер вновь запросит потерянную часть. В TCP нет ни пропавших данных, ни (в случае многочисленных тайм-аутов) разорванных соединений.

- *Упорядоченность* — если два сообщения последовательно отправлены, первое сообщение достигнет приложения-получателя первым. Если участки данных прибывают в неверном порядке, TCP отправляет неупорядоченные данные в буфер до тех пор, пока все данные не могут быть упорядочены и переданы приложению.

- *Тяжеловесность* — TCP необходимо три пакета для установки сокет-соединения перед тем, как отправить данные. TCP следит за надёжностью и перегрузками.

- *Потоковость* — данные читаются как поток байтов, не передается никаких особых обозначений для границ сообщения или сегментов.

Примеры приложений, использующих TCP:

- Веб Браузеры
- E-mail
- Программы для передачи файлов