Модель и основные принципы Интернета

Как мы уже видим, взаимодействие всех известных приложений в Интернете может быть сведено к виду надежного двунаправленного потока байтов. Инициирование такого потока, его надежное поддержание требует, как мы увидели, взаимодействие большого числа компонентов. Несмотря на различие приложений, скорости взаимодействия, передаваемых данных, есть общие характерные механизмы, обеспечивающие такое взаимодействие. Например, приложения «не задумываются» о том по какому маршруту пойдут данные через Интернет. Практически всегда приложения полагаются на то, что посланные данные будут доставлены корректно, т.е. тому, кому они должны быть доставлены и в том виде, в каком были отправлены.

Создатели компьютерных сетей и Интернета с самого начала постарались разработать такую модель взаимодействия, чтобы приложения:

- никак не зависели от того, как реализована передача их данных;
- могли многократно использовать одни и те же средства передачи, а не создавать их каждый раз заново.

Эти свойства были достигнуты благодаря использованию нескольких принципов, которые мы рассмотрим. Один из них — это **принцип уровневости**, который мы сейчас рассмотрим на примере 4-х уровневой модели Интернета. (Здесь надо отметить, что принцип уровневости, это частный случай принципа структуризации, который люди принимают всегда, когда сталкиваются со сложными проблемами, построением сложных социальных, технических систем). Другой важный принцип – это **принцип инкапсуляции**. Еще один – это принцип **коммутации пакетов**. Все эти принципы мы обязательно рассмотрим, а сейчас сосредоточимся на принципе уровневости.

Этот принцип предполагает, что все операции по передаче данных распределяются между несколькими немногочисленными уровнями. Каждый уровень предоставляет непосредственно вышележащему и только ему доступ к операциям, или как еще говорят, сервисам, реализуемым на этом уровне. Этот доступ осуществляется с помощью строго определенного интерфейса между уровнями. Между каждой парой уровней этот интерфейс свой. Вышележащий уровень не должен зависеть и знать, как реализована та или иная операция на нижележащем уровне.

Рассмотрим, за что отвечает каждый из 4-х уровней этой Модели Интернета (везде далее просто Модели). Важно все время помнить, что основное предназначение всех уровней в этой Модели – обеспечить приложениям надежными средствами передачи данных.

Каждый уровень:

- имеет строго определенное предназначение
- обеспечивает определенный набор сервисов для вышележащего уровня,
- механизмы и сервисы вышележащего уровня никак не зависят от того как реализованы сервисы нижележащего уровня;
- между уровнями определен интерфейс, представляющий собой набор операций и структур данных.

Начнем наше рассмотрение с самого нижнего уровня – Канального (Link) уровня.

Интернет состоит из хостов, линий связи/каналов и маршрутизаторов. Данные передаются скачками (hops) по линиям (см. рис. Организация компьютерной сети)

Рисунок

Данные передают кадрами. Каждый кадр состоит из тела, где хранятся собственно передаваемые данные, и заголовка. В заголовке указано, куда и кому кадр должен быть доставлен, откуда кадр был отправлен и т.д. (на предыдущем рисунке поясним сказанное). Канальный уровень отвечает за надежную передачу кадров по каналам. Вы уже на практике сталкивались с Ethernet и WiFi линиям.

Следующий вышележащий уровень – Сетевой - этот уровень отвечает за доставку пакета в нужное место. Здесь фрагменты передаваемых данных называются пакеты. Пакет – это самодостаточная структура, содержащая собственно передаваемые данные (тело) и информацию о том, откуда пакет был отправлен, куда (рисуем пакет). Сетевой уровень передает пакет канальному уровню, сообщая по какому каналу его надо передать. Другими словами, Канальный уровень предоставляет сервис Сетевому уровню по передаче пакетов по каналам.

На другом конце канала находится Router (Маршрутизатор). Канальный уровень маршрутизатора принимает кадр, проверяет правильность его доставки, преобразует кадр в пакет и передает его Сетевому уровню маршрутизатора.

Сетевой уровень маршрутизатора отвечает за формирование маршрута, определяя направление следующего скачка (hop), глядя на адрес назначения. И так hop-by-hop пакет должен быть доставлен к месту назначения.

Маршрутизатор определяет, какому следующему маршрутизатору пакет должен быть передан.

Тем самым, определяя по какому каналу пакет должен быть передан. Канальный уровень формирует кадр и переедет его. Таким образом, Сетевому уровню не надо «думать» как передать пакет по каналу, заботиться о корректности передачи и т.д. Передача данных по каналу WiFi или Ethernet происходит по-разному, но для Сетевого уровня эти особенности не видны. Таким образом, Сетевой уровень может работать, не обращая внимания на особенности, различии каналов. Это достигается за счет строго определенного и зафиксированного в стандартах интерфейса (API) между Сетевым и Канальным уровнями.

Сетевой уровень – это сердцевина Интернета, его основным протоколом является Internet Protocol (IP), который является «клеем», соединяющим между собой разные части Интернета. Когда мы посылаем пакет в Интернет, мы обязательно используем IP. Позже мы подробно рассмотрим работу этого протокола. А пока мы рассмотрим некоторые его свойства.

Прежде всего, IP доставляет пакет по возможности (best efforts), т.е. доставка не гарантирована. Пакет может быть потерян, разрушен, нарушен порядок, т.е. никто в Интернете не гарантирует, что ранее посланный пакет будет получен ранее позже посланного.

Что же делать приложению, если ему нужен надежный сервис гарантирующий доставку, порядок пакетов в потоке? Для этого ему нужно воспользоваться сервисом Транспортного уровня. На этом уровне работает протокол TCP (Transport Control Protocol). Этот протокол обеспечит доставку всех пакетов потока в нужное место, в определенном порядке. Если какой-то или какие-то пакеты будут потеряны, то TCP потребует от IP повторить передачу потерянных пакетов столько раз, сколько потребуется чтобы потерянный пакет был получен. Если при передаче, порядок пакетов был нарушен, то TCP восстановит его.

Приложению не требуется создавать у себя все те механизмы, которые необходимые, чтобы обеспечить гарантированную доставку пакетов в нужном порядке. Нетрудно представить, что такие приложения как WWW, e-почта, BitTorrent нуждаются в такой поддержке. Однако могут быть и другие случаи. Например, мы передаем видео поток. Пусть какой-то пакет с видеокадром был потерян. Если мы будем ждать пока, будет обнаружена потеря, потерянный кадр будет восстановлен и переслан картинка у клиента «замрет», потребуется буферизация оставшихся кадров и т.д. Проще смириться с

временным ухудшением картинки на экране, из-за потери кадра и продолжать передачу. В таких случаях, когда приложению не нужна гарантированная доставка всех кадров, лучше использовать протокол UDP - User Datagram Protocol. UDP берет пакет у одного приложения и старается его доставить адресату – другому приложению без гарантии доставки.

Таким образом, на транспортном уровне у приложения есть выбор: либо служба надежной доставки с сохранением упорядочения пакетов в потоке, либо негарантированная доставка отдельных пакетов. В первом случае, используют сервис TCP, во втором – сервис UDP.

Наконец, наверху этой 4-х уровневой модели находится Прикладной уровень, где «живут» приложения. Все они используют один из сервисов Транспортного уровня через строго определенный API.

Приложения для взаимодействия между собой используют свой протокол, с помощью которого они определяют синтаксис и семантику предаваемых данных. Например, как мы уже видели WWW-клиент, взаимодействует с WWW-сервером с помощью системы команд, передаваемых вместе с операндами этих команд как ASCII строки. WWW- клиента заботит то, как эти ASCII строки попадут к WWW-серверу, не были при этом повторные передачи пакетов, каких и сколько раз. TCP обеспечивает этот сервис, используя сервис Сетевого уровня.

Следует подчеркнуть, что при взаимодействии каждый уровень взаимодействует с одноименным уровнем на противоположной стороне: Прикладной с прикладным, транспортный с транспортным, сетевой с сетевым и т.д.

Когда приложению требуется передать данные через Интернет, то оно обращается к Транспортному уровню. Транспортный уровень, используя сервис Сетевого уровня, передает их Транспортному уровню на другой стороне. Сетевой уровень нарезает данные на пакты, указывая адрес соседнего маршрутизатора. Так, hop-by-hop, пакеты идут от маршрутизатора к маршрутизатору, пока не достигнут места назначения. Сетевой уровень использует сервис Канального уровня, который умеет передавать кадры через линии Ethernet, WiFi, GSM, DSL и многие другие системы передачи данных [Смелянский т.1].

В 80-е годы международным институтом стандартизации ISO была разработана 7-ми уровневая модель взаимодействия в сетях. Эта модель имеет больше методологический уровень, чем практический. В нем Канальный уровень 4-х уровневой модели представлен 2-я: Канальным и Физическим. Сетевой уровень в обеих моделях более или менее одинаков, Транспортный представлен Транспортным и Сессии, а Прикладной представлен уровнем Представления и Прикладным. Нумерация часто используются 7-ми уровневой. Поэтому часто о Канальном уровне говорят как о L2, о Сетевом – L3, транспортный – L4, хотя в нашей модели это 1, 2 и 3 уровни соответственно [См. Смелянский т.2, гл.1, т.1 гл.1 и гл.2].