# Что такое Интернет?

Поскольку физической основой сети Web является Интернет, то для понимания многих вопросов данного курса потребуется кратко ознакомиться со структурой и протоколами Интернета.

По сути, это самая большая в мире сеть, не имеющая единого центра управления, но работающая по единым правилам и предоставляющая своим пользователям единый набор услуг. Интернет можно рассматривать как "сеть сетей", каждая из которых управляется независимым оператором – поставщиком услуг Интернета (ISP, Internet Service Provider).

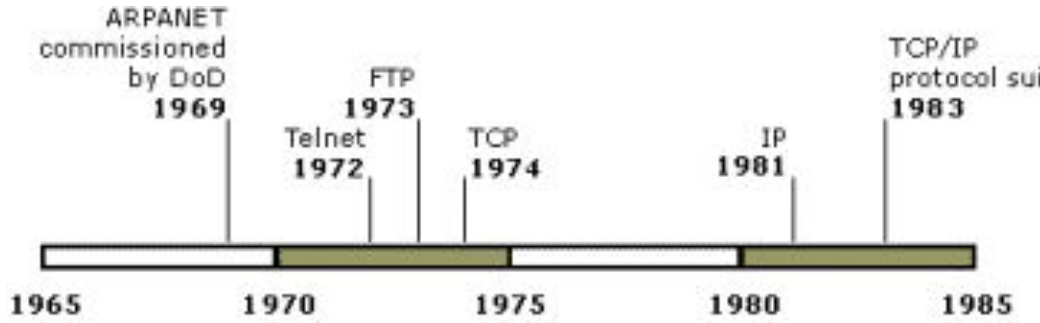
С точки зрения пользователей Интернет представляет собой набор информационных ресурсов, рассредоточенных по различным сетям, включая ISP-сети, корпоративные сети, сети и отдельные компьютеры домашних пользователей. Каждый отдельный компьютер в данной сети называется *хостом* (от английского термина host).

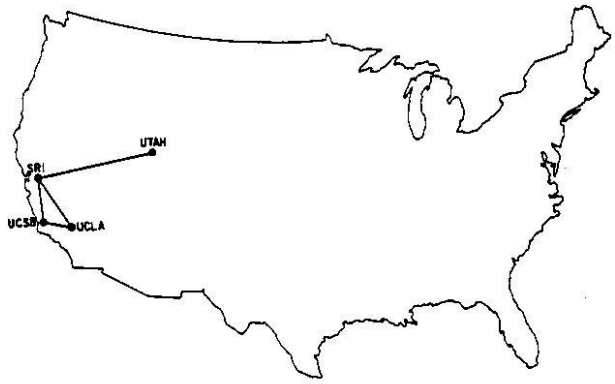
Сегодняшний Интернет обязан своему появлению объединенной сети ARPANET (1969 – 1990), которая начиналась как скромный эксперимент в новой тогда технологии коммутации пакетов.

Сеть ARPANET была развернута в 1969 г. и состояла поначалу всего из четырех узлов с коммутацией пакетов, используемых для взаимодействия горстки хостов и терминалов. Первые линии связи, соединявшие узлы, работали на скорости всего 50 Кбит/с. Сеть ARPANET финансировалась управлением перспективного планирования научно-исследовательских работ ARPA (Advanced Research Projects Agency) министерства обороны США и предназначалась для изучения технологии и протоколов коммутации пакетов, которые могли бы использоваться для кооперативных распределенных вычислений.

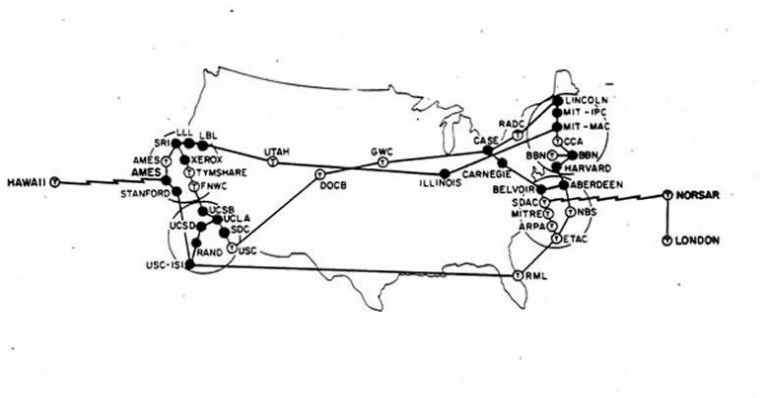
# История развития TCP/IP

Интернет, базирующийся на стеке протоколов TCP/IP, развился из сети ARPANET (Advanced Research Projects Agency Network), построение которой началось в 1969 году в США на базе университетов под надзором министерства обороны.



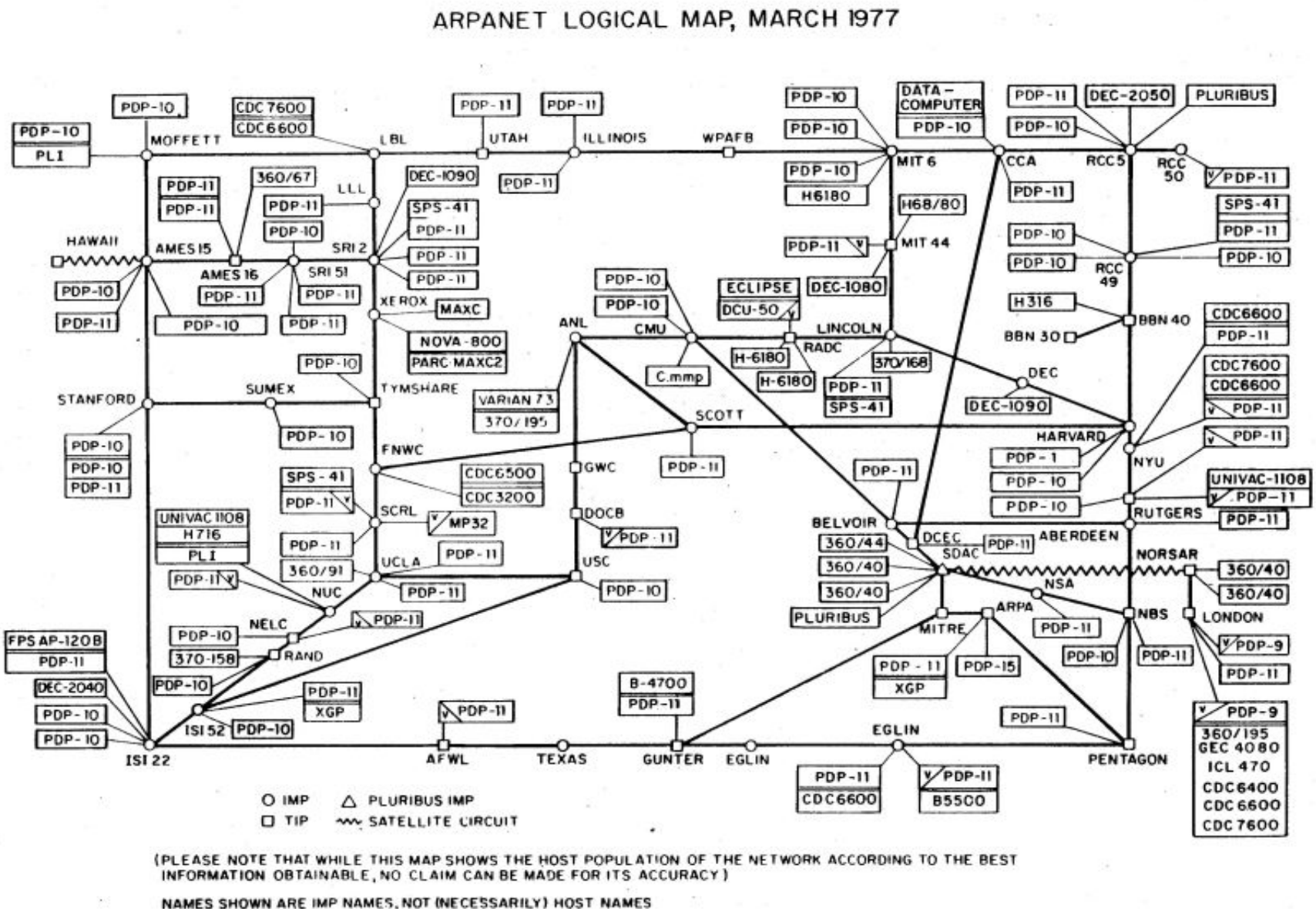




декабрь 1969 года




1973 год

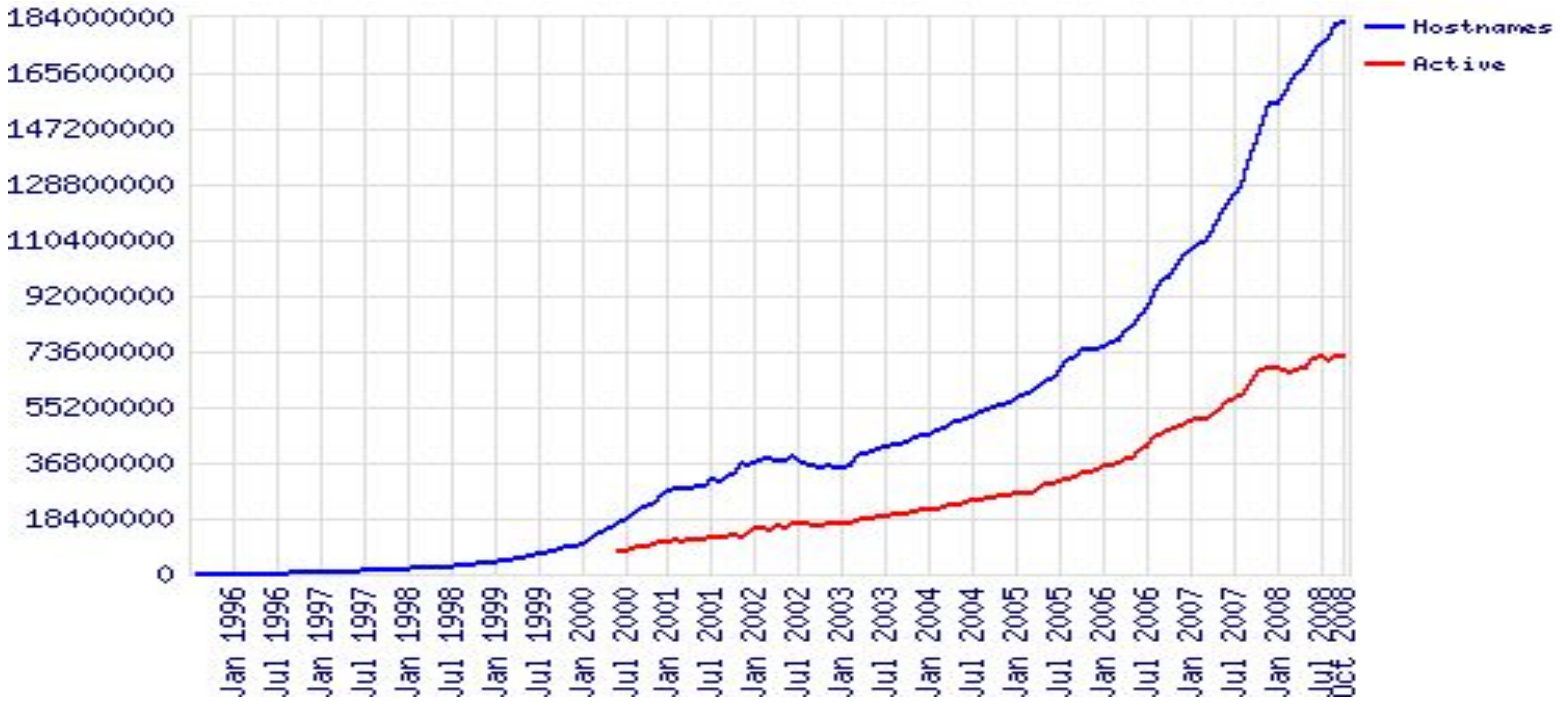
# Логическая схема сети ARPANET




(PLEASE NOTE THAT WHILE THIS MAP SHOWS THE HOST POPULATION OF THE NETWORK ACCORDING TO THE BEST INFORMATION OBTAINABLE, NO CLAIM CAN BE MADE FOR ITS ACCURACY)

NAMES SHOWN ARE IMP NAMES, NOT (NECESSARILY) HOST NAMES

# **Динамика роста числа хостов**
(как формально зарегистрированных, так активно функционирующих)



# Интернет является децентрализованной сетью

**Достоинства:**

Легкость наращивания Интернета путем заключения соглашения между двумя ISP (Internet Service Provider).

**Недостатки:**

- сложность модернизации технологий и услуг Интернета, поскольку требуются согласованные усилия всех поставщиков услуг;
- невысокая надежность услуг Интернета;
- ответственность за работоспособность отдельных сегментов этой сети возлагается на поставщиков услуг Интернета.

**Типы поставщиков услуг Интернета:**

- просто поставщик услуг Интернета выполняет транспортную функцию для конечных пользователей – передачу их трафика в сети других поставщиков услуг Интернета;
- поставщик интернет-контента имеет собственные информационно-справочные ресурсы, предоставляя их содержание в виде веб-сайтов;
- поставщик услуг хостинга предоставляет свои помещения, каналы связи и серверы для размещения внешнего контента;
- поставщик услуг по доставке контента занимается только доставкой контента в многочисленные точки доступа с целью повышения скорости доступа пользователей к информации;
- поставщик услуг по поддержке приложений предоставляет клиентам доступ к крупным универсальным программным продуктам, (например SAP R3);
- поставщик биллинговых услуг обеспечивает оплату счетов по Интернету.

# Иерархическая декомпозиция

Одной из концепций, реализующих декомпозицию - разбиение сложной задачи на несколько более простых задач-модулей, является многоуровневый подход. Такой подход дает возможность проводить разработку, тестирование и модификацию каждого отдельного уровня независимо от других уровней.

*Иерархическая декомпозиция* позволяет, перемещаясь в направлении от более низких к более высоким уровням переходить к более простому представлению решаемой задачи.

Каждый из уровней должен поддерживать *интерфейс* с выше- и нижележащими уровнями собственной иерархии средств и интерфейс со средствами взаимодействия другой стороны на том же уровне иерархии. Данный тип интерфейса называется *протоколом*.

Иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети, называется *стеком протоколов*.

В рамках модели OSI средства взаимодействия делятся на семь уровней: прикладной, представления, сеансовый, транспортный, сетевой, канальный и физический.

В распоряжение программистов предоставляется прикладной программный интерфейс, позволяющий обращаться с запросами к самому верхнему уровню, а именно, - уровню приложений.

# Стандартизация - документы *RFC*

*RFC* (англ. Request for Comments) — документ из серии пронумерованных информационных документов Интернета, содержащих технические спецификации и Стандарты, широко применяемые во Всемирной сети.

В настоящее время первичной публикацией документов *RFC* занимается *IETF (Инженерный совет Интернет - **Internet Engineering Task Force** )* под эгидой открытой организации Общество Интернета ( *ISOC* ). Правами на *RFC* обладает именно Общество Интернет.

Формат *RFC* появился в 1969 г. при обсуждении проекта *ARPANET*. Первые RFC распространялись в печатном виде на бумаге в виде обычных писем, но уже с декабря 1969 г., когда заработали первые сегменты ARPANET, документы начали распространяться в электронном виде.

# Примеры популярных RFC-документов

| Номер RFC | Тема |
|---|---|
| RFC 768 | UDP (User Datagram Protocol) |
| RFC 791 | IP |
| RFC 793 | TCP (Transmission Control Protocol). |
| RFC 822 | Формат электронной почты, заменен RFC 2822 |
| RFC 959 | FTP |
| RFC 1034 | DNS (Domain Name System) — концепция |
| RFC 1035 | DNS — внедрение |
| RFC 1591 | Структура доменных имен |
| RFC 1738 | URL |
| RFC 1939 | Протокол POP3 (Post Offise Protocol) |

(Протокол обмена почтовой информацией POP3 предназначен для разбора почты из почтовых ящиков пользователей на их рабочие места при помощи программ-клиентов. Если по протоколу SMTP (Simple Mail Transfer Protocol) пользователи отправляют корреспонденцию через Интернет, то по протоколу POP3 пользователи получают корреспонденцию из своих почтовых ящиков на почтовом сервере в локальные файлы)

| | |
|---|---|
| RFC 2026 | Процесс стандартизации в Интернете |
| RFC 2045 | MIME |
| RFC 2231 | Кодировка символов |
| RFC 2616 | HTTP |
| RFC 2822 | Формат электронной почты |
| RFC 3501 | IMAP (Internet Message Access Protocol |

- протокол прикладного уровня для доступа к электронной почте)

Все *Рекомендации* W3C открыты, то есть, не защищены патентами и могут внедряться любым человеком без каких-либо финансовых отчислений Консорциуму.

# Стек протоколов TCP/IP

- Эти протоколы изначально ориентированы на глобальные сети, в которых качество соединительных каналов не идеально. Он позволяет создавать глобальные сети, компьютеры в которых соединены друг с другом самыми разными способами от высокоскоростных оптоволоконных кабелей и спутниковых каналов до коммутируемых телефонных линий. TCP/IP соответствует модели OSI достаточно условно и содержит 4 уровня. Прикладной уровень стека соответствует трем верхним уровням модели OSI: прикладному, представления и сеансовому.
- В сети данные всегда передаются блоками относительно небольшого размера. Каждый блок имеет префиксную часть (заголовок), описывающую содержимое блока, и суффиксную, содержащую, например, информацию для контроля целостности передаваемого блока данных.
- Название стека протоколов TCP/IP состоит из названий двух разных протоколов. Протокол IP (Internet Protocol) представляет собой протокол нижнего (сетевого) уровня и отвечает за передачу пакетов данных в сети. Он относится к так называемым протоколам *датаграмм* и работает без подтверждений. Последнее означает, что при его использовании доставка пакетов данных не гарантируется и не подтверждается. Не гарантируется также и то, что пакеты достигнут пункта назначения в той последовательности, в которой они были отправлены.
- К протоколам сетевого уровня относится также протокол межсетевых управляющих сообщений *ICMP* (Internet Control Message Protocol), предназначенный для передачи маршрутизатором источнику информации об ошибках при передаче пакета.
- Очевидно, что намного удобнее передавать данные по каналу, который работает корректно, доставляя все пакеты по порядку. Поэтому над протоколом IP работает протокол передачи данных более высокого (транспортного) уровня — TCP (Transmission Control Protocol). Посылая и принимая пакеты через протокол IP, протокол TCP гарантирует доставку всех переданных пакетов данных в правильной последовательности.
- Следует отметить, что при использовании протокола IP обеспечивается более быстрая передача данных, так как не тратится время на подтверждение приема каждого пакета. Есть и другие преимущества. Одно из них заключается в том, что он позволяет рассылать пакеты данных в широковещательном режиме, при котором они достигают всех компьютеров физической сети. Что же касается протокола TCP, то для передачи данных с его помощью необходимо создать канал связи между компьютерами. Он и создается с использованием протокола IP.

# Структура и принципы WWW

- Сеть WWW образуют миллионы *веб-серверов*, расположенных по всему миру. *Веб - сервер* является программой, запускаемой на подключенном к сети компьютере и передающей данные по протоколу HTTP.
- Для идентификации ресурсов (зачастую файлов или их частей) в WWW используются идентификаторы ресурсов *URI* (Uniform Resource Identifier).
- Для определения местонахождения ресурсов в этой сети используются локаторы ресурсов *URL* (Uniform Resource Locator). Такие URL-локаторы представляют собой комбинацию URI и системы DNS.
- Доменное имя (или IP-адрес) входит в состав URL для обозначения компьютера (его сетевого интерфейса), на котором работает программа веб-сервер.
- На клиентском компьютере для просмотра информации, полученной от веб-сервера, применяется специальная программа - *веб-браузер*. Основная функция веб-браузера - отображение гипертекстовых страниц (веб-страниц). Множество веб-страниц образуют *веб-сайт*.

# Прокси-сервер

- *Прокси-сервер* (proxy-server) - служба в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам.
- Сначала клиент подключается к прокси-серверу и запрашивает какой-либо ресурс, расположенный на другом сервере. Затем прокси-сервер либо подключается к указанному серверу и получает ресурс у него, либо возвращает ресурс из собственного *кеша* (если имеется). В некоторых случаях запрос клиента или ответ сервера может быть изменен прокси-сервером в определенных целях. Также прокси-сервер позволяет защищать клиентский компьютер от некоторых сетевых атак.
- Чаще всего прокси-серверы применяются для следующих целей:
  - обеспечение доступа с компьютеров локальной сети в Интернет;
  - кеширование данных: если часто происходят обращения к одним и тем же внешним ресурсам, то можно держать их копию на прокси-сервере и выдавать по запросу, снижая тем самым нагрузку на канал во внешнюю сеть и ускоряя получение клиентом запрошенной информации.
  - сжатие данных: прокси-сервер загружает информацию из Интернета и передает информацию конечному пользователю в сжатом виде.
  - защита локальной сети от внешнего доступа: например, можно настроить прокси-сервер так, что локальные компьютеры будут обращаться к внешним ресурсам только через него, а внешние компьютеры не смогут обращаться к локальным вообще (они "видят" только прокси-сервер).
  - ограничение доступа из локальной сети к внешней: например, можно запретить доступ к определенным веб-сайтам, ограничить использование интернета каким-то локальным пользователям, устанавливать квоты на трафик или полосу пропускания, фильтровать рекламу и вирусы.
  - анонимизация доступа к различным ресурсам. Прокси-сервер может скрывать сведения об источнике запроса или пользователе. В таком случае целевой сервер видит лишь информацию о прокси-сервере, например, IP-адрес, но не имеет возможности определить истинный источник запроса. Существуют также искажающие прокси-серверы, которые передают целевому серверу ложную информацию об истинном пользователе.

# Протоколы Интернет прикладного уровня

**Самый верхний уровень в иерархии протоколов Интернет занимают следующие протоколы прикладного уровня:**

- *DNS* - распределенная система доменных имен, которая по запросу, содержащему доменное имя хоста сообщает IP адрес;
- *HTTP* - протокол передачи гипертекста в Интернет;
- *HTTPS* - расширение протокола HTTP, поддерживающее шифрование;
- *FTP* (File Transfer Protocol - RFC 959) - протокол, предназначенный для передачи файлов в компьютерных сетях;
- *Telnet* (TELecommunication NETwork - RFC 854) - сетевой протокол для реализации текстового интерфейса по сети;
- *SSH* (Secure Shell - RFC 4251) - протокол, позволяющий производить удаленное управление операционной системой и передачу файлов. В отличие от Telnet шифрует весь трафик;
- *POP3* – протокол почтового клиента, который используется почтовым клиентом для получения сообщений электронной почты с сервера;
- *IMAP* - протокол доступа к электронной почте в Интернет;
- *SMTP* – протокол, который используется для отправки почты от пользователей к серверам и между серверами для дальнейшей пересылки к получателю;
- *LDAP* - протокол для доступа к службе каталогов X.500, является широко используемым стандартом доступа к службам каталогов;
- *XMPP* (Jabber) - основанный на XML расширяемый протокол для мгновенного обмена сообщениями в почти реальном времени;
- *SNMP* - базовый протокол управления сети Internet.

## Почтовые протоколы

Хотя Telnet и FTP были (и остаются) полезными, первым приложением, совершившим переворот в сознании пользователей компьютеров сети ARPANET, стала электронная почта.

До сети ARPANET существовали системы электронной почты, но все они были однокомпьютерными системами. В 1972 г. *Рэй Томлинсон* (Ray Tomlinson) из компании BBN написал первый пакет, предоставляющий распределенные почтовые услуги в компьютерной сети из нескольких компьютеров.

Уже к 1973 г. исследования управления ARPA показали, что три четверти всего трафика сети ARPANET составляла электронная почта. Польза электронной почты оказалась столь велика, что все больше пользователей стремилось подключиться к сети ARPANET, в результате чего возрастала потребность в добавлении новых узлов и использовании высокоскоростных линий. Таким образом, появилась тенденция, сохраняющаяся и по сей день.

- *POP3* (Post Office Protocol Version 3 - RFC 1939) — протокол, который используется почтовым клиентом для получения сообщений электронной почты с почтового сервера;
- *IMAP* (Internet Message Access Protocol - RFC 3501) — протокол доступа к электронной почте. Аналогичен POP3, однако предоставляет пользователю богатые возможности для работы с почтовыми ящиками, находящимися на центральном сервере. Электронными письмами можно манипулировать с компьютера пользователя (клиента) без необходимости постоянной пересылки с сервера и обратно файлов с полным содержанием писем.
- *SMTP* (Simple Mail Transfer Protocol — RFC 2821) — протокол, предназначенный для передачи электронной почты. Используется для отправки почты от пользователей к серверам и между серверами для дальнейшей пересылки к получателю. Для приема почты почтовый клиент должен использовать протоколы POP3 или IMAP.

## Протокол HTTP

Базовым протоколом сети гипертекстовых ресурсов Веб является протокол HTTP. В его основу положено взаимодействие " *клиент-сервер* ", то есть предполагается, что:

- потребитель- *клиент* инициировав соединение с поставщиком-сервером посылает ему запрос;

- поставщик- *сервер*, получив запрос, производит необходимые действия и возвращает обратно клиенту ответ с результатом.

При этом возможны два способа организации работы компьютера-клиента:

*Тонкий клиент* - это компьютер-клиент, который переносит все задачи по обработке информации на сервер. Примером тонкого клиента может служить компьютер с браузером, использующийся для работы с веб-приложениями.

*Толстый клиент*, напротив, производит обработку информации независимо от сервера, использует последний в основном лишь для хранения данных.

## Структура базового протокола HTTP.

(RFC 1945, RFC 2616 - протокол прикладного уровня для передачи гипертекста)

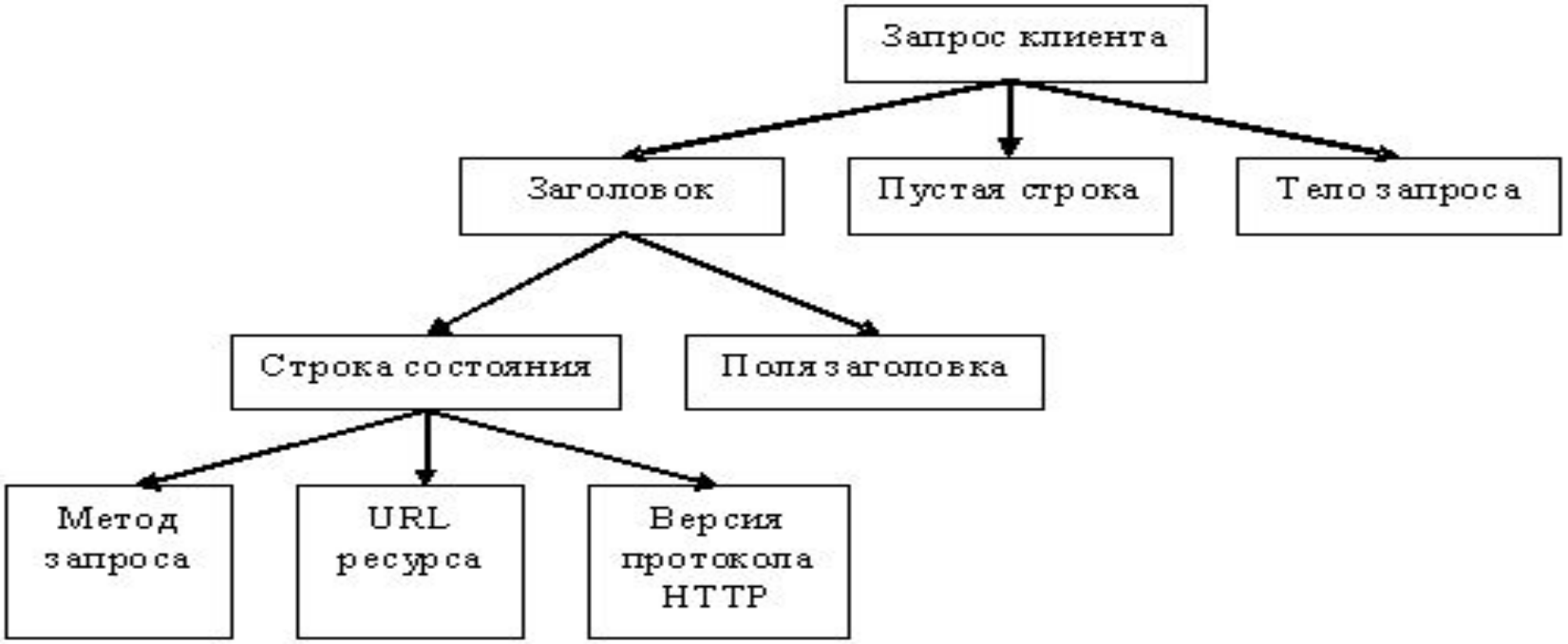
Центральным объектом в HTTP является *ресурс*, на который указывает URI в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере файлы.

Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя изначально данный протокол предназначен для передачи символьной информации. На первый взгляд это может показаться излишней тратой ресурсов. Действительно, данные в символьном виде занимают больше памяти, сообщения создают дополнительную нагрузку на каналы связи, однако подобный формат имеет много преимуществ (удобочитаемость, простота устранения ошибок).

# Классическая схема HTTP-сеанса:

- Установление TCP-соединения;
- Запрос клиента;
- Ответ сервера;
- Разрыв TCP-соединения.

Обычно запрос клиента представляет собой требование передать HTML-документ или какой-нибудь другой ресурс, а ответ сервера содержит код этого ресурса.



В состав HTTP-запроса, передаваемого клиентом серверу, входят следующие компоненты:

- Строка состояния (иногда для ее обозначения используют также термины строка-статус, или строка запроса).
- Поля заголовка.
- Пустая строка.
- Тело запроса.

*Строку состояния* вместе с *полями заголовка* иногда называют также *заголовком запроса*.

## Формат строки состояния

метод_запроса URL_ресурса версия_протокола_HTTP

**Метод**, указанный в строке состояния, определяет способ воздействия на ресурс, URL которого задан в той же строке.

Метод может принимать значения **GET, POST, HEAD, PUT, DELETE** и т.д. Несмотря на обилие методов, для веб-программиста по-настоящему важны лишь два из них: **GET и POST**.

**GET.** Согласно формальному определению, метод GET предназначается для получения ресурса с указанным URL.

Получив запрос GET, сервер должен прочитать указанный ресурс и включить код ресурса в состав ответа клиенту. Ресурс, URL которого передается в составе запроса, не обязательно должен представлять собой HTML-страницу, файл с изображением или другие данные. URL ресурса может указывать на исполняемый код программы, который, при соблюдении определенных условий, должен быть запущен на сервере. В этом случае клиенту возвращается не код программы, а данные, сгенерированные в процессе ее выполнения. Несмотря на то что, по определению, метод GET предназначен для получения информации, он может применяться и в других целях. Метод GET вполне подходит для передачи небольших фрагментов данных на сервер.

**POST.** Согласно тому же формальному определению, основное назначение метода POST - передача данных на сервер. Однако, подобно методу GET, метод POST может применяться по-разному и нередко используется для получения информации с сервера. Как и в случае с методом GET, URL, заданный в строке состояния, указывает на конкретный ресурс. Метод POST также может использоваться для запуска процесса.

Методы HEAD и PUT являются модификациями методов GET и POST.

**Версия протокола** HTTP, как правило, задается в следующем формате:

HTTP/версия.модификация

**Поля заголовка**, следующие за строкой состояния, позволяют уточнять запрос, т.е. передавать серверу дополнительную информацию. Поле заголовка имеет следующий формат:

Имя_поля: Значение Назначение поля определяется его именем, которое отделяется от значения двоеточием.

| **Поля заголовка HTTP -запроса** | **Значение** |
|---|---|
| Host | Доменное имя или IP-адрес узла, к которому обращается клиент |
| Referer | URL документа, который ссылается на ресурс, указанный в строке состояния |
| From | Адрес электронной почты пользователя, работающего с клиентом |
| Accept | MIME-типы данных, обрабатываемых клиентом. Это поле может иметь несколько значений, отделяемых одно от другого запятыми. Часто поле заголовка Accept используется для того, чтобы сообщить серверу о том, какие типы графических файлов поддерживает клиент |
| Accept-Language | Набор двухсимвольных идентификаторов, разделенных запятыми, которые обозначают языки, поддерживаемые клиентом |
| Accept-Charset | Перечень поддерживаемых наборов символов |
| Content-Type | MIME-тип данных, содержащихся в теле запроса (если запрос не состоит из одного заголовка) |
| Content-Length | Число символов, содержащихся в теле запроса (если запрос не состоит из одного заголовка) |
| Range | Присутствует в том случае, если клиент запрашивает не весь документ, а лишь его часть |
| Connection | Используется для управления TCP-соединением. Если в поле содержится Close, это означает, что после обработки запроса сервер должен закрыть соединение. Значение Keep-Alive предлагает не закрывать TCP- соединение, чтобы оно могло быть использовано для последующих запросов |
| User-Agent | Информация о клиенте |

# Пример HTML-запроса, сгенерированного браузером

GET http://oak.oakland.edu/ HTTP/1.0

Connection: Keep-Alive

User-Agent: Mozilla/4.04 [en]  (Win95;  I)

Host: oak.oakland.edu

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*

Accept-Language: en

Accept-Charset: iso-8859-l,*,utf-8

Получив от клиента запрос, сервер должен ответить ему. Подобно запросу клиента, ответ сервера также состоит из четырех компонентов:

- Строка состояния.
- Поля заголовка.
- Пустая строка.
- Тело ответа.

Ответ сервера клиенту начинается со строки состояния

Версия_протокола Код_ответа Пояснительное_сообщение

**Версия_протокола** задается в том же формате, что и в запросе клиента, и имеет тот же смысл.

**Код_ответа** - это трехзначное десятичное число, представляющее в закодированном виде результат обслуживания запроса сервером.

**Пояснительное_сообщение** дублирует код ответа в символьном виде. Это строка символов, которая не обрабатывается клиентом. Она предназначена для системного администратора или оператора, занимающегося обслуживанием системы, и является расшифровкой кода ответа.

В теле ответа содержится код ресурса, передаваемого клиенту в ответ на запрос. Это не обязательно должен быть HTML-текст веб-страницы. В составе ответа могут передаваться изображение, аудио-файл, фрагмент видеоинформации, а также любой другой тип данных, поддерживаемых клиентом.

О том, как следует обрабатывать полученный ресурс, клиенту сообщает содержимое поля заголовка **Content - type**.

- В качестве значения этого поля указывается **MIME -тип** содержимого запроса или ответа. MIME -тип также передается в поле заголовка Accept, присутствующего в запросе.
- Спецификация **MIME (Multipurpose Internet Mail Extension** - многоцелевое почтовое расширение Internet) первоначально была разработана для того, чтобы обеспечить передачу различных форматов данных в составе электронных писем. Однако применение MIME не исчерпывается электронной почтой. Средства MIME успешно используются в WWW и, по сути, стали неотъемлемой частью этой системы.
- Стандарт MIME разработан как расширяемая спецификация, в которой подразумевается, что число типов данных будет расти по мере развития форм представления данных. Каждый новый тип в обязательном порядке должен быть зарегистрирован в IANA (Internet Assigned Numbers Authority).
- До появления MIME компьютеры, взаимодействующие по протоколу HTTP, обменивались исключительно текстовой информацией. Для передачи изображений, как и для передачи любых других двоичных файлов, приходилось пользоваться протоколом FTP.
- В соответствии со спецификацией MIME, для описания формата данных используются *тип* и *подтип*. *Тип* определяет, к какому классу относится формат содержимого HTTP-запроса или HTTP-ответа. *Подтип* уточняет формат. Тип и подтип отделяются друг от друга косой чертой: тип/подтип
- Поскольку в подавляющем большинстве случаев в ответ на запрос клиента сервер возвращает исходный текст HTML-документа, то в поле Content-type ответа обычно содержится значение **text/html**. Здесь идентификатор text описывает тип, сообщая, что клиенту передается символьная информация, а идентификатор html описывает подтип, т.е. указывает на то, что последовательность символов, содержащаяся в теле ответа, представляет собой описание документа на языке HTML

# Обеспечение безопасности передачи данных HTTP

- Поскольку протокол HTTP предназначен для передачи символьных данных в открытом (незашифрованном) виде, то лица, имеющие доступ к каналу передачи данных между клиентом и сервером, могут без труда просматривать весь трафик и использовать его для совершения несанкционированных действий. В связи с этим предложен ряд расширений базового протокола, направленных на повышение защищенности интернет-трафика от несанкционированного доступа.
- Самым простейшим является расширение HTTPS, при котором данные, передаваемые по протоколу HTTP, "упаковываются" в криптографический протокол SSL или TLS, тем самым обеспечивая защиту этих данных. В отличие от HTTP, для HTTPS по умолчанию используется TCP-порт 443.
- SSL (Secure Sockets Layer) - криптографический протокол, обеспечивающий безопасную передачу данных по сети Интернет. При его использовании создается защищенное соединение между клиентом и сервером. SSL изначально разработан компанией Netscape Communications. Впоследствии на основании протокола SSL 3.0 был разработан и принят стандарт RFC, получивший название TLS. Этот протокол использует шифрование с открытым ключом для подтверждения подлинности передатчика и получателя. Поддерживает надежность передачи данных за счет использования корректирующих кодов и безопасных хэш-функций.
- Для доступа к веб-страницам, защищенным протоколом SSL, в URL вместо схемы http, как правило, подставляется схема https, указывающая на то, что будет использоваться SSL-соединение. Стандартный TCP-порт для соединения по протоколу https — 443. Для работы SSL требуется, чтобы на сервере имелся SSL -*сертификат*.

# В сети Веб поддерживаются 3 типа аутентификации при клиент-серверных взаимодействиях:

- Basic - базовая аутентификация, при которой имя пользователя и пароль передаются в заголовках http-пакетов. Пароль при этом не шифруется и присутствует в чистом виде в кодировке base64. Для данного типа аутентификации использование SSL является обязательным.
- Digest - дайджест-аутентификация, при которой пароль пользователя передается в хешированном виде. По уровню конфиденциальности паролей этот тип мало чем отличается от предыдущего, так как атакующему все равно, действительно ли это настоящий пароль или только хеш от него: перехватив удостоверение, он все равно получает доступ к конечной точке. Для данного типа аутентификации использование SSL является обязательным.
- Integrated - интегрированная аутентификация, при которой клиент и сервер обмениваются сообщениями для выяснения подлинности друг друга с помощью протоколов NTLM или Kerberos. Этот тип аутентификации защищен от перехвата удостоверений пользователей, поэтому для него не требуется протокол SSL. Только при использовании данного типа аутентификации можно работать по схеме http, во всех остальных случаях необходимо использовать схему https.

# Популярно о протоколах 1

***Сжатие*:** У тебя отрезают левую руку на входе, а на выходе - пришивают клонированную правую (и зеркально повернутую, разумеется). То же с ногами и вообще со всем, что имеет регулярную структуру.

***Коррекция ошибок*:** К спине пришивают твою же фотографию. Если на выходе ты не похож - корректируют лицо.

***Время жизни пакета*:** Все перемещения по коридору - пока горит спичка. Не успел - умри героем.

***DNS*:** Чтобы узнать, где колодец в деревне Гадюкино, ты сначала идешь к президенту, потом к губернатору и т. д.

***Динамический IP*:** Каждое утро все меняются паспортами.

***Текст-ориентированный протокол*:** Вместо тебя отправляют твой словесный портрет.

# Популярно о протоколах 2

***MIME-код*:** Справка, что ты не верблюд.

***Уровни протоколов*:** Чистое поле. Нужно перейти от одного края к другому. Строится огромная арка, внутри арки мостовая, посреди мостовой кладут ж/д полотно, к рельсам приваривают сваи и на них ставят огромную гранитную глыбу с туннелем внутри, в туннеле прокладывают трубу диаметром полметра, по которой ты и ползешь пока горит спичка к президенту (сжатый и с коррекцией ошибок).

***Пинги*:** Иди посмотри, Иван Петрович не ушел еще?.

***Маскарадинг*:** Один паспорт на всю семью.

***IPv6*:** Китайский паспорт.