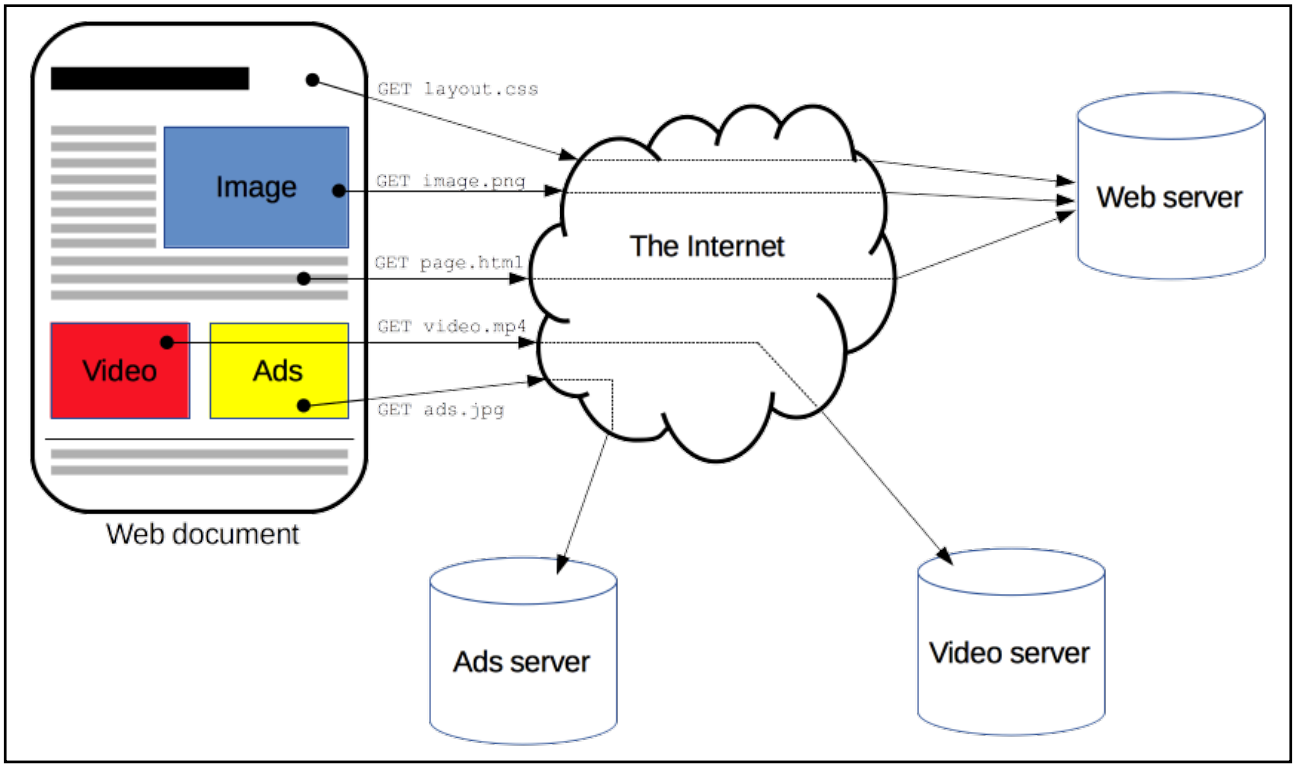
Занятие 2. Протоколы прикладного уровня. Протокол HTTP. Основные заголовки протокола HTTP. Заголовки X-* протокола HTTP и их использование

Протоколы прикладного уровня обеспечивают взаимодействие сети и пользователя. В качестве транспорта протоколы прикладного уровня могут использовать любой протокол транспортного уровня, в основном используются протоколы TCP или UDP. Стандарты протоколов прикладного уровня могут разрабатываться как под конкретный тип сервиса/ сервера, например, WEB-сервер, сервис электронной почты, FTP-сервер, сервер разрешения доменных имен DNS и т.п., так и производителями программного обеспечения или оборудования для собственных нужд. Кроме того некоторые протоколы прикладного уровня могут сами являться транспортами для сервисов и движков, работающих поверх сервисов этих протоколов. Например, поверх протокола HTTP работают такие протоколы как XML-API сервисов онлайн-бронирования, JSON-API яваскриптового движка React и т.п.

Протокол HTTP

ВЕРСИИ ПРОТОКОЛА HTTP

Протокол HTTP (HyperText Transfer Protocol) - это протокол прикладного уровня передачи данных, изначально в виде текстовых документов в определенном формате, в настоящее время используется для передачи произвольных данных и даже потоков данных.



Первая реализация протокола HTTP была создана в 1989-1991 году и включала в себя четыре составляющие:

- Текстовый формат для передаваемых по протоколу документов, HTML (HyperText Markup Language)
- Простая реализация протокола для передачи документов в формате HTML, HTTP
- Клиент для отображения передаваемых по протоколу документов, браузер, самый первый имел название WorldWideWeb
- Сервер для обслуживания клиентских запросов и отдачи документов, ранняя версия web-сервера httpd

Первая реализация протокола была очень проста, клиент мог получить конкретный документ одним запросом по заранее установленному между ним и сервером соединению, позднее эта реализация получила название версии HTTP/0.9. Сам протокол, а так же клиенты и серверы продолжили развиваться и следующая версия протокола HTTP/1.0, появилась возможность передавать документы, отличные от текстовых документов в формате HTML, были добавлены заголовки как в клиентских запросах, так и ответах сервера, коды статуса ответа сервера. Постепенно эти возможности встраивались как в клиентов, так и в web-сервера. Версия протокола HTTP/1.0 так и не стала стандартом, лишь сводом общих правил и применений. Первым официальным стандартом протокола стала версия HTTP/1.1. По сравнению с предыдущей версией HTTP/1.1 получил ряд существенных улучшений: повторное использование уже установленных соединений между клиентом и сервером, возможность конвейерной обработки запросов, передачи ответа от сервера клиенту по частям, дополнительные кеширующий механизмы, широкие возможности согласования поддерживаемых типов и параметров передаваемых данных, поддержку размещения на сервере с одним IP-адресом нескольких имен хостов/доменов, что открыло возможности услуг по размещению сайтов на серверах провайдеров. Версия протокола HTTP/1.1 была очень стабильна (в части изменений протокола) и широко используется в течение более чем 15 лет. Но с каждым годом сайты становятся все сложнее и функциональнее, становятся практически полноценными приложениями, данные, передаваемые по протоколу HTTP все более объемными. В 2015 году был официально стандартизована, опубликована и введена новая версия протокола HTTP/2, в отличие от предшествующей версии протокол стал не текстовым, а бинарным, поддерживает параллельные запросы внутри одного соединения между клиентом и сервером, обладает более сжатыми заголовками, чтобы сократить объемы передаваемых служебных данных, сервер получил возможность принудительно обновлять кешированные данные на стороне клиента. Большая часть мирового WEB-трафика идет по протоколу версии HTTP/2. И развитие протокола HTTP на этом не останавливается.

Взаимодействие клиента и сервера происходит при помощи отправки сообщений. Отправляемое клиентом сообщение серверу называется запрос (Request), сообщение от сервера клиенту называется ответ (Response). В общем виде сообщение состоит из трех частей: стартовая строка (Starting Line, определяющая тип сообщения), заголовки (Headers) и тело сообщения (Message Body, в некоторых ситуациях может отсутствовать).

Стартовая строка запроса содержит HTTP-метод, URI (Universal Resource Identifier, последовательность символов, идентифицирующая абстрактный или физический ресурс) ресурса, версию протокола. Стартовая строка ответа содержит версию протокола, код статуса ответа (Status Code), название кода статуса.

ПРИМЕР ПРОСТОГО HTTP-ЗАПРОСА БЕЗ ТЕЛА:

```
GET / HTTP/2

Host: www.google.ru

Accept: */*

User-Agent: Mozilla/5.0 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
88.0.4324.96 Safari/537.36
```

ПРИМЕР HTTP-ОТВЕТА:

```
HTTP/2 200

Date: Tue, 02 Feb 2021 01:01:01 GMT

Expires: -1

Cache-Control: private, max-age=0

Content-Type: text/html; charset=UTF-8

Strict-Transport-security: max-age=31536000

Server: gws

X-XSS-Protection: 0

X-Frame-Options: SAMEORIGIN

Set-Cookie: 1P_JAR=2021-02-02-01; expires=Thu, 04-Mar-2021 01:01:01 GMT;
path=/; domain=.google.ru; Secure; SameSite=none

Set-Cookie: NID=208=K… expires=Wed, 04-Aug-2021 01:01:01 GMT; path=/;
domain=.google.ru; Secure; HttpOnly; SameSite=none

Accept-Ranges: none

Vary: Accept-Encoding

<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"
lang="ru"><head><meta charset="UTF-8"><meta conte…
```

НЕКОТОРЫЕ МЕТОДЫ HTTP-ЗАПРОСА

В зависимости от того, какое действие клиент хочет совершить относительно указанного в URI ресурса, он должен в запросе указывать определенный метод, поддерживаемый сервером.

GET - запрашивает документ по URI ресурса, запросы с использованием этого метода обычно могут только получать данные с сервера

HEAD - запрашивает заголовки ответа от сервера по URI ресурса, тело документа не передается клиенту

POST - используется для отправки данных на сервер определенному в URI ресурсу, обычно вызывает изменения данных, находящихся на сервере, или их состояния

PUT - используется для замены существующего на сервере ресурса по заданному URI данными из запроса

DELETE - удаляет на сервере ресурс по заданному URI

OPTIONS - используется для получения описания параметров соединения с сервером, возможных методов

Поскольку не все методы являются «безопасными» для ресурсов сервера, то обычно не все методы и поддерживаются WEB-серверами. Например, часто для определенных URI или частей сайта разрешаются только методы GET и HEAD от клиента, для части GET, HEAD и POST, а методы PUT, DELETE полностью запрещаются.

КОДЫ СТАТУСОВ HTTP-ОТВЕТА

Код ответа указывает, был ли успешно выполнен запрос клиента, а так же, какие действия клиент может или должен дальше предпринять. Основные коды ответа определены в стандарте, однако WEB-серверы могут реализовать свои собственные коды ответов. Все коды сгруппированы в пять классов:

- Информационные 100-199
- Успешные 200-299
- Перенаправление запроса 300-399
- Ошибки клиента 400-499
- Ошибки сервера 500-599

Код	Название	Описание	Версия протокола
Информационные			
100	Continue	Промежуточный ответ сервера, означающий, что клиент может продолжить передавать запросы	Только HTTP/1.1
101	Switching Protocol	Ответ указывает, что сервер переключился на использование версии протокола, запрошенной клиентом в заголовке Upgrade	Только HTTP/1.1
102	Processing	Ответ указывает, что сервер получил запрос, но его обработка еще не завершена	Только HTTP/1.1
103	Early Hints	В ответе указываются ресурсы, которые могут быть загружены, пока готовится тело основного ответа от сервера	Только HTTP/1.1
Успешные			
200	OK	Запрос успешно обработан	HTTP/0.9 и выше
201	Created	Успешный ответ на запрос клиента с методом PUT, означает успешную замену ресурса на сервере содержимым запроса клиента	HTTP/0.9 и выше
202	Accepted	Не поддерживаемый ответ, означает, что запрос принят, но еще не обработан	HTTP/0.9 и выше
203	Non-Authoritative Information	Означает, что ответ получен не от исходного сервера, а из другого источника	HTTP/0.9 и HTTP/1.1
204	No Content	Нет содержимого, в ответ клиенту посылаются только заголовки	HTTP/0.9 и выше
205	Reset Content	Означает, что клиенту необходимо сбросить отображение содержимого документа, породившего запрос	Только HTTP/1.1
206	Partial Content	Код используется, когда клиент осуществляет загрузку содержимого по частям в несколько потоков и запрашивает определенный диапазон содержимого ресурса	Только HTTP/1.1
Перенаправление запроса			
300	Multiple Choice	Означает, что запрос имеет более одного возможного варианта ответа	HTTP/0.9 и выше

Код	Название	Описание	Версия протокола
301	Moved Permanently	Означает, что URI запрашиваемого ресурса изменен, новый URI должен быть отправлен в специальном заголовке ответа	HTTP/0.9 и выше
302	Found	Означает, что ресурс временно изменен и клиенту понадобится использовать его в будущих запросах (например, для обновления локального кеша клиента)	HTTP/0.9 и выше
303	See Other	Означает, что клиент будет перенаправлен для получения ресурса в другой URI методом GET	HTTP/0.9 и HTTP/1.1
304	Not Modified	Используется для локального кеширования на клиента, означает, что ресурс не был изменен	HTTP/0.9 и выше
305	Use Proxy	Означает, что запрашиваемый ресурс может быть доступен только через proxy-сервер	Только HTTP/1.1
306	Switch Proxy	Не используется, но подразумевалось сообщение клиенту о необходимости использовать указанный proxy-сервер во всех последующих запросах	Только HTTP/1.1
307	Temporary Redirect	Означает, что клиент должен получить ресурс по другому URI с использованием того же метода, что и в исходном запросе	Только HTTP/1.1
308	Permanent Redirect	Аналогично коду 301	В разработке
Ошибки клиента			
400	Bad Request	Означает, что сервер не понимает запрос из-за неправильного синтаксиса или не принимает неверно сформированный запрос клиента	HTTP/0.9 и выше
401	Unauthorized	Для получения запрашиваемого ответа требуется аутентификация клиента	HTTP/0.9 и выше
402	Payment Required	Зарезервирован для будущего использования, предполагался для использования в цифровых платежных системах	HTTP/0.9 и HTTP/1.1
403	Forbidden	У клиента нет прав на доступ к запрашиваемому ресурсу	HTTP/0.9 и выше
404	Not Found	Запрашиваемый ресурс не найден	HTTP/0.9 и выше
405	Method Not Allowed	На сервере запрещено использование метода из запроса клиента	Только HTTP/1.1
406	Not Acceptable	Означает, что сервер не смог найти контент, удовлетворяющий требованиям из заголовков запроса клиента	Только HTTP/1.1
407	Proxy Authentication Required	Требуется аутентификация для proxy-сервера	Только HTTP/1.1
408	Request Timeout	Означает, что сервер хотел бы прервать неиспользуемое соединение	Только HTTP/1.1
409	Conflict	Означает, что запрос конфликтует с текущим состоянием сервера	Только HTTP/1.1

Код	Название	Описание	Версия протокола
410	Gone	Означает, что запрашиваемый ресурс удален с сервера	Только HTTP/1.1
411	Length Required	Сервер требует указание заголовка Content-Length в запросе клиента	Только HTTP/1.1
412	Precondition Failed	Указанные клиентом условия в заголовках не могут быть выполнены сервером	Только HTTP/1.1
413	Request Entity Too Large	Размер запроса превышает лимит, установленный на сервере	Только HTTP/1.1
414	Request-URI Too Long	В запросе клиента указан слишком длинный URI	Только HTTP/1.1
415	Unsupported Media Type	Формат запрашиваемых данных не поддерживается сервером	Только HTTP/1.1
416	Requested Range Not Satisfiable	Указанный клиентом диапазон содержимого ресурса при частичной загрузке выходит за пределы длины запрашиваемого URI	Только HTTP/1.1
417	Expectation Failed	Запрошенные условия заголовка Expect не могут быть выполнены сервером	Только HTTP/1.1
418	I’m A Teapot	Сервер сообщает о том, что он не может приготовить кофе, потому что он чайник. Эта ошибка ссылается на Hyper Text Coffee Pot Control Protocol (гипертекстовый протокол кофейников) который был первоапрельской шуткой в 1998 году	
429	Too Many Requests	Клиент превысил ограничение на количество запросов в единицу времени, установленное сервером	HTTP/1.1
499	Client Closed Connection	Клиент не дождался ответа от сервера и закрыл соединение	Специфичный для NGINX код
Ошибки сервера			
500	Internal Server Error	Внутренняя ошибка сервера	HTTP/0.9 и выше
501	Not Implemented	Метод запроса не поддерживается сервером (единственные методы, которые обязаны поддерживаться сервером - GET, HEAD)	HTTP/0.9 и выше
502	Bad Gateway	Сервер, работающий в качестве шлюза, получил недопустимый или неправильный ответ	HTTP/0.9 и выше
503	Service Unavailable	Сервер недоступен или не готов обрабатывать запрос	HTTP/0.9 и выше
504	Gateway Timeout	Сервер, работающий в качестве шлюза, не может получить ответ вовремя	Только HTTP/1.1
505	HTTP Version Not Supported	Версия протокола HTTP, используемая клиентом, не поддерживается	Только HTTP/1.1

ЗАГОЛОВКИ ПРОТОКОЛА HTTP

С развитием протокола HTTP, чтобы добавить функциональности, решить возникающие на стороне клиента или сервера проблемы, наилучшим образом согласовать их работу во время обмена сообщениями, в запросы и ответы после стартовой строки были добавлены заголовки в определенном формате (Название заголовка : Значение заголовка), большинство заголовков внесены в стандарт протокола, однако протокол не запрещает реализацию дополнительных заголовков, названия и работа с которыми согласованы между конкретными клиентом и сервером.

Все HTTP-заголовки можно условно сгруппированы по их контексту использования:

- Общие заголовки, они применимы и к запросам и к ответам, не описывают никаким образом данные, передаваемые в теле сообщения
- Заголовки запроса, специфичны только для HTTP-запросов
- Заголовки ответа, специфичны только для HTTP-ответов
- Заголовки сущности, описывают характеристики ресурса, находящегося по URI

Так же HTTP-заголовки могут быть сгруппированы по контексту их функционала.

ПОДРОБНЕЕ О НЕКОТОРЫХ ВАЖНЫХ ЗАГОЛОВКАХ ПРОТОКОЛА HTTP

ЗАГОЛОВКИ АУТЕНТИФИКАЦИИ:

WWW-Authenticate - заголовок ответа, определяет параметры аутентификации клиента, которую необходимо пройти для получения доступа к запрашиваемому ресурсу

Authorization - заголовок запроса, содержащий в определенном формате данные для аутентификации клиента на сервере, формат определяется сервером в заголовке, описанном выше

ЗАГОЛОВКИ КЕШИРОВАНИЯ:

Cache-Control - заголовок запроса и ответа, определяет параметры кеширующих механизмов, существует несколько возможных значений заголовка, например, Cache-Control: no-cache отключает кеширование

Expires - заголовок ответа, описывающий дату, после которой данные ресурса необходимо запросить заново

Pragma - заголовок протокола HTTP/1.0, аналогичен заголовку Cache-Control, существует для обеспечения обратной совместимости

УСЛОВНЫЕ ЗАГОЛОВКИ:

Last-Modified - заголовок ответа, содержащий дату последнего обновления запрашиваемого ресурса, используется клиентами для сравнения версии кешированных данных и ресурса на сервере

ETag - заголовок ответа, содержащий уникальную строку, указывающую версию запрошенного клиентом ресурса, так же используется для сравнения версии кешированных данных и ресурса на сервере

If-Match, If-None-Match - заголовки запроса, добавляющие в него условия на значение ETag, что указанный в запросе метод будет выполнен только в результате выполнения условия

If-Modified-Since, If-Unmodified-Since - заголовки запроса, аналогично описанным выше, добавляющие условия на значение Last-Modified

ЗАГОЛОВКИ УПРАВЛЕНИЯ СОЕДИНЕНИЕМ:

Connection - общий заголовок, указывающий на то, должно ли соединение между клиентом и сервером закрываться после завершения передачи данных, для протокола HTTP/1.0 умолчанием является Connection : close - закрытие соединения сразу по окончании передачи

Keep-Alive - общий заголовок, задающий таймаут в секундах для поддержания соединения между клиентом и сервером открытым после завершения передачи или количество запросов, которые могут быть отправлены по этому соединению до его закрытия

ЗАГОЛОВКИ СОГЛАСОВАНИЯ КОНТЕНТА:

Accept - заголовок запроса, сообщающий серверу, какие контент какого типа клиент может воспринять

Accept-Charset - заголовок запроса, сообщающий серверу, ответ в какой кодировке клиент может воспринять

Accept-Encoding - заголовок запроса, сообщающий серверу, ответ с использованием какого алгоритма кодирования клиент может воспринять, обычно этот заголовок содержит информацию о поддерживаемых клиентом алгоритмах сжатия

ЗАГОЛОВКИ КОНТРОЛЯ ПЕРЕДАЧИ:

Expect - заголовок запроса, описывающий какой статус ответа клиент ожидает в ответе сервера для успешного завершения запроса, например, когда клиент собирается передать запрос с большим телом по частям

ЗАГОЛОВКИ COOKIES:

Этим заголовкам будет еще посвящён раздел в материале по видам и схемам аутентификации. Особенность этой группы заголовков состоит в том, что любой клиент должен передавать содержимое этих заголовков во всех последующих запросах к серверу в том же виде, в котором они пришли в ответе от сервера. Очень часто эти заголовки используются для организации сессии, аутентификации клиента, таргетирования, метрик.

Cookie - заголовок запроса, содержащий строку с данными, пришедшими в заголовке ответа Set-Cookie

Set-Cookie - заголовок ответа, содержащий специфичные для сервера или приложения данные, которые клиент должен передавать во всех последующих запросах

ЗАГОЛОВКИ ЗАГРУЗКИ КОНТЕНТА:

Content-Disposition - заголовок ответа, сообщающий клиенту, что необходимо начать загрузку данных вместо интерпретации и отображения, в браузерах обычно появляется диалог «Сохранить как» при получении подобного заголовка ответа

ЗАГОЛОВКИ ИНФОРМАЦИИ О ТЕЛЕ:

Content-Length - заголовок сущности, описывающий длину содержимого тела в байтах

Content-Type - заголовок сущности, описывающий MIME-тип (медиа-тип) содержимого тела

Content-Encoding - заголовок сущности, описывающий в основном алгоритм сжатия содержимого тела

Content-Location - заголовок сущности, описывающий другой адрес, по которому располагается запрашиваемый ресурс

ЗАГОЛОВКИ РЕДИРЕКТА:

Location - заголовок ответа, содержащий URL (Uniform Resource Locator, унифицированный указатель ресурса), по которому клиент должен перейти для получения данных

ЗАГОЛОВКИ КОНТЕКСТА ЗАПРОСА:

From - заголовок запроса, содержащий email-адрес «хозяина» клиента, в основном применяется разработчиками краулеров, чтобы была возможность связаться с разработчиком или поддержкой в случае обнаружения проблем

Host - заголовок запроса, содержащий доменное имя (имя хоста), по которому идет обращения, например, www.google.ru, опционально может содержать номер TCP-порта

Referer - заголовок запроса, содержащий адрес страницы, с которой клиент сделал текущий запрос, например, при переходе по ссылки с одной страницы на другую

User-Agent - заголовок запроса, содержащий строку, описывающую клиента: название и версия браузера, код производителя, название и версия движка, операционной системы

ЗАГОЛОВКИ КОНТЕКСТА ОТВЕТА:

Allow - заголовок ответа, содержащий список поддерживаемых сервером методов

Server - заголовок ответа, содержащий информацию о названии и версии HTTP-сервера, обслуживающего запрос

ЗАГОЛОВКИ ДИАПАЗОНОВ:

Accept-Range - заголовок ответа, указывающий поддерживает ли сервер передачу по частям, если поддерживает, то указывается в каких единицах изменения может быть указан диапазон

Range - заголовок запроса, указывающий, какой диапазон данных ответа клиент хочет получить

If-Range - заголовок запроса, создающий условия предотвращения загрузки диапазонов разных версий ответа, например, если в процессе загрузки данных клиентом на сервере произошло обновление

Content-Range - заголовок ответа, указывающий какому диапазону данных ответа принадлежат данные в теле

ДРУГИЕ ЗАГОЛОВКИ:

Date - общий заголовок, содержащий дату и время сообщения

Retry-After - заголовок ответа, сообщающий клиенту, когда сделать повторную попытку запроса в случае ошибки

Upgrade - общий заголовок, сообщающий о необходимости перевести текущее соединение клиента с сервером на другую версию протокола HTTP

ЗАГОЛОВКИ X-* ПРОТОКОЛА HTTP

Как уже упоминалось выше, кроме заголовков, вошедших в стандарты и спецификации версий протоколов HTTP, сервер и клиент могут ввести дополнительные заголовки, работа с которыми будет согласована между ними. Ранее к таким заголовкам было принято относить заголовки с именами, начинающимися с «X-». Однако со временем многие такие дополнительные заголовки, разработанные и внедренные сервисами, работающими по протоколу HTTP, были внесены в стандарты и спецификации протокола.

Одной из популярных групп X-* заголовков являются заголовки, добавляемые к запросам клиента, проходящим через прокси-серверы или балансировщики нагрузки.

Прокси-сервер - это промежуточный сервер между клиентом и целевым вэб-сервером. Прокси-серверы могут как обеспечивать доступ клиента к внутренним серверным ресурсам, находящимся в отличной от клиента сети или в защищенной сетевой зоне, могут контролировать и балансировать доступ клиентов к серверным ресурсам, анонимизировать запросы клиентов, сжимать данные запросов и ответов для экономии трафика, кешировать часть ответов от сервера для обеспечения более быстрого доступа клиентов к данным. Очень часто клиенты даже не подозревают о прохождении их запросов через прокси-серверы.

Проходя через прокси сервер некоторые заголовки HTTP-запроса могут быть добавлены или изменены в зависимости от задач и целей прокси-сервера, большинство заголовков при прохождении через прокси-сервер остаются в неизменном виде. Например, для клиентов, не поддерживающих сжатие, запросы к прокси-серверу и ответы от целевого вэб-сервера, передаваемые клиенту, будут проходить без сжатия, однако запросы и ответы между прокси-сервером и целевым сервером будут сжаты, что позволит обеспечить совместимость. Или при прохождении запросов через прокси-сервер, обеспечивающий балансировку, в запросы, отправляемые целевому серверу могут добавляться специальные заголовки, например, соответствующие региону клиента, позволяющие выбрать наиболее подходящий целевой сервер из группы серверов. Или просто передать информацию о клиенте, которую сервер не сможет получить из TCP-соединения с прокси-сервером, например, внешний IP-адрес клиента.

ЗАГОЛОВКИ ПРОКСИ-СЕРВЕРОВ:

X-Forwarded-For, X-Real-IP - заголовок запроса, содержащий исходный внешний IP-адрес клиента

X-Forwarded-Proto, X-Real-Proto - заголовок зарпоса, содержащий протокол (HTTP, HTTPS), который клиент использовал в исходном запросе

X-Forwarded-Host - заголовок запроса, содержащий доменное имя хоста, указанного клиентом в запросе в заголовке Host

НЕКОТОРЫЕ ДРУГИЕ ЗАГОЛОВКИ:

X-Powered-By - заголовок ответа, который может содержать некоторые данные окружения и параметров сервера, которые могут быть полезны клиентам, обычно заголовке передается не вся информация, чтобы избежать попыток использования уязвимостей конкретных версий софта

X-Frame-Options - заголовок запроса, указывающий, позволено ли браузеру отображать содержимое ответа во фрейме или нет

X-XSS-Protection - заголовок ответа, не реализованный или отключенный частью браузеров, включающий защиту от cross-site scripting

X-DNS-Prefetch-Control - заголовок ответа, включающий опцию в поддерживающих ее браузерах предварительного разрешения доменных имен по ссылкам из загружаемого документа еще до запроса этих ссылок, чтобы ускорить работу в последующих запросах

X-UA-Compatible - заголовок ответа, применимый к Internet Explorer, указывающий на то, в каком из возможных режимов должна отображаться страница

X-Robots-Tag - заголовок ответа, содержащий информацию о том, как роботы и краулеры должны индексировать страницу и отображать данные в публичных поисковых ресурсах

X-Content-Type-Options - заголовок ответа, отключающий в браузерах автоматическое определение MIME-типа (медиатипа) содержимого ответа и заставляющий использовать данные из заголовка Content-Type