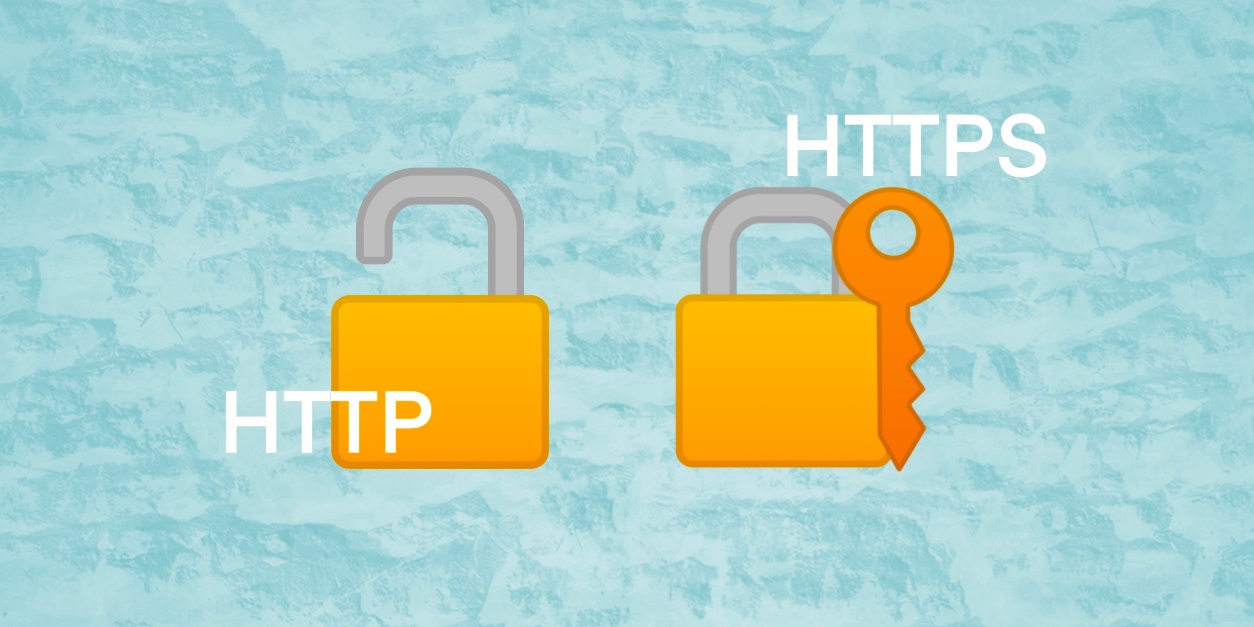
Что такое HTTP и HTTPS?

Рассказываем о популярных протоколах, на которых построена всемирная паутина, а также отвечаем на вопросы новичков. Без понимания основ перейти к изучению более сложных тем не получится, а значит самое время восполнить пробелы в знаниях.

11[Обсудить](javascript:void;)



Что такое HTTP?

**HTTP** – широко распространённый протокол передачи данных, изначально предназначенный для гипертекстовых документов, то есть документов, которые могут содержать ссылки, позволяющие организовать переход к другим документам.

HTTP 1.1

Аббревиатура **HTTP** расшифровывается как **HyperText Transfer Protocol**, *«протокол передачи гипертекста»*. В соответствии со спецификацией [**OSI**](http://en.wikipedia.org/wiki/OSI_model), **HTTP** является протоколом прикладного (верхнего, 7-го) уровня. Актуальная на данный момент версия протокола, HTTP 1.1, описана в спецификации [RFC 2616](http://tools.ietf.org/html/rfc2616).

Также **HTTP** часто используется как протокол передачи информации для других протоколов прикладного уровня, таких как **SOAP**, **XML-RPC** и **WebDAV**. В таком случае говорят, что протокол **HTTP** используется как «*транспорт*». **API** многих программных продуктов также подразумевает использование **HTTP** для передачи данных – сами данные при этом могут иметь любой формат, например, **XML** или **JSON**. Как правило, передача данных по протоколу **HTTP** осуществляется через **TCP/IP**-соединения. Серверное программное обеспечение при этом обычно использует **TCP**-порт 80 (и, если порт не указан явно, то обычно клиентское программное обеспечение по умолчанию использует именно **80**-й порт для открываемых **HTTP**-соединений), хотя может использовать и любой другой.

Как отправить HTTP-запрос?

Чтобы сформировать **HTTP**-запрос, необходимо составить стартовую строку, а также задать по крайней мере один заголовок – это заголовок **Host**, который является обязательным, и должен присутствовать в каждом запросе. Дело в том, что преобразование доменного имени в **IP**-адрес осуществляется на стороне клиента, и, соответственно, когда вы открываете **TCP**-соединение, то удалённый сервер не обладает никакой информацией о том, какой именно адрес использовался для соединения. Однако фактически сетевое соединение во всех случаях открывается с узлом**212.24.43.44**, и даже если первоначально при открытии соединения был задан не этот **IP**-адрес, а какое-либо доменное имя, то сервер об этом никак не информируется — и именно поэтому этот адрес необходимо передать в заголовке **Host**.

Начало формы

Конец формы

[Хочешь попасть в компанию мечты или просто стать популярным в IT-сообществе?](javascript:void%200;" \t ")

[Поделись своим опытом и знаниями в нашей лаборатории онлайн-курсов, получив % с выручки курса! Поможем со съемками, продюсированием, программой курса. С тебя — экспертиза.](javascript:void%200;" \t ")

[Подробнее](javascript:void%200;" \t ")

**Метод** (*в англоязычной тематической литературе используется слово method, а также иногда слово verb – «глагол»*) представляет собой последовательность из любых символов, кроме управляющих и разделителей, и определяет операцию, которую нужно осуществить с указанным ресурсом. Спецификация **HTTP 1.1** не ограничивает количество разных методов, которые могут быть использованы, однако в целях соответствия общим стандартам и сохранения совместимости с максимально широким спектром программного обеспечения как правило используются лишь некоторые, наиболее стандартные методы, смысл которых однозначно раскрыт в спецификации протокола.

* GET – получение ресурса;
* POST – создание ресурса;
* PUT – обновление ресурса;
* DELETE – удаление ресурса.

*Обратите внимание на тот факт, что спецификация HTTP не обязывает сервер понимать все методы (которых на самом деле гораздо больше, чем 4) – обязателен только****GET****, а также не указывает серверу, что он должен делать при получении запроса с тем или иным методом. Это значит, что сервер в ответ на запрос****DELETE****/index.html****HTTP/1.1****не обязан удалять страницу index.html на сервере, так же как на запрос GET /index.html HTTP/1.1 не обязан возвращать вам страницу index.html, он может ее удалять, например :)*

**URI** (*Uniform Resource Identifier, унифицированный идентификатор ресурса*) – путь до конкретного ресурса (например, документа), над которым необходимо осуществить операцию (например, в случае использования метода **GET** подразумевается получение ресурса). Некоторые запросы могут не относиться к какому-либо ресурсу, в этом случае вместо **URI** в стартовую строку может быть добавлена звёздочка (астериск, символ «\*»). Например, это может быть запрос, который относится к самому веб-серверу, а не какому-либо конкретному ресурсу.

Как прочитать отчет HTTP-запроса?

Версия протокола здесь задаётся так же, как в запросе.

**Код состояния**(*Status Code*) – три цифры (*первая из которых указывает на класс состояния*), которые определяют результат совершения запроса. Например, в случае, если был использован метод **GET**, и сервер предоставляет ресурс с указанным идентификатором, то такое состояние задаётся с помощью кода **200**. Если сервер сообщает о том, что такого ресурса не существует – **404**. Если сервер сообщает о том, что не может предоставить доступ к данному ресурсу по причине отсутствия необходимых привилегий у клиента, то используется код 403. Спецификация**HTTP 1.1** определяет 40 различных кодов **HTTP**, а также допускается расширение протокола и использование дополнительных кодов состояний.

**Пояснение к коду состояния** (*Reason Phrase*) – текстовое (но не включающее символы **CR** и LF**)** пояснение к коду ответа, предназначено для упрощения чтения ответа человеком. Пояснение может не учитываться клиентским программным обеспечением, а также может отличаться от стандартного в некоторых реализациях серверного ПО.

**Ответ:**

http-request.example

HTTP/1.1 301

server: nginx/1.19.1

date: Wed, 17 Mar 2021 10:54:34 GMT

content-type: text/html

content-length: 169

location: http://demo.yourapi.ru/

strict-transport-security: max-age=15724800; includeSubDomains

<html>

<head><title>301 Moved Permanently</title></head>

<body>

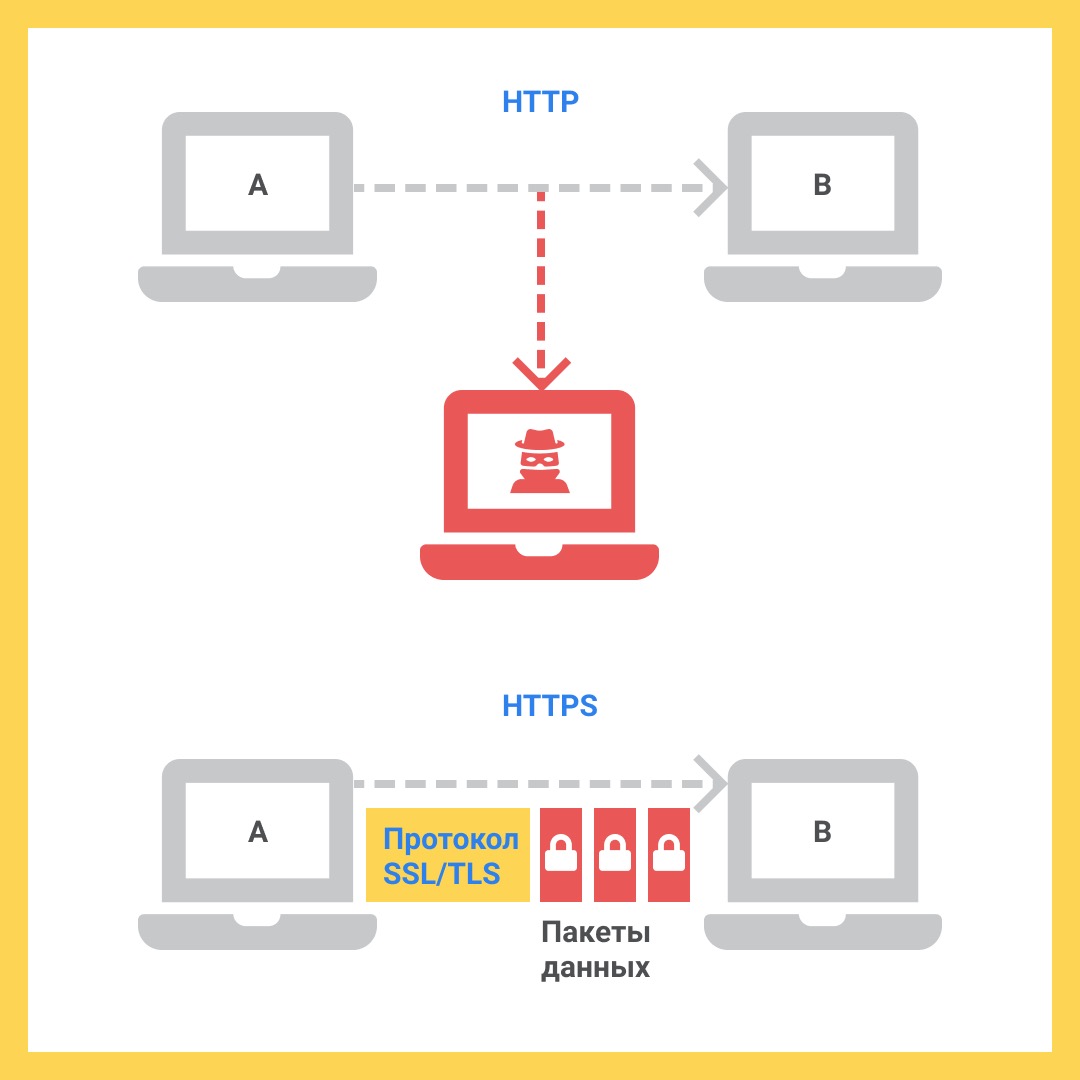
<center><h1>301 Moved Permanently</h1></center>

<hr><center>nginx/1.19.6</center>

</body>

</html>

Безопасность HTTP-запроса

Немного о безопасности :-) Без нее никуда!

Сам по себе протокол **HTTP** не предполагает использование шифрования для передачи информации. Тем не менее, для **HTTP** есть распространённое расширение, которое реализует упаковку передаваемых данных в криптографический протокол **SSL** или **TLS**.

Название этого расширения – **HTTPS** (*HyperText Transfer Protocol Secure*). Для **HTTPS**-соединений обычно используется **TCP**-порт **443**. **HTTPS** широко используется для защиты информации от перехвата, а также, как правило, обеспечивает защиту от атак вида *man-in-the-middle* — в том случае, если сертификат проверяется на клиенте, и при этом приватный ключ сертификата не был скомпрометирован, пользователь не подтверждал использование неподписанного сертификата, и на компьютере пользователя не были внедрены сертификаты центра сертификации злоумышленника.

На данный момент **HTTPS** поддерживается всеми популярными веб-браузерами.

Что такое HTTPS?

**HTTPS** (*Hypertext Transport Protocol Secure*) – это протокол, который обеспечивает конфиденциальность обмена данными между сайтом и пользовательским устройством. Безопасность информации обеспечивается за счет использования криптографических протоколов **SSL/TLS**, имеющих 3 уровня защиты:

* Шифрование данных позволяет избежать их перехвата;
* Сохранность данных – любое изменение данных фиксируется;
* Аутентификация защищает от перенаправления пользователя.

Этапы взаимодействия с сертификатом

В каких случаях необходим сертификат HTTPS?

Обязательное использование защищенного протокола передачи данных требует вся информация, касающаяся проведения платежей в интернете: оплата товаров в интернет-магазинах любым способом (*индивидуальная платежная карта, онлайн системы платежей и пр.*), оплата услуг через интернет-банкинг, совершение платежей в онлайн сервисах (*казино, online-курсы и т.п*.) и многое другое.

Использовать протокол **HTTPS** рекомендуется также на сайтах, которые для доступа к определенному контенту запрашивают личные данные пользователей, например, номер паспорта – такие данные необходимо защищать от перехвата злоумышленниками.

Если на вашем сайте используется что-либо похожее, то вам стоит серьезно задуматься над переходом на **HTTPS**. Поэтому далее мы рассмотрим, что для этого необходимо.

Что нужно для перехода сайта на HTTPS?

Работа протокола **HTTPS** основана на том, что компьютер пользователя и сервер выбирают общий секретный ключ, с помощью которого и происходит шифрование передаваемой информации. Этот ключ уникальный и генерируется для каждого сеанса. Считается, что его подделать невозможно, так как в нем содержится более 100 символов. Во избежание перехвата данных третьим лицом используется цифровой сертификат – это электронный документ, который идентифицирует сервер. Каждый владелец сайта (сервера) для установки защищенного соединения с пользователем должен иметь такой сертификат.

В этом электронном документе указываются данные владельца и подпись. С помощью сертификата вы подтверждаете, что:

* лицо, которому он выдан, действительно существует;
* оно является владельцем сервера (сайта), который указан в сертификате.

Первое, что делает браузер при установке соединения по протоколу **HTTPS**– проверку подлинности сертификата, и только в случае успешного ответа начинается обмен данными.

Сертификатов существует несколько видов в зависимости от следующих факторов:

* необходимого уровня безопасности;
* количества доменных имен и поддоменов;
* количества владельцев.

Это уже тема для отдельной статьи. Выдают их специализированные центры сертификации на возмездной основе и на определенный период, поэтому важно не забывать продлевать действие сертификата.

\*\*\*

*Для начинающих пользователей также интересна тема про инструменты, которые необходимы для тестирования своего API – я готов продолжить серию публикаций.*

*P.S. Это моя первая статья на ресурсе*[proglib.io](https://proglib.io/)*:-)*