УДК 004.05

ПРОТОКОЛ HYPERTEXT TRANSFER PROTOCOL SECURE. БЕЗОПАСНОСТЬ WEB-СЕРВИСОВ

Д.С. ТАТАРИН
(Представлено: канд. тех. наук, доц. И.Б. БУРАЧЁНОК)

Рассмотрены особенности протокола HyperText Transfer Protocol Secure (HTTPS) и проблемы, с которыми можно столкнуться при его использовании. Сделаны выводы о необходимых мерах для обеспечения полной безопасности при использовании протокола HTTPS.

При вводе названия сайта все привыкли видеть вначале ссылки http:// или HyperText Transfer Protocol (HTTP) – «протокол передачи гипертекста» – стандартный протокол передачи данных от сервера, на котором находится сайт, к пользователю. Однако, не смотря на всю его популярность, все больше сайтов предпочитают использовать более продвинутый протокол – HTTPS, так как он защищает передаваемые данные от перехвата злоумышленниками путем их шифрования. В представленной статье рассмотрим этот протокол более подробно: как он работает, кому рекомендуется использовать и что нужно, чтобы его подключить к сайту.

HTTPS – это протокол, который обеспечивает конфиденциальность обмена данными между сайтом и пользовательским устройством. Безопасность информации обеспечивается за счет использования криптографических протоколов SSL (Cure Sockets Layer) – слой защищенных сокетов и его предшественника TLS (Transport Layer Security) – протокол защиты транспортного уровня, далее SSL/TLS, имеющих три уровня защиты:

- шифрование данных – позволяет избежать их перехвата;
- сохранность данных – любое изменение данных фиксируется;
- аутентификация – защищает от перенаправления пользователя.

SSL можно сравнить с «фантиком», в который заворачивают данные HTTP, чтобы скрыть их от посторонних. Протокол SSL/TLS помогает двум незнакомым друг с другом пользователям Интернета установить защищенное соединение через обычный, незащищенный канал. С помощью математических алгоритмов оба пользователя – клиент и сервер – договариваются о секретном ключе, не передавая его напрямую через соединение. Даже если кто-то сумеет подключиться к соединению и перехватить все передаваемые данные, расшифровать их ему не удастся.

Протокол SSL использует многослойную среду: с одной стороны, от него находится протокол программы-клиента, например, (Internet Message Access Protocol (IMAP) – протокол прикладного уровня для доступа к электронной почте, File Transfer Protocol (FTP) – протокол передачи файлов по сети и HTTP), а с другой – транспортный TCP/IP. Название TCP/IP происходит из двух важнейших протоколов семейства –Transmission Control Protocol (TCP)/Internet Protocol (IP) или протокол управления передачей/интернет протокол. Для SSL шифрования используются симметричные и ассиметричные ключи, полученные с помощью различных математических моделей [1].

Далее проведем сравнительный анализ использования протоколов HTTPS и HTTP.

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. Атаку man-in-the-middle можно увидеть на рисунке.

На данный момент HTTPS поддерживается всеми популярными веб-браузерами.

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.

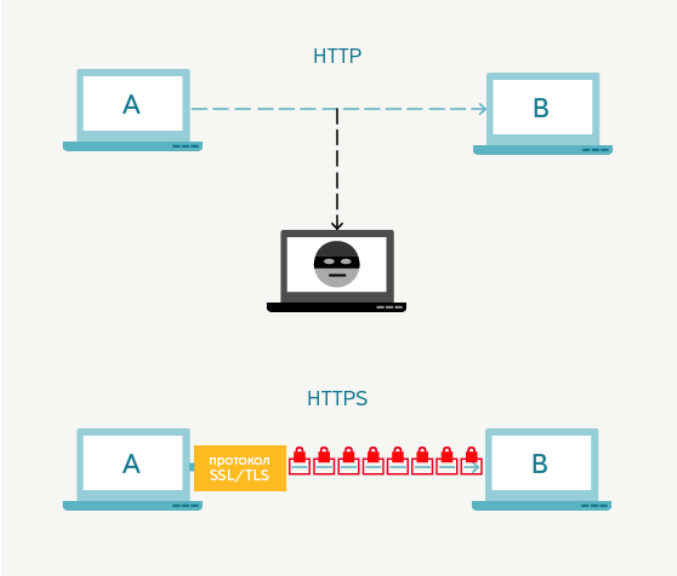

Рисунок. – Сравнение HTTP и HTTPS

Работа протокола HTTPS основана на том, что компьютер пользователя и сервер выбирают общий секретный ключ, с помощью которого и происходит шифрование передаваемой информации. Это 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 (ЭД), который идентифицирует сервер. Каждый владелец сайта (сервера) для установки защищенного соединения с пользователем должен иметь такой сертификат. В этом ЭД 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, это проверку подлинности сертификата, и только в случае успешного ответа начинается обмен данными.

Подробнее остановимся на проблемах, с которыми можно столкнуться, используя HTTP.

1. Совместное использование HTTP и HTTPS. Когда сайты используют смешанный функционал HTTP и HTTPS, это потенциально приводит к информационной угрозе для пользователя. Например, если основные страницы некоторого сайта загружаются, используя HTTPS, а Cascading Style Sheets (CSS) и JavaScript загружаются по HTTP, то злоумышленник в момент загрузки последних может подгрузить свой код и получить данные HTML-страницы. Многие сайты, несмотря на такие уязвимости, загружают контент через сторонние сервисы, которые не поддерживают HTTPS. Механизм HSTS позволяет предотвратить подобные уязвимости, заставляя принудительно использовать HTTPS соединение даже там, где по умолчанию используется HTTP [2].

2. Атаки с использованием анализа трафика. В HTTPS были обнаружены уязвимости, связанные с анализом трафика. Атака с анализом трафика – это тип атаки, при которой выводятся свойства защищённых данных канала путём измерения размера трафика и времени передачи сообщений в нем. Анализ трафика возможен, поскольку шифрование SSL/TLS изменяет содержимое трафика, но оказывает минимальное влияние на размер и время прохождения трафика. В мае 2010 года исследователи из Microsoft Research и Университета Индианы обнаружили, что подробные конфиденциальные пользовательские данные могут быть получены из второстепенных данных, таких, например, как размеры пакетов. Анализатор трафика смог заполучить историю болезней, данные об использовавшихся медикаментах и проведённых операциях пользователя, данные о семейном доходе и пр. Все это было произведено несмотря на использование HTTPS в нескольких современных веб-приложениях в сфере здравоохранения, налогообложения и др. [3].

3. Человек посередине HTTPS. При атаке «человек посередине» используется то, что сервер HTTPS отправляет сертификат с открытым ключом в браузер. Если этот сертификат не заслуживает доверия, то канал передачи будет уязвимым к атаке злоумышленника. Такая атака заменяет оригинальный сертификат, удостоверяющий HTTPS-сервер, модифицированным сертификатом. Атака проходит успешно, если пользователь пренебрегает двойной проверкой сертификата, когда браузер отправляет

предупреждение. Это особенно распространено среди пользователей, которые часто сталкиваются с само заверенными сертификатами при доступе к сайтам внутри сети частных организаций.

В результате исследования можно сделать вывод о том, что протокол HTTPS достаточно гибок и удобен в использовании, он обеспечивает конфиденциальность обмена данными между сайтом и пользовательским устройством. Благодаря протоколу HTTPS стало возможным сохранять в тайне конфиденциальную информацию. Для организации общения клиента с сервером HTTPS использует для шифрования протокола SSL/TLS с достаточно высоким уровнем безопасности.

ЛИТЕРАТУРА

1. Как работает SSL? Принцип работы https соединения [Электронный ресурс]. – Режим доступа: https://www.ipipe.ru/info/kak-rabotaet-ssl-sertificat.html. – Дата доступа: 12.09.2018.

2. How to Deploy HTTPS Correctly [Электронный ресурс]. – Режим доступа: https://www.eff.org/https-everywhere/deploying-https. – Дата доступа: 13.09.2018.

3. Side-Channel Leaks in Web Applications: a Reality Today, a Challenge Tomorrow [Электронный ресурс]. – Режим доступа: https://www.microsoft.com/en-us/research/publication/side-channel-leaks-in-web-applications-a-reality-to-ay-a-challenge-tomorrow – Дата доступа: 12.09.2018.