Secure Sockets Layer 3.0: обзор и анализ.

Студента 118 группы Иванова Петра Сергеевича

15 апреля 2005

Эссе по курсу "Защита информации", кафедра радиотехники, Московский физико-технический институт (ГУ МФТИ), http://www.re.mipt.ru/infsec

Из-за постоянно возрастающих объемов все более важных данных, безопасность их передачи через Интернет становится крайне важной. Сегодня каждый пользователь общей сети регулярно посылает различные типы данных, от электронной почты до номера кредитной карты, и он хотел бы, чтобы при передаче по общим сетям его данные были бы защищены. Для этого был приспособлен протокол Secure Sockets Layer, предназначенный для защиты данных при их передаче, что включает в себя все сетевые службы, использующие TCP/IP, с целью поддерживать распространенные прикладные задачи коммуникации клиента и сервера.

Протокол SSL был изначально разработан фирмой Netscape, чтобы обеспечить безопасность передачи данных, передаваемых через пользовательские средства HTTP, LDAP и POP3. SSL разработан так, чтобы использовать TCP как средство коммуникации, осуществляя надежную и аутентифицированную связь между двумя точками в сети (например, между клиентом службы и ее сервером). Несмотря на то, что SSL может быть использован для защиты данных при передаче данных в любой сетевой службе, он используется в основном в HTTP приложениях типа сервер-клиент. Сегодня почти каждый сетевой THHP сервер может осуществлять связь с использованием SSL, ведь Internet Explorer и Netscape Navigator обладают средствами клиентской части SSL.

Цели и архитектура SSL

Главные задачи SSL следующие:

- Аутентификация клиента и сервера друг для друга: протокол SSL поддерживает использование стандартных технологий криптографии с общим и личным ключом с целью аутентификации взаимодействующих сторон друг для друга. Хотя более распространен случай аутентификации сервера для клиента, SSL может использовать те же методы для аутентификации клиента.
- Обеспечение целостности данных: во время сессии, данные не должны быть подвержены внешнему и внутреннему влиянию.
- Обеспечение защищенности данных: данные, находящиеся в процессе передачи между сервером и клиентом, должны быть защищены от перехвата и должны иметь возможность быть прочитанными только адресатом. Это требование обязательно как для данных, относящихся к самому протоколу (диалог безопасности во время установления связи), так и для данных приложения, посылаемых в течение самой сессии.

SSL – это, по сути, не один протокол, но скорее набор протоколов, объединенных в два слоя:

1. протокол для обеспечения защиты и целостности данных: этот слой состоит из SSL Record Protocol.
2. протоколы, разработанные для развертывания соединения, в этом слое находятся три протокола: SSL Handshake Protocol, SSL ChangeCipher SpecProtocol и SSL Alert Protocol.

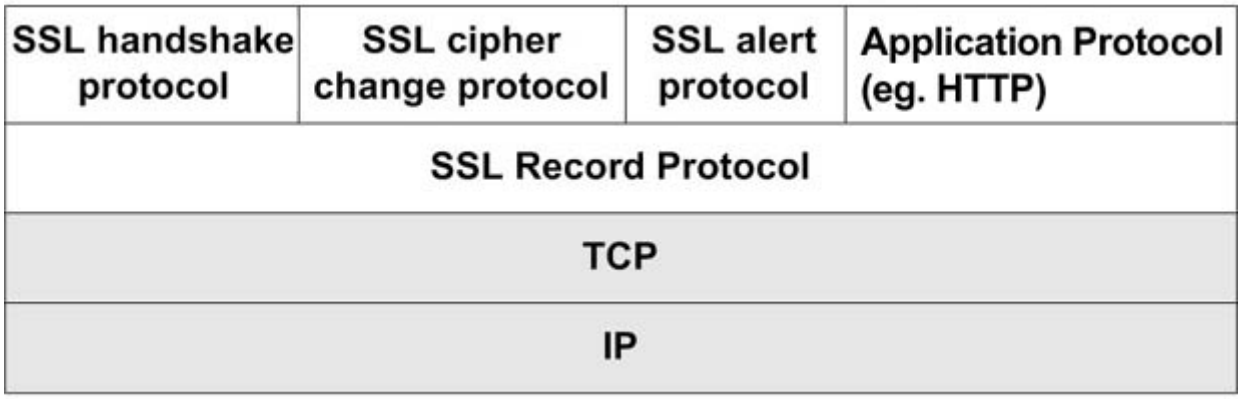

рис.1. Слои протокола SSL.

SSL использует эти протоколы для решения описанных выше задач. Протокол записи SSL отвечает за шифрование и сохранение целостности данных, остальные три протокола покрывают области менеджмента сессий, управления параметров шифрования и передачи сообщений SSL между клиентом и сервером.

Сессия SSL и установление соединения.

Для начала, имеет смысл задать описания концепций и характерных элементов:

- соединение: в терминах SSL, это соединение (peer-to-peer) двух узлов в сети.
- сессия: это взаимоотношение клиента и сервера, задающее набор параметров, таких как используемые алгоритмы, номер сессии и так далее. Сессия SSL создается так называемым Протоколом Рукопожатия (Handshake Protocol), позволяющим разделять параметры между разными соединениями сервера и клиента, и сессии используются, чтобы избежать передачи новых параметров для каждого соединения. Это означает, что одна сессия разделяется между многими SSL соединениями между сервером и клиентом. Теоретически возможно разделять многие сессии для одного соединения, но эта возможность на практике не используется. Концепции сессии и соединения в SSL включают в себя несколько параметров, которые используются для связи с поддержкой SSL между клиентом и сервером. В процессе установления соединения протоколом Handshake устанавливается метод шифрования и набор параметров, последовательно использующихся в течение сессии. Состояние сессии определяется следующими параметрами:
- идентификатор сессии: это идентификатор, созданный сервером для идентификации сессии с выбранным клиентом.
- сертификат узла: сертификат узла X.509
- метод сжатия: метод предварительного сжатия данных перед шифрованием
- Спецификация алгоритма, называемая CipherSpec: определяет алгоритм шифрования массивов данных (например, DES) и алгоритм хеширования , используемы в данной сессии (например, MD5)
- главный секрет (master secret): 48-битное число, секрет между сервером и клиентом.
- признак восстановимости: признак, говорящий о том, может ли эта сессия быть использована для создания новых соединений.

Согласно спецификации, состояние соединения SSL определяется следующими параметрами:

- Случайные данные, создаваемые клиентом и сервером для каждого соединения.
- Серверный MAC секрет записи: секретный ключ, используемый для данных, записанных сервером.

- Клиентский MAC секрет записи: секретный ключ, используемый для данных, записанных клиентом.
- Серверный ключ для записи: ключ шифра для данных, зашифрованных сервером и расшифрованных клиентом
- Клиентский ключ для записи: ключ шифра для данных, зашифрованных клиентом и расшифрованных сервером
- Номер последовательности: номер последовательности, хранимый сервером отдельно для отправленных и полученных сообщений в течение сессии

Аббревиатура MAC – это Message Authentication Code, Код Аутентификации Сообщений, используемый для передачи данных в течение сессии.

Протокол Записи SSL

Протокол записи SSL включает в себя обеспечении защищенности сообщения и сохранение его целостности. Для осуществления этих функциональностей он используется на самом верхнем уровне протокола SSL. Назначение этого протокола: взять сообщение приложения, которое должно быть передано, фрагмент данных, включить их в сообщение с правильными заголовками и создать объект, называемый записью, который затем шифруется и передается средствами протокола TCP. Первым шагом при подготовке передачи данных приложения является его фрагментация, разбиение на блоки по 16кб или меньше, с дальнейшим преобразованием их в запись. Эти фрагменты данных могут быть в сжаты, хотя спецификация протокола SSL 3.0 не содержит протокола сжатия, так что в настоящее время сжатие не используется.
В этот момент начинается создание записи, когда к каждой порции данных добавляется заголовок, возможная информация для достижения требуемого размера и MAC. Заголовок, добавляемый к каждому фрагменту данных, содержит два блока информации, а именно длину записи и длину блока данных, добавленного к оригинальной информации. На следующем этапе данные сконструированной записи содержат следующие элементы:

- Основные данные
- Дополнение до требуемой структуры (набивка)
- Величина MAC

MAC отвечает за проверку целостности данных, включенных в переданную запись. Это результат хэш-функции со специфичным алгоритмом, например MD5 или SHA-1. MAC определяется как результат хэш-функции, получающей следующие данные:
MAC = hash (секретный ключ, основные данные, набивка, номер последовательности).

Секретный ключ при создании MAC – это или клиентский MAC секрет записи, или серверный MAC секрет записи, в зависимости от того, какая сторона готовит пакет. При получении сообщения вторая сторона вычистят свое значение MAC, и сравнивает его с полученным. Если они совпали, то это означает, что данные не оказались изменены при передаче. Длина вычисляемого MAC зависит от метода вычисления. Далее, данные вместе с MAC шифруются заранее заданным алгоритмом симметричного шифрования, например DES. Шифруются как данные, так и MAC. Подготовленные данные дополняются следующими полями заголовка:

- Тип содержимого: определяет, какую полезную нагрузку несет сообщение, для определения, какие протоколы более высокого уровня использовать для дальнейшей обработки данных, включенных в пакет. Возможные значения:

change_cipher_spec, alert, handshake и application_data, ссылающиеся на соответствующие протоколы.

- Главная версия (major version): устанавливает главную часть используемой версии протокола Для SSL 3.0 эта величина равна 3.
- Подверсия (minor version): устанавливает дополнительную часть используемой версии протокола Для SSL 3.0 эта величина равна 0.

С добавлением этих полей подготовка записи завершается. После этого сообщение посылается по назначению.

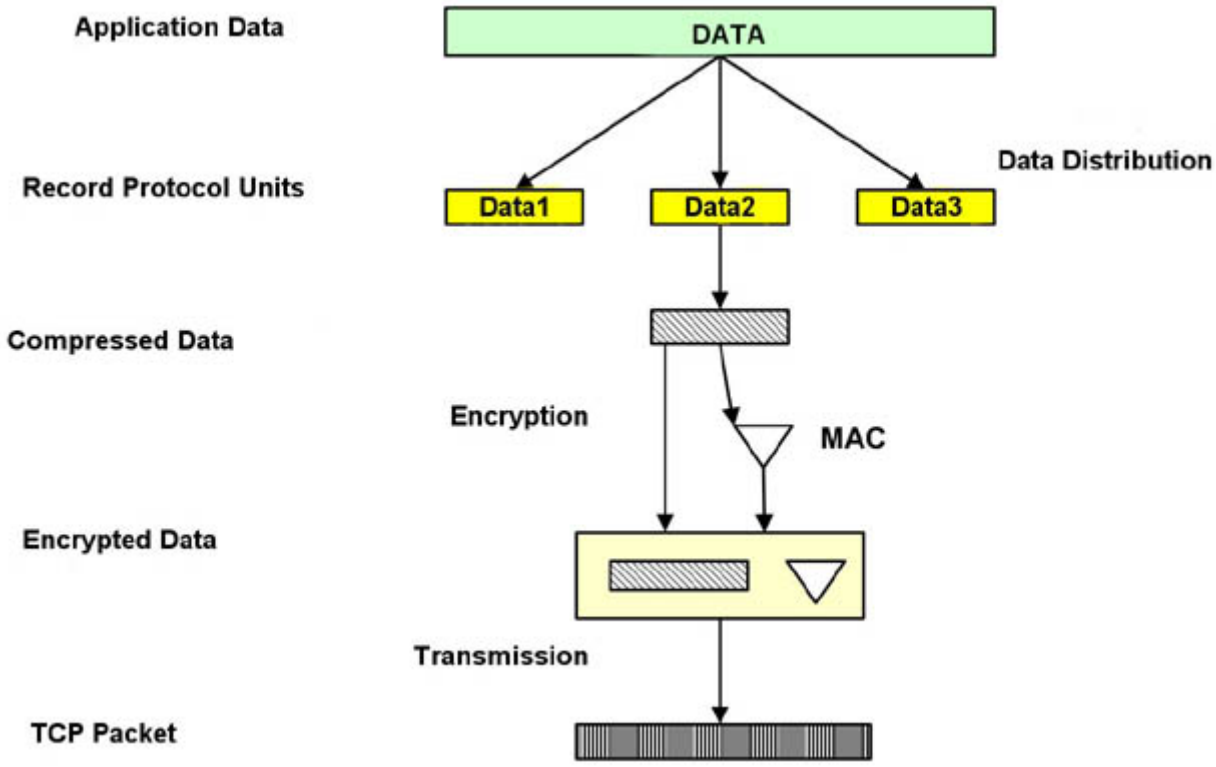

рис.2 создание пакета при протоколе записи SSL

Протокол Записи SSL используется для передачи любых данных в течение сессии, как сообщений, так и информации других протоколов SSL (например, hanshake), а также данных приложений.

Протокол сигнализации (Alert Protocol) предназначен для передачи сообщений, связанных с обменом данных и функционированием протокола. Каждое сообщение протокола Alert состоит из 2 байт. Первый байт всегда принимает значение, “warning”(1) или “fatal”(2), определяющее степень значимости сообщения. Посылка сообщение со статусом “fatal” любой из сторон немедленно приведет к закрытию сессии SSL. Следующий байт сообщения содержит код ошибки.

Протокол ChangeCipher Spec – простейший из протоколов SSL. Он состоит из единственного сообщения с величиной 1. Единственным предназначением этого сообщения является установка в фиксированное состояние незаконченной сессии, что приводит, например, к определению используемого набора протоколов. Это сообщение должно быть послано клиентом серверу и наоборот. После обмена сообщениями состояние сессии считаемся подтвержденным.

Протокол Рукопожатия (Handshake Protocol) – самая сложная часть протокола SSL. Он используется для инициации сессии между сервером и клиентом. В сообщениях этого протокола определяются различные компоненты, такие как алгоритмы и ключи, используемые для шифрования данных. С помощью этого протокола можно аутентифицировать стороны друг для друга и установить требуемые параметры сессии между ними.
Процесс обмена сообщениями между сервером и клиентом показан ниже.

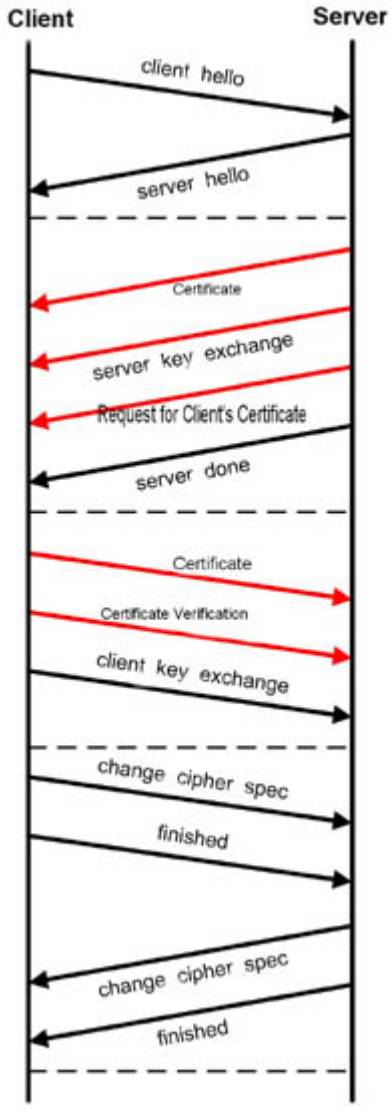

Рис.3 обмен сообщениями при инициации сессии

Он может быть разбит на 4 части, которые разделены на рисунке горизонтальными прерывистыми линиями. В течение первой стадии должно быть установлено логическое соединение между клиентом и сервером, после чего идет установка параметров соединения. Клиент посылает серверу сообщение client_hello, содержащее набор данных, таких как:

- Версия: самая высокая версия SSL, которую поддерживает сервер.
- Случайное число, состоящее из 32-битной метки времени и 28 бит случайно сгенерированных данных. Эти данные используются для защиты сессии обмена ключами между сторонами, вступающими в соединение.
- Сессионный идентификатор: номер, определяющий сессию. Ненулевой значение этого параметра говорит о том, что клиент хочет обновить параметры существующего соединения или создать новое соединение в рамках этой сессии.

Нулевое значение этого параметра означает желание клиента создать новую сессию.
- Набор шифров: список алгоритмов шифрования и методов обмена ключами, известных клиенту.

В ответ на это сервер посылает свое сообщение server_hello с тем же набором полей, содержащее следующие данные:

- Версия: наименьшая версия SSL, поддерживаемая сервером.
- Случайные данные: создаются по тем же правилам, что и у клиента, но независимо от него.
- Сессионный ID: при ненулевом клиентском поле возвращаемое значение будет таким же, при нулевом сервер пошлет значение идентификатора новой сессии.
- Набор шифров: это поле используется для отправки одного набора протоколов из тех, которые предложил клиент. Первый элемент этого поля – метод обмена ключами между сторонами. Следующий – указание алгоритмов шифрования и хеширования, которые будут использоваться в инициируемой сессии, и специфические параметры.

Набор алгоритмов шифрования и обмена ключами устанавливает три компоненты:

- Метод обмена ключами
- Алгоритм шифрования
- Функция, используемая для получения величины MAC.

Сервер начинает следующую стадию, посылая его сертификат клиенту для аутентификации. Сообщение клиенту содержит либо один сертификат, либо набор X509 сертификатов. Они нужны для аутентификации, как сервера, так и пути выдачи сертификата доверенным официальным узлом выдачи сертификатов серверу. Этот шаг может быть опущен, если метод обмена ключами не требует посылки сертификата, например, в случае анонимного метода Диффи-Хельмана. Кроме того, сервер может запросить сертификат с клиента. Конечной стадией этого этапа является сообщение server_done, показывающее, что сервер закончил отправку своих сообщений. После этого сервер ждет ответа от клиента, который должен подтвердить сертификат сервера, его данные и источник, а также прочие параметры, посланные в сообщении server_hello. Подтверждение клиента содержит:

- Проверка того, что сертификат не истек, по содержащейся в нем информации.
- Проверка того, что тело сертификата включено в список доверенных Certifying Authorities (CA, сертифицирующих служб) клиента. Если CA из сертификата сервера не найдена в списке клиента, клиент пытается проверить подпись CA. Если информация по нему не может быть получена, клиент завершает процедуру аутентификации либо возвращая сигнал об ошибке, либо запрашивая пользователя о дальнейших действиях.
- Проверка аутентичности общего ключа CA, выдавшего сертификат. Если CA включена в список доверенных CAS клиента, то ключ сравнивается с тем, который присутствует в списке.
- Проверка того, что доменное имя, используемое в сертификате, совпадает с именем сервера, показанным в сертификате сервера.

После успешного завершения всех ступеней сервер считается аутентифицированным. Если все параметры совпадают, и сертификат сервера подтвержден, клиент посылает серверу одно или несколько сообщений. Следующим будет сообщение client_key_exchange, предназначенное для доставки ключей. Содержание этого сообщения зависит от выбранного метода обмена ключами. По запросу сервера вместе с этим

сообщением может быть отправлен сертификат клиента для его подтверждения сервером. Это завершает третью фазу обмена.

Четвертая фаза предназначена для подтверждения ужа полученных сообщений и проверка правильности данных соединения. Клиент посылает сообщение change_cipher_spec и устанавливает подготовленный набор параметров алгоритмов и ключей в текущий набор. После чего клиент посылает сообщение finished, защищенное выбранными алгоритмами. Это требуется для подтверждения того, что выбранные параметры корректны. В ответ на это сервер посылает такую же последовательность сообщений. Если обе стороны успешно прочитали сообщения finished, это подтверждает правильность передачи всех данных и параметров. В этот момент сессия TCP между клиентом и сервером закрывается, но состояние сессии поддерживается, что позволяет восстановить соединение в рамках этой сессии с использованием сохраненных параметров.

Атаки на протокол SSL

1. **Раскрытие шифров**.

Протокол SSL использует множество криптографических алгоритмов, при успешной атаке на которые протокол уже не может считаться безопасным. Атака такого рода может проводиться путем записи сообщений сессии и затем подборкой ключа сессии, в случае успеха которой возможно получить всю информацию, переданную в процессе обмена. SSL пытается сделать цену подобного рода атак выше, чем выгоду от успешной атаки.

2. **Атака открытого текста**.

Такая атака может быть проведена, когда недоброжелателю известен состав пересылаемых данных. В этом случае он может составить таблицу, где зашифрованные строки известного текста являются ключами, таким образом, ключ сессии будет соответствовать одному из этих блоков данных.
Такие атаки возможны по природе SSL, например, строки типа “GET” или “SET” являются очень распространенными. Способ блокирования заключается в том, чтобы сделать объем оборудования, требуемого для проведения атаки, неприемлемо большим. Использование сессионного ключа длиной 128 бит делает составление такого словаря невозможным, а использование комбинированных технологий неприемлемо долгим.

3. **Атака отклика**

Атака отклика сводится к воспроизведению клиентской части записанной сессии злоумышленником. SSL противостоит этому с помощью кода nonce, который злоумышленник не может предугадать. Если код nonce имеет длину 128 бит, то запись множества сессий с целью угадать «правильную» становится практически невозможной.

4. **Человек посередине**

Атака man-in-the-middle заключается в том, что злоумышленник прикидывается сервером для клиента и клиентом для сервера. На SSL эта атака не проходит, так как в протокол включено использование сертификатов. В сертификате сервера содержится его общий ключ, имя и имя CA. Если злоумышленник подает клиенту ложный сертификат, то он не пройдет проверку подписи, так как секретный ключ сервера «третьему» неизвестен.

Использованные материалы:
- спецификация протокола SSL 3.0.
http://wp.netscape.com/eng/ssl3/
- Протоколы защищенных каналов: SSL, PPTP, IPSec
http://kunegin.narod.ru/ref5/ipsec/index.htm