Последовательность выполнения операций при вызове удаленной процедуры:

* процедура клиента вызывает клиентскую заглушку;
* клиентская заглушка создает сообщение и вызывает службу RPC локальной ОС;
* служба RPC пересылает сообщение службе RPC удаленной ОС;
* служба RPC удаленной ОС вызывает серверную заглушку и передает ей сообщение;
* серверная заглушка извлекает из сообщения параметры и вызывает удаленную процедуру;
* удаленная процедура исполняет код и возвращает параметры и значение возврата серверной заглушке;
* серверная заглушка формирует сообщение и вызывает службу RPC свой локальной ОС;
* служба PRC своей локальной ОС сервера пересылает сообщение службе RPC локальной ОС клиента;
* служба RPC локальной ОС клиента возвращает сообщение клиентской заглушке;
* заглушка извлекает результаты и возвращает их.

SSL

**Услуги**

SSL обеспечивает несколько услуг для данных, полученных от прикладного уровня.

**Фрагментация**

Сначала SSL делит данные на блоки 214 байтов или меньше.

**Сжатие**

Каждый фрагмент данных сжат с использованием одного из методов сжатия без потери методом, согласованным по договору между клиентом и сервером. Эта услуга является дополнительной.

**Целостность сообщений**

Чтобы сохранять целостность данных, SSL использует ключевую хэш-функцию для сохранения кода проверки подлинности (MAC).

**Конфиденциальность**

Чтобы обеспечить конфиденциальность, первоначальные данные и код проверки подлинности (MAC) зашифрованы, с использованием криптографии с симметричными ключами.

**Организация кадра**

К зашифрованной полезной нагрузке добавляется заголовок. Полезную нагрузку затем передают достоверному протоколу транспортного уровня.

***Алгоритмы смены ключей:***

NULL -> RSA -> Анонимный Диффи-Хелмана -> Кратковременный Диффи-Хелмана (RSA или DSS) -> Фиксированный Диффи-Хелмана (RSA или DSS) -> Fortezza

*NULL (пустой указатель)*

В этом методе нет никакой смены ключей. Между клиентом и сервером не установлен предварительный главный секретный код.

И клиент, и сервер должны знать значение предварительного главного секретного кода.

*RSA*

В этом методе предварительный главный секретный код – 48-байтовое случайное число, созданное клиентом, зашифрованное открытым ключом RSA сервера и передаваемое серверу. Сервер должен передать свой сертификат шифрования/дешифрования RSA.

*Анонимный протокол Диффи-Хеллмана*

Это самый простой и наиболее ненадежный метод. Предварительный главный секретный код устанавливают между клиентом и сервером, используя протокол Диффи-Хеллмана. При этом передают половину ключа в исходном тексте – это называется анонимный протоколом Диффи-Хеллмана, потому что ни одна сторона не известна другой.

*Кратковременный метод Диффи-Хеллмана*

Чтобы сорвать атаку «посредника» может быть использована кратковременная смена ключей методом Диффи-Хеллмана. Каждая сторона передает ключ Диффи-Хеллмана, подписанный своим секретным ключом. На приемной стороне должны проверить подпись, используя открытый ключ передатчика. Обмен открытыми ключами для проверки использует либо RSA-, либо DSS-сертификат цифровой подписи.

*Фиксированный метод Диффи-Хеллмана*

Все объекты группы могут подготовить фиксированные параметры (g и p). Затем каждый объект может создать фиксированную половину ключа (gx). Для дополнительной безопасности каждая отдельная половина Диффи-Хеллмана вставляется в сертификат, проверенный центром сертификации (CA). Другими словами, две стороны отдельно не обмениваются полуключами. CA передает полуключи в специальном сертификате RSA или DSS. Когда клиент должен вычислить предварительный главный секретный код, он использует свой собственный фиксированный полуключ и полуключ сервера, полученный в сертификате. Сервер делает то же самое, но в обратном порядке. Обратите внимание, что в этом методе не передаются сообщения смены ключей, а происходит только обмен сертификатами.

**Алгоритмы хэширования:**

NULL ->MD5 -> SHA-1

*NULL (пустой указатель)*

Две стороны могут отказаться использовать алгоритмы хэширования. В этом случае сообщение не заверено.

*MD5*

Две стороны могут выбрать MD5 как алгоритм хэширования. В этом случае используется алгоритм хэширования MD5 – 128-битовый.

*SHA-1*

Две стороны могут выбрать SHA как алгоритм хэширования. В этом случае используется алгоритм хэширования SHA-1 на 160 битов.

**Алгоритмы сжатия:**

Сжатие является дополнительной услугой в SSLv3. Для SSLv3 не определен алгоритм сжатия. Поэтому заданным по умолчанию методом сжатия служит NULL. Однако система может использовать любой алгоритм сжатия по выбору сторон.

**Сеансы и соединения**

SSL отличает соединения от сеанса. Сеанс – связь между клиентом и сервером. После того, как сеанс установлен, эти две стороны имеют общую информацию, такую как идентификатор сеанса, сертификат, подтверждающий подлинность каждого из них (в случае необходимости), метод сжатия (если необходимо), набор шифров и главный секретный код. Эта информация используется для того, чтобы создать ключи для сообщения, содержащего шифр установления подлинности.

Для двух объектов, чтобы начать обмен данными, установление сеанса связи необходимо, но недостаточно: они не должны создать между собой соединение. Эти два объекта обмениваются двумя случайными числами и создают, используя главный секретный ключ, ключи и параметры, необходимые для того, чтобы обмениваться сообщениями, включая установление подлинности и секретность.

Сеанс может состоять из многих соединений. Соединение между двумя сторонами может быть закончено и восстановлено в пределах одного и того же сеанса. Когда соединение закончено, эти две стороны могут также закончить сеанс, но это необязательно. Сеанс может быть приостановлен и продолжен снова.

**Четыре протокола:**

SSL содержит 4 протокола на двух уровнях. Протокол передачи записей – переносящий информацию. Он переносит на транспортный уровень сообщения от трех других протоколов, а также данные, поступающие от прикладного уровня. Сообщения из этого протокола записей – это полезная нагрузка для транспортного уровня, обычно TCP. Протокол установления соединения обеспечивает параметры безопасности для Протокола записей. Он устанавливает набор шифров и задает ключи и параметры безопасности, чтобы известить о ситуациях, отклоняющихся от нормы.

**Фаза 1: установление характеристик для обеспечения безопасности**

**ClientHello.** Сообщение содержит:

* самый высокий номер версии SSL, которую может поддерживать клиент
* 32-байтовое случайное число (от клиента), которое будет использоваться для генерации главного секретного кода (мастер кода)
* ID сеанса, который определяет список алгоритмов, поддерживаемых клиентом
* Список методов сжатия, которые клиент может поддержать

**SerserHello.** Сообщение содержит:

* Номер версии SSL. Это два номера версии:
  + Наиболее высокий номер, поддерживаемый клиентом
  + И наиболее высокий, который поддерживается сервером
* 32-байтовое случайное число (от сервера), которое будет использоваться для генерации главного секретного ключа (мастер кода)
* ID сеанса, который определяет сеанс
* Выбранный шифр из списка клиента
* Выбранный метод сжатия из списка клиента

**Фаза 2: Смена ключей сервера и установление подлинности**

В фазе 2 сервер, если необходимо, подтверждает свою подлинность. Передатчик может передать свой открытый ключ и может также запросить сертификат клиента. В конце сервер объявляет, что процесс ServerHello окончен.

**Certificate** (сертификат) – если это требуется, сервер передает сообщение certificate, чтобы подтвердить свою подлинность. Сообщение включает список сертификатов типа X.509. Если алгоритм смены ключей – анонимный Диффи-Хеллман, то сертификат не нужен.