УДК 681.327.8

П.В. КОЛЕСНИКОВ

ОБЗОР ПРОТОКОЛОВ ЗАЩИТЫ ИНФОРМАЦИИ В ОТКРЫТЫХ СЕТЯХ

В настоящее время банковская и другая информация, требующая защиты, в лучшем случае передается по корпоративным сетям или защищенным каналам связи. Построение таких сетей требует дополнительных затрат на их создание и использование, поэтому является не эффективным. Альтернатива корпоративным сетям- это объединение локальных сетей или отдельных машин, используя уже существующие сети, созданные на базе арендуемых и коммутируемых каналов связи сетей общего пользования (таких как Интернет).

Для организации такой сети необходима, там где требуется, защита сети от воздействия вирусов, злоумышленников, результатов ошибок в администрировании сети, а также от других угроз. Сейчас в мире компании-производители программного обеспечения и оборудования предпринимают усилия по разработке открытых (свободных для распространения и реализации) протоколов и стандартов в области защиты информации. Целью данной работы является исследование существующих протоколов с точки зрения обеспечения безопасности.

Эти протоколы предусматривают организацию защиты данных на различных уровнях Модели Взаимосвязи Открытых Систем (ВОС):

Таблица 1

Протоколы защиты	**Уровень ВОС**
SHTTP, S/MIME, PGP	Прикладной
SOCKS, SSL/TLS	Сеансовый
IPSec, SKIP	Сетевой
PPTP, L2TP	Канальный

Можно выделить следующие закономерности в реализации протоколов.

Чем ниже уровень ВОС, на котором организуется защита, тем она прозрачнее для приложений и незаметнее для пользователей; однако, тем меньше набор реализуемых услуг безопасности, и тем сложнее организация управления

Чем выше уровень ВОС, на котором реализуется защита, тем шире набор услуг безопасности, надежнее контроль доступа и проще конфигурирование правил доступа; однако, тем сложнее становится защита для приложений и пользователей. Применение одновременно протоколов защиты на различных уровнях модели усиливает действие каждого из протоколов

Протоколы сетевого уровня, как правило, являются фильтрами сообщений.

Протокол SOCKS описывает взаимодействие клиентов через proxy-серверы по протоколу TCP/IP Общая схема взаимодействия по протоколу SOCKS v4 сводится к следующему:

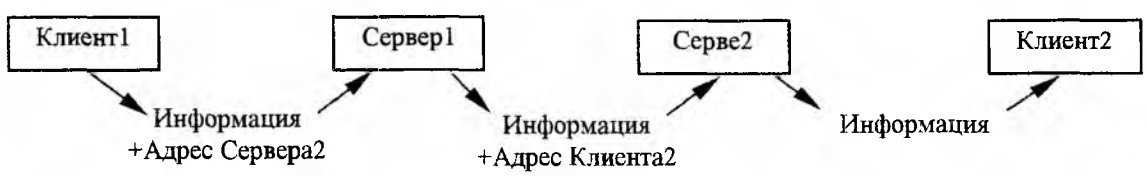


Рис. 1

Пользователь и удаленный сервер взаимодействуют друг с другом по цепочке соединений. Пользователь, желающий установить соединение с каким-либо сервером в сети, соединяется вместо этого с SOCKS-сервером и сообщает ему адрес удаленного сервера, тем самым криптоаналитик, перехвативший сообщение не может знать конечного адресата сообщения, т.к. оно передается в зашифрованном виде. Далее сам SOCKS-сервер соединяется с удаленным сервером-адресатом и передает ему сообщение. В протоколе предусмотрена аутентификация серверов и клиентов.

Одним из широко используемых протоколов в Интернет является протокол TLS. Это дальнейшая реализация протокола SSL. Главная цель TLS протокола- обеспечить сохранность и аутентичность данных при обмене сообщениями между двумя приложениями. Свойства TLS протокола:

- Создание соединения между двумя сторонами, закрытого алгоритмами шифрования.
- Способность расширения.

Протокол обеспечивает основной подход, с возможностью настроить нужные алгоритмы шифрования. Это дает возможность достичь еще две подцели. Во-первых, не нужно создавать новый протокол, который, возможно, будет обладать более слабой защитой. Во-вторых, созданная библиотека функций протокола не будет столь громоздкой. Одно из достоинств TLS протокола- это его независимость от протокола приложения. Для протоколов уровня высшего чем TLS он является совершенно прозрачным. Кроме того, TLS стандарт не определяет, как именно протоколы ведут защиту по TLS; решение как реализовать TLS протокол в конкретной ситуации и интерпретировать сертификаты аутентификации зависит от разработчика протоколов, реализованных над TLS.

<table>
<tr><td>TLS Протокол приветствия</td><td>TLS Протокол смены шифра</td><td>TLS Протокол сообщений</td><td>HTTP</td><td>FTP</td><td>SMTP</td><td>...</td></tr>
<tr><td colspan="3" rowspan="3">TLS Протокол обмена записями</td><td colspan="4">Сжатие</td></tr>
<tr><td colspan="4">Шифрование</td></tr>
<tr><td colspan="4">Аутентификация</td></tr>
<tr><td colspan="7">TCP</td></tr>
<tr><td colspan="7">IP</td></tr>
</table>

Рис. 2

Протокол разбит на два уровня: TLS протокол обмена записями и TLS—клиенты. TLS протокол обмена записями на вход получает запись, разбивает данные на блоки, сжимает их, накладывает подпись, шифрует и отправляет получателю. Информация от протокола обмена записями передается выше по стеку протоколов, либо к приложениям клиента, либо к другим составляющим протокола TLS. В последнем случае это служебная информация, необходимая для организации защищенного обмена. Существует 4 вида служебных протоколов (TLS-клиентов): протокол приветствия, сигнальный протокол, протокол смены шифра, протокол обмена данными приложения.

Протокол обмена записями находится на нижнем уровне протокола TLS и на вершине транспортного протокола TCP. TLS протокол обмена записями обеспечивает защиту соединения, которая основана на свойствах.

Соединение защищено симметричными алгоритмами шифрования (такими как DES, RC4). Ключи для этих алгоритмов генерируются уникальными для каждого соединения и основаны на общем секрете, выработанном на другом протоколе (таком как TLS приветствие). TLS протокол обмена записями может также использоваться без шифрования.

Поддерживается аутентичность соединения. Передача сообщений содержит проверку аутентичности, для которой используются алгоритмы хеш—функций такие как SHA, MD5. Протокол обмена записями может выполняться без алгоритма аутентификации, но обычно он используется.

Состояние соединения содержит информацию о том, как осуществляется обмен информацией в текущем соединении. Каждое состояние соединение определяет свои алгоритмы сжатия и шифрования. Всегда при обмене существует 4 состояния соединения: для записи, для чтения (от клиента к серверу и наоборот) и два дежурных состояния для записи и чтения.

Каждое состояние соединения включает следующую информацию:

- Состояние алгоритма шифрования: ключ шифрования; для блочных алгоритмов в режиме поточного шифрования- синхромаркер, для поточного шифра- состояние ключа.
- Закрытый ключ аутентификации
- Счетчик соединения: число размером 64 бита, которое увеличивается после каждой посланной записи и сбрасывается при установке нового текущего состояния соединения.

Протокол приветствия определяет параметры для дежурных состояний, и момент, с которого эти состояния станут активными. В первоначальном текущем состоянии, после активизации соединения, использование алгоритмов шифрования и сжатия заблокировано, пока нет договоренности о параметрах. **Протокол смены ключевых параметров** получает всего один байт на свой вход, который указывает на необходимость сменить текущие состояния для протокола обмена записями. Это сообщение требует подтверждения со стороны получателя для синхронной смены состояний.

Во время работы протокола приветствия производятся следующие действия и устанавливаются параметры защиты для соединений чтения и записи в протоколе TLS:

- Проводится аутентификация клиента и сервера с использованием X509v3 сертификатов.
- Выбирается основной алгоритм шифрования (null, «Поточное шифрование», «RC4 с ключом 40 бит», «RC4 с ключом 128 бит», «Блочные шифры в поточном режиме», «RC2 с ключом 40 бит», «DES с ключом 40 бит», «DES с ключом 54 бит», «тройной-DES с ключом 168 бит», «Idea (128 бит)», «Fortezza (96 бит)»).

Этот параметр включает размер ключа в алгоритме, сколько ключей является закрытыми, размер блока шифра, тип алгоритма (stream, block).

- Выбирается алгоритм аутентификации (null, «MD5» 128-бит, «SHA-1 160-бит»), включая размер данных, возвращаемых алгоритмом аутентификации.
- Выбирается метод сжатия и дополнительная информация, которая необходима для этого алгоритма.
- Выбирается главный секрет: 48 бит, общий для обоих сторон соединения.
- Выбирается случайное значение клиента: 32 битное значение, предоставляемое клиентом.
- Выбирается случайное значение сервера: 32 битное значение, предоставляемое сервером.

Главный секрет и случайные значения клиента и сервера (всего 102 бита) хешируются в последовательность байт, которая разбивается и присваивается следующим ключевым переменным: секрет подписи клиента, секрет подписи сервера, ключ шифрования клиента, ключ шифрования сервера, синхромаркер клиента (при использовании DES для поточного шифрования), синхромаркер сервера (при использовании DES для поточного шифрования). Таким образом информации в общем секрете должно быть достаточно для получения ключевых параметров с допустимой степенью надежности. Как видно из соответствующих значений для реализации протокола с экспортными ограничениями, для генерации ключей мы имеем всего 102 бита, что является очень маленьким числом.

Параметры клиента используются сервером, когда он принимает и обрабатывает сообщения, а параметры сервера- клиентом. После того как параметры защиты выбраны и сгенерированы ключи, могут быть назначены текущие состояния соединения.

Передаваемое сообщение перед передачей по сети разбивается на блоки длины 2^{14} байт или меньше, над которыми производятся операции в следующим порядке. Первая операция- сжатие, с точки зрения криптографии не представляет особого интереса.

Шифрование и аутентификация. Аутентификация приводится перед шифрованием информации по следующей формуле:

$$\text{HMAC\_hash(MAC\_Секрет, счетчик, сжатый текст)} \qquad (1)$$

Шифрование происходит по формуле

$$\text{Chipher(сжатый текст, подпись, \{возможное выравнивание длины\})} \qquad (2)$$

Добавление выравнивания длины до размера блока шифра происходит при использовании блочных шифров.

Рассмотрим **протокол приветствия**. Уровень, находящийся выше TLS, не может быть всегда уверен, что установлено максимально надежное соединение между двумя сторонами: есть ряд способов для криптоаналитика заставить протокол установить минимально защищенное соединение. Протокол организован так, чтобы уменьшить этот риск, но возможно, например, блокировать доступ к порту, на котором реализован сервер защиты, чтобы заставить вести неавторизованное соединение. Пользователь высшего уровня должен для себя решить, какая защита ему необходима и никогда не передавать данные по каналу меньшей защищенности, чем необходимо. Защищенным является канал, использующий 3DES с 1024 битным ключом RSA с узлом, чей сертификат был верифицирован, в отличие от соединения с ключом 40 бит. Эти цели достигаются при использовании протокола приветствия. Приветствие проходит в два этапа. На первом этапе производится аутентификация клиента и сервера. Клиент отсылает запрос на сертификат сервера. Возможны варианты, когда соединение устанавливается без проверки сертификатов, с проверкой сертификата сервера, и с сертификатами сервера и клиента. На втором этапе происходит выбор алгоритмов шифрования и аутентификации. Возможно, что в ходе приветствия не будет выбрано применение никаких алгоритмов защиты.

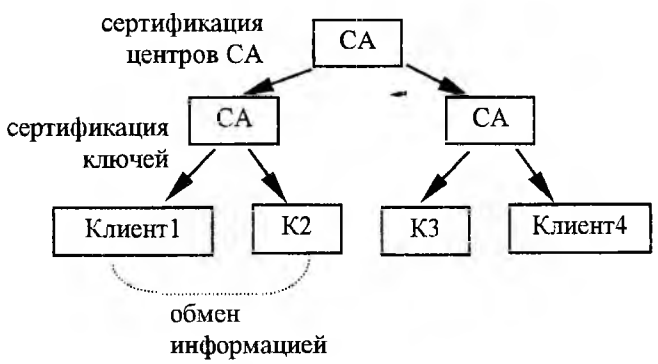
Аутентификация клиента и сервера происходит по их сертификатам, которыми обмениваются во время протокола приветствия. Сертификат содержит открытый ключ клиента, для

иесимметричного алгоритма. В сертификате содержится информация о пользователе, с которым будет установлена связь. Достоверность этой информации гарантируется подписью третьей стороны (службы сертификации), которой доверяют оба участника обмена сообщениями. Кроме того, указывается дата действия сертификата.

Таблица 2

Название поля	**Содержимое**
Информация о пользователе	Идентификационное имя, открытый ключ
Информация о подписи	Имя службы сертификации, цифровая подпись
Период действия	Дата начала действия, дата конца действия
Административная информация	Версия, серийный номер

После истечения срока действия сертификата или после его компрометации, информация о нем заносится в список компрометации (Revocation List), который может быть доступен всем пользователям. Подпись службы сертификации можно проверить на основании ее открытого ключа. Возникает вопрос: нельзя ли как-нибудь проверить подлинность самого этого ключа и как можно гарантировать его аутентичность. Этот вопрос решен. На самом деле существует иерархия центров аутентификации, открытый ключ каждого следующего из них сертифицирован центром, более старшим в этой иерархии. Таким образом, увеличивается надежность всей системы.



CA- (Center Authority) -центр сертификации

Рис. 3

Как показывает анализ TLS протокола, он обеспечивает необходимые функции целостности, причастности и скрытости содержания, но длины ключей, используемых в прототипе, не обеспечивают требуемого уровня криптостойкости. Например, длина ключа для симметричных шифров (общий секрет) составляет 48 бит, поэтому предлагается использование этих протоколов, но с ключами, отвечающими современному состоянию вычислительной техники.

В заключение следует заметить, что на смену арифметике, используемой сейчас в протоколе TLS, приходит арифметика с использованием эллиптических кривых. Существует возможность настройки протокола на использование таких криптосхем. Можно построить протокол, реализованный на шифрах: схема шифрования с использованием эллиптических кривых, цифровая подпись на эллиптических кривых, схема Диффи-Хелмана на эллиптических кривых. Следующая реализация протокола TLS предположительно должна содержать эти алгоритмы.

Харьковский государственный технический университет радиоэлектроники

Поступила в редколлегию 15.03.2000