НАУКОВО-ТЕХНІЧНИЙ ЖУРНАЛ "ЗАХИСТ ІНФОРМАЦІЇ" №4, 2003

УДК 681.14

В.П. Широчин, В.Е. Мухин, В.Б. Замков

Способы и средства повышения эффективности защищенных протоколов передачи данных

1. Введение

В настоящее время сетевые технологии обработки информации получают все более широкое распространение. Растет число пользователей глобальной компьютерной сети Internet и количество сайтов, при этом расширяется периметр угроз безопасности передаваемой и обрабатываемой информации. В сетях обрабатывается и передается значительный объем конфиденциальной инфор-мации, такой как коммерческие тайны, патенты на изобретения, персональные данные, финансовая информация и т.п. При передаче по сети незащищенных данных злоумышленник имеет возможность перехватить передаваемую информацию, что в последствии может привести к серьезным последствиям. В частности, если при доступе пользователя к удаленному компьютеру пароль передается в незашифрованном виде, злоумышленник может определить его сетевое имя (login) и пароль и, фактически, получить все полномочия данного пользователя по отношению к ресурсам. Таким образом, возникает необхо-димость разработки защищенных протоколов сетевой передачи данных.

Для обеспечения безопасной обработки информации в сетях необходимо реализовать:

- шифрование передаваемой информации;
- подтверждение подлинности (аутентификацию) субъектов сети.

Концепция защищенной передачи информации предусматривает следующие основные механизмы защиты канала передачи данных: [1]

1. аутентификацию субъектов сети;
2. гарантию целостности сообщений;
3. закрытие содержания сообщений;
4. гарантию подлинности автора сообщений;
5. обеспечение невозможности отказа автора от своих сообщений;
6. скрытие служебной информации, например структуры сети, адресата и получателя;
7. защиту от статистического анализа трафика.

Современные защищенные протоколы передачи данных в сетях должны под-держивать большинство выше перечисленных функций. Кроме основных требо-ваний, относящихся к степени защищенности информации, к средствам защиты передаваемой информации также добавляются дополнительные:

- минимальный объем служебной информации, передаваемой по сети;
- возможность работы в существующих незащищенных сетях (например, в Internet);
- простота установки и настройки.

2. Обобщенная структура корпоративной компьютерной сети

Рассмотрим типичную корпоративную компьютерную сеть, в которой осуществляется доступ к защищенному серверу с рабочих станций, находящихся в нескольких локальных сетях.

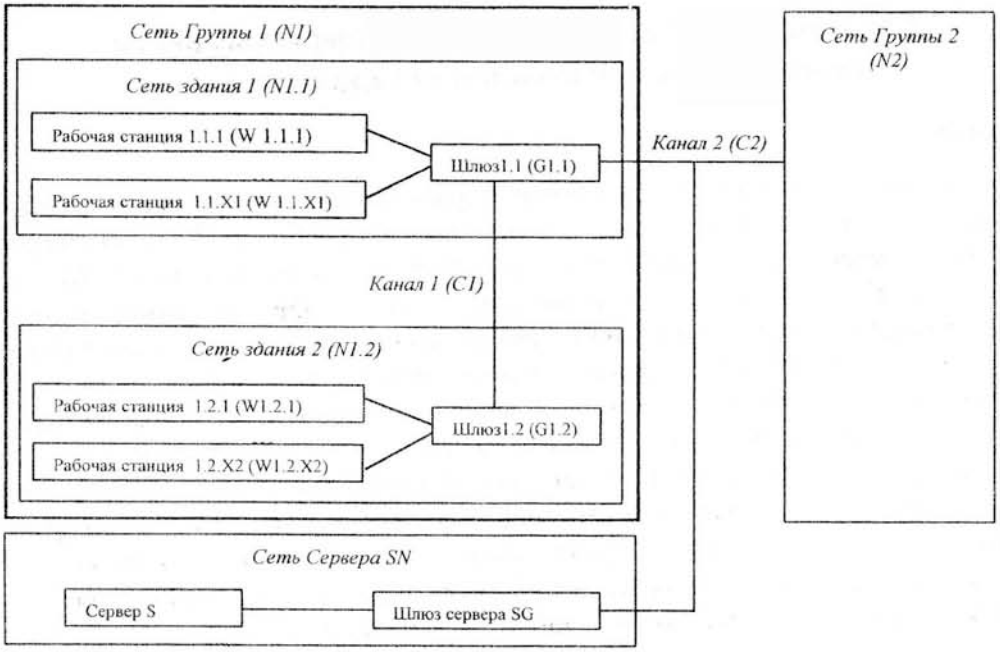

Рис.1. Структура корпоративной компьютерной сети

На рис.1 представлена конфигурация данной сети. В данной структуре все рабочие станции Группы 1 объединены в сеть N1, причем эта сеть разбита на две подсети N1.1 и N1.2, каждая из которых объединяет рабочие станции одного из двух зданий, в которых располагается группа. Сети N1.1 и N1.2 объединены между собой посредством канала C1, соединяющего шлюзы G1.1 и G1.2. Шлюз G1.1 подсоединен к внешнему каналу C2, по которому осуществляется доступ в сеть Internet через шлюз GS. На компьютерах, выполняющих функции шлюза, выполняются сервисы сетевых экранов, прокси-сервера, трансляции сетевых адресов и маршрутизации.

Группы рабочих станций могут располагаться на достаточно большом удалении друг от друга, например в разных городах, причем группа может иметь сложную внутреннюю структуру. При этом необходимо обеспечить защиту информации, передаваемой по каналам C1 и C2 и сократить ее объем, поскольку канал связи большой протяженности является относительно медленным. Такая топология реализуется, например, в больших организациях, в банковских сис-темах, в системах продажи билетов, при подключении кассовых аппаратов и т.п.

При передаче информации рабочие станции обмениваются с сервером с использованием протоколов прикладного уровня, например Telnet или X-Window, при этом формируются пакеты малых размеров. При большом количестве рабочих станций в группе объем данных, передаваемых между сервером и группой по внешнему каналу (через Internet), является значительным. Таким образом, при применении обычных средств передачи данных и защиты трафика, количество служебной информации может превысить объем прикладных данных, что существенно снижает скорость обработки информации в сети.

3. Анализ существующих защищенных протоколов передачи данных

В компьютерных сетях защита информации может одновременно выполняться на нескольких уровнях эталонной модели OSI/ISO. При этом увеличивается защищенность сети, но, ввиду выполнения криптографических преобразований и передачи служебной информации, снижается пропускная способность прикладных данных в сети. Поэтому на

практике средства защиты в компьютерных сетях реализуются на одном уровне модели OSI/ISO (канальном, сетевом, транспортном, сеансовом или прикладном). В настоящее время разработаны и применяются целый ряд защищенных сетевых протоколов: [2]

- на прикладном уровне SHTTP, S/MIME, PGP, SSH;
- на сеансовом уровне SOCKS, SSL/TLS;
- на сетевом уровне IPSec, SKIP;
- на канальном уровне PPTP, L2TP.

В общем случае, на прикладном уровне формируется пакет данных вида (рис.2):

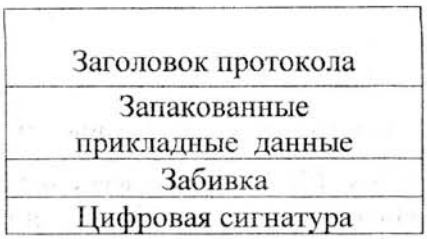

Рис.2. Обобщенная структура пакета передаваемых данных

Таким образом, пакет состоит из полезных (прикладных) данных и служебной информации.

В настоящее время в практических приложениях чаще всего используются протоколы SSH, SSL, TLS, SOCKS. Для всех вышеперечисленных протоколов выделяется общий алгоритм взаимодействия.

1. Создается **сессия** между взаимодействующими субъектами. При этом происходит согласование используемых алгоритмов сжатия данных (Compression), шифрования (Encryption), проверки целостности (Data Integrity) и обмена ключами шифрования (Key Exchange Methods). Далее, в соответствии с выбранным алгоритмом шифрования генерируется секретный ключ и весь переда-ваемый трафик шифруется с использованием выбранных параметров. Также возможна аутентификация рабочей станции для сервера и, при необходимости, сервера для рабочей станции.
2. Для каждого нового TCP-соединения создается новый **канал.**

Рассмотрим особенности реализации наиболее распространенных протоколов SSL3.0 и SOCKSv5.

Протокол **SSL3.0** обеспечивает конфиденциальность обмена данными между прикладными процессами рабочей станции и сервера. Для создания нового защищенного соединения транспортного уровня создается новая сессия или используются параметры ранее открытой сессии. Таким образом, создается новый канал, для которого определены параметры защиты информации и некоторые дополнительные параметры. Данные каждого канала передаются по отдельному TCP-соединению. [3,4] Структура пакета SSL-канала представлена на рис.3.

Поле	**Размер, байт**
ContentType	1
ProtocolVersion	2
Length	2
Fragment	Length

Рис.3. Структура пакета SSL-канала

Поле ContentType определяет тип данных, передаваемых в данном пакете: управляющие пакеты или данные прикладного процесса, ProtocolVersion – версию протокола SSL, Length – длину поля Fragment, Fragment – блок, содержащий параметры управляющего пакета или данные прикладного процесса. Структура блока Fragment для блочных алгоритмов шифрования изображена на рис.4.

Поле	Размер, байт
Content	ContentLength
MAC	HashSize
Padding	PaddingLength
PaddingLength	1

Рис.4. Структура блока Fragment для блочных алгоритмов шифрования

Здесь поле Content – передаваемые данные, MAC – цифровая сигнатура (строится на основе хэш-функции), поле Padding – байты забивки, PaddingLength содержит размер поля Padding в байтах. Эту величину подбирают так, чтобы размер Length структуры Fragment:

$$Length = CompressedLength + HashSize + PaddingLength + 1 \quad (1)$$

был кратен размеру блока шифрования.

Как видно из рис.3 и рис.4, в протоколе передается служебная информация (поля ContentType, ProtocolVersion, Length, MAC, Padding, PaddingLength) и прикладные данные (Content). Для ускорения реализации протокола необходимо минимизировать удельный вес служебной информации и тем самым сократить затраты на ее передачу.

Оценим дополнительные затраты на передачу служебной информации при реализации протокола SSL3.0 на базе протоколов TCP/IP. Рассмотрим наиболее вероятный случай: величина HashSize для широко распространенных хэш-функций MD2 и MD5 равна 16 байтам, размер блока шифрования для многих современных алгоритмов шифрования, в частности DES и TripleDES, равен 8 байт. Следовательно, математическое ожидание PaddingLength – 4 байт.

Тогда средняя величина дополнительных затрат FragmentOverhead при формировании структуры Fragment равна:

$$FragmentOverhead = HashSize + PaddingLength + 1 = 16+4+1 = 21 \text{ байт} \quad (2)$$

Средняя величина дополнительных затрат на передачу служебной информации SSL3.0Overhead в протоколе SSL3.0:

$$SSL3.0Overhead = 5 + FragmentOverhead = 26 \text{ байт} \quad (3)$$

Минимальный размер заголовка протокола IP – 20 байт, TCP – 20 байт. Таким образом, дополнительные затраты SSL3.0TotalOverhead при передаче одного пакета данных составляют:

$$SSL3.0TotalOverhead = IpHeaderSize+TcpHeaserSize + SSLOverhead = \\ =20+20+26 = 66 \text{ байт} \quad (4)$$

Протокол **SOCKSv5** используется в работе шлюзов, обеспечивающих доступ пользователей внутренней сети к серверам внешней сети. SOCKSv5 поддерживает аутентификацию субъектов и шифрование всего трафика, передаваемого между шлюзом и рабочей станцией в защищаемой сети, а также создание цепи Socks-серверов. [5]

Создание канала для отдельного защищаемого TCP-соединения начинается с передачи рабочей станцией запроса через управляющее соединение, в котором указывается адрес удаленного сервера, с которым устанавливается соединение через шлюз SOCKS. SOCKS-сервер устанавливает TCP-соединение с удаленным сервером и посылает

клиенту ответ, содержащий адрес и порт. Используя эти данные, рабочая станция устанавливает новое TCP-соединение с SOCKS-сервером -- соединение канала.

Все данные, которые передаются между рабочей станцией и SOCKS-сервером (данные управляющего соединения и соединений каналов), шифруются с исполь-зованием параметров сессии. В большинстве случаев пакет, в который инкапсу-лируется передаваемый трафик, аналогичен структуре Fragment протокола SSL3.0 (рис.4).

Таким образом, дополнительные затраты на передачу служебной информации SOCKSv5TotalOverhead в протоколе SOCKSv5 для одного пакета данных состав-ляют:

SOCKSv5TotalOverhead= IpHeaderSize +TcpHeaserSize +FragmentOverhead=
20+20+21= 61 байт (5)

4. Разработка защищенного протокола передачи данных повышенной эффективности

4.1. Общая концепция протокола

Одной из целей разработки новых защищенных протоколов передачи данных является уменьшение удельного объема служебной информации, передаваемой по внешним каналам корпоративных сетей при использовании систем защиты трафика. Как показано в [6], пропускная способность прикладных данных сети, в которой реализуются средства защиты данных, зависит от затрат времени на реализацию защитных функций и передачу служебной информации.

В общем случае, к разрабатываемому защищенному протоколу передачи данных предъявляются следующие требования:

- степень безопасности протокола должна быть достаточно высокой;
- дополнительная нагрузка на аппаратные ресурсы устройств, входящих в состав сети должна минимизироваться;
- модернизация программно-аппаратных ресурсов для поддержки протокола должна быть минимальной и максимально экономичной.

Для повышения эффективности средств передачи данных в компьютерных сетях предлагается новый протокол сеансового уровня. Он наследует свойства протокола SOCKSv5, обеспечивает защищенную передачу данных через незащищенные сети, работающие на основе протокола TCP/IP (например, Internet), и при этом поддерживает более высокую пропускную способность прикладных данных в сети и защищенность по сравнению с существующими средствами.

Для работы протокола необходимо сначала установить соединение транспортного уровня между рабочими станциями, т.е. **базовое соединение**. Перед установлением соединения производится аутентификация субъектов, для чего используются цифровые сигнатуры, обеспечивающие целостность переда-ваемой информации. Степень защиты сигнатуры должна быть значительно выше, чем защита соединения.

После аутентификации определяются следующие параметры соединения:

- максимальный размер пакета;
- алгоритм сжатия данных;
- алгоритм шифрования;
- алгоритм проверки целостности данных;
- алгоритм генерации ключей шифрования.

Ключи шифрования определяются в соответствии с выбранным крипто-алгоритмом. Все передаваемые пакеты формируются в виде структуры SessionPacket. Систему, обеспечивающую передачу данных в таком виде назовем **соединением**.

4.2. Создание каналов и передача данных

Для передачи данных каждого соединения транспортного уровня организуется отдельный **канал**. При создании канала определяется идентификатор канала ChannelId, причем значение ChannelId уникально для каждой сессии. Таким образом, канал однозначно идентифицируется IP-адресами взаимодействующих узлов и идентификатором канала ChannelId. Конечные узлы (в рассматриваемой сети это рабочие станции W.x.y.z и сервер S) хранят информацию о соответствии ChannelId адресу узла, с которым происходит взаимодействие по данному каналу, и номерам TCP-портов.

Передаваемые данные прикладного уровня преобразуются в структуру ChannelPacket. Канальные пакеты мультиплексируются в пакеты сессии.

Рассмотрим структуру пакетов канала и сессии. На рис.5 представлена структура пакета ChannelPacket.

Поле	Длина (байт)	Описание
ChannelCommand	1	команда
ChannelId	2	идентификатор канала
DataLength	2	длина передаваемых данных
ChannelData	DataLength	прикладные данные

Рис.5. Структура пакета ChannelPacket

Здесь поле ChannelCommand определяет содержание поля ChannelData и действия шлюза, принявшего этот пакет. Если поле ChannelData содержит передаваемые данные прикладного уровня, то в поле ChannelCommand установлено значение CH_DATA. Также поле ChannelCommand определяет тип управляющих пакетов, например, запрос на открытие канала, сообщение об успешно обработанном запросе или об ошибке, требование закрытия канала. Поле ChannelId содержит идентификатор канала, к которому относится данный пакет, поле DataLength содержит длину поля ChannelData.

4.3. Передача группового пакета

Предлагаемый протокол предусматривает передачу групповых пакетов, для чего из пакетов данных, передаваемых по каждому каналу в виде структур ChannelPacket, в два этапа формируется структура SessionData. На первом этапе формируется структура MultiChannel (рис. 6)

Название	Длина (байт)	Описание
NumberOfChannelPackets	2	количество канальных пакетов
ChannelPacket(ch_1)	ChLength(ch_1)	структура ChannelPacket, полученная по каналу ch_1
. . .		
ChannelPacket(ch_N)	ChLength(ch_N)	структура ChannelPacket, полученная по каналу ch_N (N = numberOfChannelPackets)

Рис.6. Структура пакета MultiChannel

Пусть существует M каналов, обслуживаемых данным шлюзом. Из этих M каналов, в данный момент времени происходит передача пакетов, по N каналам ($N \le M$), при этом N записано в поле NumberOfChannelPackets. Обозначим i-й канал из N каналов, передающих пакеты в данный момент времени как ch_i. Тогда ChannelPacket(ch_i) – структура ChannelPacket, полученная по каналу ch_i, и ChLength(ch_i) – длина ChannelPacket(ch_i)

На втором этапе структура MultiChannel сжимается с использованием специального алгоритма сжатия, в результате чего формируется пакет CompressedData, при этом его длина CLength зависит от алгоритма сжатия и пере-даваемой информации. Далее формируется структура SessionData (рис.7)

Название	**Длина (байт)**	**Описание**
SessionCommand	1	команда сессии
UeLength	2	длина всего пакета, включая поле Length
PaddingLength	1	длина забивки
CompressedData	CLength	структура CompressedData
Padding	PaddingLength	Забивка

Рис.7. Структура пакета SessionData

Поле SessionCommand содержит команды, относящиеся к управлению сессией, например, запрос на повторный обмен ключами, переопределение параметров соединения, разрыв соединения и т.д. Для передачи канальных пакетов в виде структуры MultiChannel, поле SessionCommand должно содержать значение MC_DATA. Для передачи единственного пакета используется команда SC_DATA. В этом случае пакет CompressedData получается из структуры SingleChannel, содержащей поля ChannelId и ChannelData.

При передаче множественных пакетов значение поля PaddingLength определяется таким образом, чтобы суммарная длина UeLength структуры SessionData:

$$UeLength = 1 + 2 + 1 + CLength + PaddingLength = 6 + CLength + PaddingLength \quad (6)$$

была кратна длине блока шифрования.

Полученная структура SessionData шифруется с помощью выбранного алгоритма шифрования, в результате чего получается пакет EncryptedPacket длиной UeLength

Далее, на основе алгоритма проверки целостности вычисляется цифровая сигнатура MAC. Именно такая последовательность (сначала шифрование, затем цифровая сигнатура) обеспечивает быстрое обнаружение и удаление повреж-денных пакетов. В результате, формируется пакет SessionPacket, который передается через базовое соединение (рис.8).

Название	Длина (байт)	Описание
EncryptedPacket	EpLength	структура EncryptedPacket
MAC	HashSize	цифровая сигнатура

Рис.8. Структура пакета SessionPacket

Оценим дополнительные затраты на передачу служебной информации при использовании предлагаемого протокола на основе базовых протоколов TCP/IP. При этом будем предполагать, что используются такие же характеристики шифруемого канала, как и при оценке дополнительных затрат в аналогах (протоколах SSL3.0 и Socksv5) и сжатие не используется, т.е. CompressedPacket идентичен UncompressedPacket.

При передаче единственного пакета данных дополнительные затраты рассчитываются как:

SingleChannelHeaderSize = 2

SessionDataHeaderSize = 1 + 2 + 1 + PaddingLength = 4 + 4 = 8

TotalOverhead = IpHeaderSize + TcpHeaserSize + SessionDataHeaderSize + SingleChannelHeaderSize + HashSize = 20 + 20 + 8 + 2 + 16 = 66 байт (7)

Далее определим величину дополнительных затрат при передаче одного группового пакета.

ChannelPacketHeaderSize = 1 + 2 + 2 = 5
MultiChannelHeaderSize = 2 (8)

Тогда для множества пакетов дополнительные затраты на передачу служебной информации TotalOverhead рассчитываются как:

TotalOverhead = IpHeaderSize + TcpHeaserSize + SessionDataHeaderSize + MultiChannelHeaderSize + ChannelPacketHeaderSize * N + HashSize = 20 + 20 + 8 + 2 + 5 * N + 16 = 66 + 5 * N байт, (9)

где N – количество канальных пакетов, передаваемых в групповом пакете сессии.

Выражение (9) показывает, что при реализации данного протокола для передачи множественных пакетов дополнительные затраты также зависят от количества пакетов, но при этом растут не столь быстро, как в случае протоколов SSL3.0 и SOCKSv5.

4.4. Работа шлюза

Рассмотрим работу шлюза в обобщенном варианте, на примере шлюза G1.1 в корпоративной сети (рис. 1). С точки зрения протокола передачи данных, рабочие станции работают аналогично шлюзам, различия заключаются в обработке принятых и передаваемых пакетов.

Шлюз G1.1 имеет установленные сессии с несколькими рабочими станциями W1.1.1 - W.1.X1 и шлюзами G1.2 и GS. Шлюз G1.1 получает пакеты сессий (SessionPacket) от шлюза G1.2 и от рабочих станций W1.1.1 - W.1.X1 и извлекает из них множество канальных пакетов (ChannelPacket). В течение определенного интервала времени все полученные таким образом канальные пакеты хранятся в буфере. Причем для каждого канального пакета сохраняются IP-адреса отправи-теля и получателя, идентифицирующие сессию. По истечении заданного времени все канальные пакеты, находящиеся в буфере, помещаются в один групповой пакет сессии, который отправляется шлюзу SG.

4.5. Преимущества и недостатки предложенного протокола

Предлагаемый выше протокол обеспечивает повышение пропускной способ-ности прикладных данных при повышенном уровне защищенности их передачи.

Повышение пропускной способности прикладных данных в протоколе реализуется за счет:

- мультиплексирования пакетов данных, поступающих от конечных узлов;
- сжатия заголовков протокола TCP/IP.

Мультиплексирование пакетов, поступающих от конечных узлов реализуется следующим образом. Как было показано выше, шлюз помещает в один пакет сессии несколько пакетов каналов. С другой стороны, благодаря специфике механизма создания канала, его параметры (адреса конечных узлов и номера TCP-портов) могут быть определены взаимодействующими сторонами один раз до передачи информации. Таким образом, при дальнейшем взаимодействии нет необходимости снова передавать эту информацию.

Сжатие заголовков протоколов возможно ввиду того, что в TCP-пакет помещается и передается от одного шлюза к другому уже подготовленный пакет сессии (SessionPacket). Поэтому маршрутизаторы, находящиеся на пути между двумя шлюзами, не нуждаются в информации, относящейся к взаимодействию конечных узлов. Таким образом, существует возможность заменить заголовок протоколов IP и TCP идентификатором канала, который в предлагаемой реализации занимает 2 байта. Также

сокращается объем передаваемой служебной информации для поддержки функций защиты данных: поля MAC и Padding присутствуют только в пакете сессии.

За счет применения в протоколе более совершенных криптографических алгоритмов повышается его защищенность, и как результат, безопасность канала передачи данных. В существующих защищенных протоколах передачи данных для шифрования обычно используются алгоритмы DES и TripleDES. В нашем протоколе предлагается использовать криптоалгоритм ANUBIS, который имеет характеристики защищенности на несколько порядков выше, чем у DES и TripleDES. [7] При этом, ввиду свойств ANUBIS, затраты времени на проведения шифрования несколько снижаются. Особенностью ANUBIS является применение ключей шифрования длиной 128, 160, 192, 224, 256, 288, 320 бит, что позволяет шифровать блоки данных соответствующего размера и обеспечивает адаптив-ность длины пакетов.

Возможна аппаратная реализация криптографических алгоритмов на одном шлюзе, что позволяет при достаточно низких затратах (единственный крипто-процессор на шлюзе) обеспечить шифрование высокой стойкости. Данный подход обусловлен тем, что рабочие станции, в рассматриваемой корпоративной сети – относительно маломощные компьютеры, не способные обеспечить такое шифрование в реальном времени. Таким образом, применение алгоритма шифрования повышенной эффективности при передаче информации между шлюзами позволяет существенно повысить защищенность протокола передачи данных при минимальных затратах на усовершенствование оборудования.

К числу некоторых недостатков предлагаемого протокола следует отнести:

- необходимость модернизации программного обеспечения на каждой рабочей станции;
- выделение дополнительных временных и аппаратных ресурсов на шлю-зах;
- появление открытой информации в процессе шифрования/дешифрования на шлюзах, что требует дополнительных мер по их защите.

Однако выше перечисленные недостатки не являются существенными и в целом эффективность предлагаемого протокола выше, чем у существующих стандартных средств обеспечения защиты передачи данных.

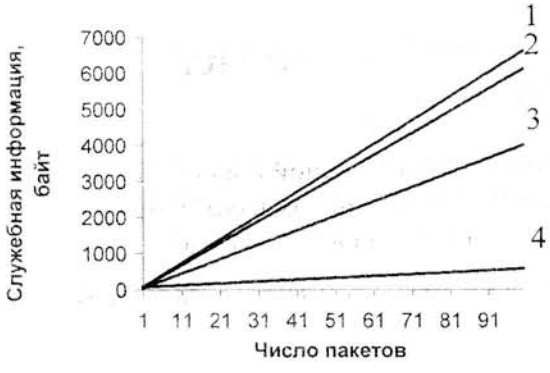

Рис.9. Зависимость объема служебной инфор-мации от числа передаваемых пакетов в протоколах

1 – протокол SSL3.0, 2 – протокол SOCKSv5,

3 – протокол TCP,4 – предлагаемый протокол

Рис.10. Зависимость отношения объема служебной информации к прикладным данным от числа передаваемых пакетов в протоколах

5. Сравнительная оценка пропускной способности прикладных данных в защищенных протоколах передачи данных

На рис.9 и рис.10 представлены зависимости объема передаваемой служебной информации (в абсолютных и относительных единицах как отношение служебной информации к прикладным данным) от числа передаваемых пакетов за заданный промежуток времени для существующих и предлагаемого протокола.

Данные для рис.9 и рис. 10 получены при условиях, когда средний размер пакета протокола прикладного уровня составляет 10 байт, а размер блока шифрования данных — 8 байт. Как видно из рис.9, передаваемый объем служебной информации в предлагаемом протоколе передачи данных значительно меньше, чем в существующих защищенных протоколах (SSL3.0, Socksv5), причем эта разность растет с увеличением числа пакетов. Кроме того, рис.10 показывает, что удельный объем служебной информации в предлагаемом протоколе снижа-ется с ростом числа передаваемых пакетов, что обусловлено мультиплексиро-ванием пакетов данных и сжатием заголовков протоколов, тогда как для существующих протоколов эта характеристика остается постоянной.

На рис.11 и рис.12 представлены зависимости пропускной способности прикладных данных в корпоративных сетях, реализующих существующие и предлагаемый протоколы, соответственно от числа и размера передаваемых пакетов.

1 – протокол SSL3.0, 2 – протокол SOCKSv5, 3 – протокол TCP, 4 – предлагаемый протокол

Рис.11
Зависимость пропускной способности прикладных данных в сетях от числа переданных пакетов в протоколах

Рис.12
Зависимость пропускной способности прикладных данных в сетях от размера пакетов в протоколах

При этом пропускная способность прикладных данных в сети рассчитывается как γ (Kb/c):

$$\gamma = \frac{I}{T_П + T_C} \qquad (10)$$

где I (Kb) — объем передаваемых прикладных данных (Kb), $T_П$ (c) — время передачи прикладных данных по сети, T_C (c) — время передачи служебной информации.

Данные для рис.11 и рис.12 получены при условии, когда средний размер пакета протокола прикладного уровня составляет 10 байт, размер блока шифро-вания данных — 8 байт, пропускная способность канала сети — 5 Kb/c.

Как видно из рис.11, при применении предложенного протокола пропускная способность прикладных данных в сети растет с увеличением числа переда-ваемых

пакетов, тогда как в существующих протоколах (TCP, SSL3.0, Socksv5) она остается постоянной. Так, для количества пакетов равного 100, пропускная способность при реализации предложенного протокола возрастает в 3-5 раз.

Рис.12 показывает, что в общем случае при увеличении размера пакетов пропускная способность прикладных данных в сети растет, причем для предлагаемого протокола этот рост оказывается наибольшим, что еще раз подтверждает эффективность нового защищенного протокола.

6. Заключение

По статистике, в настоящее время большинство сетевых передач данных выполняется посредством незащищенных протоколов и без шифрования трафика, в результате чего передаваемые данные могут подвергаться перехвату злоумыш-ленниками. При передаче по сети конфиденциальной информации необходимо использовать специальные защищенные протоколы передачи данных. Одним из основных критериев уровня безопасности сетевых передач является крипто-стойкость используемого алгоритма шифрования и размер ключей шифрования. С другой стороны, увеличение размера ключа, в общем случае ведет к снижению пропускной способности сети, в которой реализуется защищенный протокол передачи данных, и, как следствие, применение стандартного защищенного протокола не во всех случаях приводит к эффективной защите. Таким образом, разработка и усовершенствование защищенных протоколов передачи данных повышенной эффективности, один из которых предложен в данной статье, обеспечивающих одновременно высокий уровень безопасности передачи данных и достаточную пропускную способность сети, является весьма актуальной задачей.

Литература:

1. В.И. Мельников. Защита информации в компьютерных системах. М., Радио и связь, 1997. – 362 с.
2. Б. Скородумова. Особенности защиты информации автоматизированных бан-ковских систем.// Банковские технологии, № 2, 2001. – с. 48-51.
3. *E.B. Kipp, H. T. Elgamal. The SSL Protocol.* Internet Draft Netscape Communica-tions Corp., June 1995. – 132 p.
4. Э. Сергеева. Протокол SSL и защита информации в системе RS-Portal.//RS-Club, № 4(19), 2000.– с. 40-44.
5. S.Kolletzki. Secure Internet Banking with Privacy Enhanced Mail – A Protocol for Relaible Exchange of Secured Order Form. // Proceedings JENC7, Budapest, May 13-16, 1996.– pp. 232- 239.
6. В.П.Широчин, В.Е. Мухин. Формализация и целевая адаптация средств аутен-тификации в компьютерных сетях. // Управляющие системы и машины, N 5/6, 2000. – с.59 -65.
7. В.Е.Мухин, В.С. Пурнынь. Повышение эффективности средств подтвержде-ния подлинности сообщений в компьютерных сетях. //Вісник НТУУ "КПІ". Серія "Інформатика, управління та обчислювальна техніка", N 37, 2002. – с. 66 - 75.

Поступила 24.04.2003