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

The Study of SSH

Пржиялковский Ян Владимирович.

2 мая 2005

Введение в SSH

В наши дни Интернет стал доступным и дешёвым и, как следствие, он стал заменять традиционные средства связи, например, телефон, факс, курьер, для доступа к ресурсам внутреннего компьютера компании. Раньше, компании поддерживали свои собственные компьютеры для удалённого доступа используя телефонные линии. Таким образом критичные для компании данные не передавались по открытым сетям. Но такой способ защитить важные данные от перехвата имеет свои недостатки: во-первых, это дорого поддерживать отдел удалённого доступа, особенно для больших организаций, во-вторых, возникает проблема доступа к сети компании в случае, когда пользователь находится далеко, в-третьих, такой способ доступа к компьютерам имеет ограничения по скорости передачи данных, что может быть существенно.

В операционных системах семейства UNIX существует несколько способов удалённого управления компьютером и передачи файлов. Это rlogin, rsh, ftp, X11, и т.д. Недостаток всех этих протоколов состоит в том, что информация передаётся по открытым каналам связи в открытом виде, в том числе и пароли доступа. Этот недостаток даёт возможность злоумышленнику перехватить важную информацию.
Чтобы убрать этот недостаток, был введён протокол SSH - протокол, позволяющий управлять компьютером безопасно, то есть информация передаётся по шифрованному каналу связи.

История SSH

Протокол SSH применяется с 1995 года. Первая версия протокола была сделана для того, чтобы заменить небезопасные протоколы удалённого доступа к компьютеру по сети rsh, rlogin, rcp. В 1997 году Internet Engineering Task Force (IETF) ввела новый стандарт протокола SSH – вторую версию. В SSH2 были исправлены некоторые серьёзные недостатки в системе безопасности, а так же была улучшена система передачи файлов. Далее популярность этого протокола быстро росла, в основном за счёт того, что это открытый стандарт, программное обеспечение распространяется бесплатно и с открытым кодом.

Описание SSH

Под SSH понимают как собственно программу, так и задействованный в ней протокол. Что касается программы, то для ее краткой характеристики следует сказать, что SSH представляет собой средство организации безопасного доступа к компьютерам при работе по небезопасным каналам связи. Для организации безопасного доступа применяется процедура аутентификации с использованием асимметричного шифрования с открытым ключом. Это обеспечивает более высокую безопасность, чем при использовании симметричного шифрования, хотя и порождает дополнительную вычислительную нагрузку. При последующем обмене данными применяется уже симметричное шифрование, более экономичное в смысле затрат процессорного времени. [1]

Стандарт SSH описывает протоколы SSH и состоит из нескольких документов, определяющих общую архитектуру протокола. Secure Shell состоит из трёх компонентов:

- Протокол транспортного уровня [The transport Layer Protocol - SSH-TRANS]. Обеспечивает аутентификацию сервера, конфиденциальность и целостность

передаваемой по сети информации. Так же поддерживается функция сжатия данных. Протокол транспортного уровня работает поверх соединения TCP/IP (порт SSH – 22-ой порт протокола TCP).

- Протокол аутентификации [The User Authentication Protocol - SSH-USERAUTH]. С целью повышения безопасности происходит аутентификация клиентской стороны для сервера. Для этого был введён этот протокол. Протокол аутентификации работает поверх протокола транспортного уровня.

- Протокол соединения [The Connection Protocol - SSH-CONNECT]. Позволяет разделить шифрованный канал связи уровня TCP/IP на несколько логических каналов, что даёт возможность безопасной от прослушивания работы с несколькими различными сервисами.

Как только установится безопасное соединение транспортного уровня SSH (поверх соединения TCP), клиент посылает запрос серверу на обслуживание. После завершения аутентификации пользователя, посылается второй запрос серверу.

Разработчики проекта протокола SSH особенно заботились о его долголетии. Протокол будет расширяемым; планируется возможность дополнения криптографических алгоритмов, используемых при работе SSH. С этой целью проектом предусмотрено, что между клиентом и сервером происходят переговоры, в результате которых выбираются методы шифрования, форматы открытых ключей и т.п., которые будут использованы в данном сеансе. [1]

Ключ хоста.

Каждый работающий с SSH сервер должен иметь не менее одного так называемого ключа хоста. У сервера может быть несколько ключей, для каждого алгоритма шифрования. Несколько хостов могут разделять один и тот же ключ между собой. Однако, каждый сервер должен иметь хотя бы один ключ, с которым работает каждый из требуемых алгоритмов работы с открытыми ключами. В настоящее время, требуемый алгоритм только один – DSS (Digital Signature Standard).

Ключ хоста используется во время обмена открытыми ключами, чтобы проконтролировать тот факт, что клиентская сторона установила связь именно с настоящим сервером, с которым ей необходимо установить связь. Чтобы это было возможным, нужно чтобы клиент a priori знал ключ хоста. Это знание реализуется в рамках одной из двух моделей:

- На машине клиента имеется локальная база данных, которая связывает имя хоста с его открытым ключом. Этот способ не требует присутствия третьей стороны для установления связи с сервером.

- Соответствие имени сервера и его открытого ключа хранится у так называемого сертифицированного агента (Certification Authority - CA) – организации, которая отвечает за проверку соответствия имени и ключа. В этом случае, клиент знает только открытый ключ этой организации, и с помощью неё может узнать ключ необходимого ей сервера.

Недостаток первой модели заключается в поддержке локальной базы. При первом соединении с сервером, когда клиент ещё не знает его открытого ключа, этот ключ

передаётся сервером пользователю. В этом случае становится возможна атака типа «человек в середине» (Man-in-the-Middle), в результате успешного применения которой может быть подменён настоящий сервер ложным. Этого недостатка не существует во второй модели, которая возлагает поддержку базы данных на третью сторону – клиенту достаточно знать только один открытый ключ – ключ сертифицированного агента. Но тут появляются высокие требования к такому агенту.

Протокол транспортного уровня (The transport Layer Protocol)

Транспортный уровень SSH является безопасным низкоуровневым транспортным протоколом. Он обеспечивает шифрование данных, криптографическую аутентификацию хоста и защиту от подмены передаваемых данных. Аутентификация на этом уровне протокола SSH основана на хостах, а не на аутентификации пользователя, как это делается в традиционных способах удалённого доступа. Аутентификация пользователя может быть реализована в протоколах высокого уровня.

При использовании TCP/IP, сервер, предоставляющий услуги SSH, ждёт запрос на создание соединения на 22-м порту (Этот порт был зарегистрирован IANA - The Internet Corporation for Assigned Names and Numbers для Secure Shell). Как только соединение устанавливается, обе стороны посылают друг другу идентификационные строки. Такая строка должна иметь следующий вид:

SSH-protocolversion-softwareversion SP comments CR LF,

где *protocolversion* – версия стандарта SSH. В настоящее время используется версия 2.0,
softwareversion – идентификация и версия используемого программного обеспечения,
SP – пробел,
comments – любой комментарий,
CR – Carriage Return,
LF – Line Feed.

Пример такой строки:

SSH-2.0-OpenSSH_3.6p1q2

Сразу после обмена идентификационными строками начинается процесс обмена ключами. Обмен ключами начинается с того, как каждая сторона посылает список поддерживаемых алгоритмов шифрования и аутентификации сообщения. В каждом списке указываются предпочитаемые алгоритмы. Обмен ключами определяет, каким образом вырабатываются сессионные ключи для шифрования и для аутентификации. Два метода обмена ключами определены в стандарте SSH:

diffie-hellman-group1-sha1 (алгоритм Diffie-Hellman с SHA-1 в качестве HASH, и Oakley Group 2 [RFC2409]),
diffie-hellman-group14-sha1 (алгоритм Diffie-Hellman с SHA-1 в качестве HASH, и Oakley Group 14 [RFC3526]).

После обмена ключами, клиентская сторона запрашивает сервис у сервера. Конкретный сервис идентифицируется своим именем. Среди них есть SSH-userauth и SSH-connection.

Протокол аутентификации (The User Authentication Protocol - SSH-USERAUTH)

Протокол аутентификации SSH – протокол аутентификации пользователя для сервера. Этот протокол работает поверх протокола транспортного уровня. То есть он возлагает функции конфиденциальности и целостности на более низкий уровень. В начале работы протокола, он получает идентификатор сессии от транспортного уровня (хэш H от первого обмена ключами). Идентификатор сессии однозначно и уникально идентифицирует сессию и используется для цифровой подписи, чтобы проконтролировать подлинность сообщения. Этот протокол должен также знать, осуществляется ли на более низком уровне конфиденциальность данных. Сервер сообщает клиенту, который из методов аутентификации может быть использован, чтобы воспользоваться сервисом. (public key, password, host based, none). Клиент, в свою очередь, может выбрать любой из предложенных сервером методов. Наиболее распространённым методом аутентификации является private key method. В этом методе аутентификацией служит обладание секретным ключом клиентом. Клиент посылает цифровую подпись, созданную с помощью секретного ключа пользователя. Эти ключи хранятся в зашифрованном виде на клиентской машине.

Протокол соединения (The Connection Protocol - SSH-CONNECT)

Протокол соединения работает поверх транспортного уровня и уровня аутентификации. Этот протокол обеспечивает интерактивные сессии управления компьютером, удалённые исполнения команд, перенаправление соединений TCP/IP, перенаправление соединений X11. Сессии протокола соединения являются логическими каналами связи. Каждая из сторон может инициировать создание такого канала. Запрос на создание канала содержит номер канала посылающего запрос. Все последующие сообщения, в которых указан этот номер, передаются в этот канал. У каналов существует система контроля потока, то есть данные не посылаются в канал, пока не придёт сообщения о том, что можно посылать следующую порцию данных.

Формат пакета SSH имеет следующий вид:

```
uint32      packet_length
byte        padding_length
byte[n1]    payload
byte[n2]    random padding
byte[m]     mac (Message Authentication Code – MAC)
```

packet_length – длина пакета в байтах, не включая mac и само поле packet_length.
padding_length – длина дополнительного заполнения random padding.
payload – данные, которые передаются с пакетом. Если используется компрессия, то это поле сжимается. n1 = packet_length - padding_length – 1.
random padding – дополнительное заполнение, его длина – не фиксированная величина. Это поле вводится для того, чтобы длина всего блока данных, который будет шифроваться (packet_length + padding_length + payload + random padding) имел размер кратный длине блока шифрования. Максимальная длина дополнительного заполнения – 255 байт. n2 = padding_length.

mac – Message Authentication Code. Если в соединении используется аутентификация самого посылаемого сообщения, то в это поле содержит соответствующую информацию. m = mac_length

Если стороны договариваются о сжатии передаваемых данных, поле payload (и только оно) сжимается договорённым методом. Поля packet_length и mac вычисляются только после сжатия поля payload. Шифрование начинается после сжатия. В современном стандарте SSH поддерживаются только два метода сжатия:

none	без компрессии,
zlib	ZLIB (LZ77) сжатие[RFC1950 и RFC1951].

Об алгоритме шифрования и ключе шифрования стороны договариваются во время обмена ключами, который происходит после обмена идентификационными строками. Настоящая версия SSH поддерживает следующие алгоритмы шифрования:

3des-cbc	3DES-CBC
blowfish-cbc	Blowfish-CBC
twofish256-cbc	Twofish-CBC; 256-битный ключ
twofish-cbc	То же самое
twofish192-cbc	Twofish-CBC; 192-битный ключ
twofish128-cbc	Twofish-CBC; 128-битный ключ
aes256-cbc	AES-CBC; 256-битный ключ
aes192-cbc	AES-CBC; 192-битный ключ
aes128-cbc	AES-CBC; 128-битный ключ
serpent256-cbc	Serpent-CBC; 256-битный ключ
serpent192-cbc	Serpent-CBC; 192-битный ключ
serpent128-cbc	Serpent-CBC; 128-битный ключ
arcfour	ARCFOUR потоковый шифр с 128-битным ключом
idea-cbc	IDEA-CBC
cast128-cbc	CAST-128-CBC
none	Без шифрования.

Почти все шифры используются в режиме цепочки блоков шифротекста (Cipher Block Chaining). Это означает, что к каждому блоку применяется бинарная операция XOR с предыдущим зашифрованным блоком перед кодированием. Это снижает риск обнаружения образца и, соответственно, риск взлома.

Контроль целостности данных достигается путём вставки в каждый пакет поля mac, которое вычисляется на основе общего секрета, последовательного номера пакета и содержимого пакета:

mac = MAC (key, sequence number, unencrypted packet).

Об алгоритме аутентификации сообщения и ключе стороны договариваются вначале соединения, во время обмена ключами. Настоящая версия SSH поддерживает следующие алгоритмы аутентификации сообщения:

hmac-sha1, hmac-sha1-96, hmac-md5, hmac-md5-96, none.

Список используемой литературы:

1. http://www.osp.ru/os/1999/02/02.htm
2. http://www1.vandyke.com/solutions/SSH_overview/
3. http://www.openSSH.com/txt/draft-ietf-secsh-architecture-12.txt
4. http://www.openSSH.com/txt/draft-ietf-secsh-transport-14.txt
5. http://www.openSSH.com/txt/draft-ietf-secsh-connect-15.txt