Студент: Владимир Пискунов
Группа 911

SSH (Secure SHell)

Содержание

I. Принцыпы работы SSH
II. Аутентификация и обмен данными
III. Туннелирование
IV. Примеры реализации протокола.
V. Уязвимость
VI. Список используемой литературы

I. Введение

Одна из самых распространенных задач, возникающих при работе в сети – удаленный доступ к другим компьютерам и предоставление такого доступа.

Для решения этой задачи используется ставший уже традиционным протокол SSH (Secure Shell). В отличие от устаревших протоколов, таких как telnet и rsh/rlogin/rcp, которые передают данные прямым текстом и подвержены обыкновенному прослушиванию и различным атакам, SSH реализует соединение с удаленным компьютером, защищающее от:

1. прослушивания данных, передаваемых по этому соединению;
2. манипулирования данными на пути от клиента к серверу;
3. подмены клиента либо сервера путем манипулирования IP-адресами, DNS либо маршрутизацией.

В дополнение к отличным характеристикам в области обеспечения безопасного клиент-серверного соединения, SSH обладает следующими возможностями:

1. сжатие передаваемых данных;
2. туннелирование каналов внутри установленного соединения – в т.ч. соединений с X-сервером;

3. широкая распространенность: существуют реализации SSH для самых различных аппаратных платформ и операционных систем.

SSH представляет собой протокол транспортного уровня, аутентификации и соединения (предполагается стандартизация IETF) и программные средства безопасного доступа к компьютерам по небезопасным каналам связи (telnet, X11, rsh, ftp). Аутентификация производится с использованием ассиметричного шифрования с открытым ключом (SSH1 - RSA, SSH2 - RSA/DSA). Обмен данными - симметричное шифрование (IDEA, DES, triple DES, ARCFOUR, BLOWFISH, CAST128, AES/Rijndael). Целостность переданных данных проверяется с помощью CRC32 в SSH1 (подделывается при атаке типа "man in the middle") или HMAC-SHA1/HMAC-MD5 в SSH2. Протокол транспортного уровня работает поверх TCP, обеспечивает аутентификацию сервера. В качестве ключа используется случайная строка, которую генерирует клиент, шифрует с помощью открытого ключа сервера - его надо откуда-то получить - и передаёт ему (сервер знает свой частный ключ и дешифрует строку, затем передаёт её клиенту, если строка дешифрована правильно, значит сервер настоящий). Протокол аутентификации работает поверх протокола транспортного уровня, обеспечивает аутентификацию клиента для сервера. Протокол соединения - поверх протокола аутентификации, мультиплексирует безопасный канал. Шифрование начинается после аутентификации сервера, но до аутентификации клиента, т.е. паролей в открытом виде не передаётся вовсе. Автоматически производится X11 forwarding (по наличию переменной DISPLAY). Возможно соединение произвольных портов TCP по защищенным каналам. SSH использует алгоритм сжатия LempleZiv (LZ77), обеспечивая такую же компрессию, как и zip-архиваторы.

II. Аутентификация и обмен данными

Часто SSH делят на три уровня

- Транспортный уровень обеспечивает взаимодействие алгоритма и обмен ключами. Обмен ключами позволяет аутентифицировать сервер и создать криптографически защищённое соединение: обеспечивая целостность, конфиденциальность и дополнительное сжатие.
- Уровень аутентификации пользователя использует установленное соединение и располагается на транспортном уровне. Этот уровень предоставляет несколько механизмов для аутентификации пользователя: здесь может использоваться

традиционная парольная аутентификация, аутентификация основанная на публичном ключе и т.д.

- Уровень соединения объединяет множество параллельных каналов, располагается на аутентификационном уровне и позволяет создавать login сессии, либо пересылать TCP соединения. Обеспечивает корректность передачи для этих каналов.

Реализация механизма работы уровней в SSH1 и SSH2 различна:

SSH1: каждый хост имеет RSA ключ хоста (обычно 1024 бит), при запуске sshd генерирует временный (1 час, на диск не записывается) RSA ключ сервера (обычно 768 бит). После соединения клиента, сервер передает ему публичный ключ хоста и публичный ключ сервера. Клиент сверяет публичный ключ хоста со своей базой ключей, затем генерирует случайное число (256 бит) и шифрует его публичными ключами хоста и сервера, после чего отсылает серверу. В дальнейшем это случайное число используется как ключ сессии - с его помощью шифруются все сообщения. Алгоритм шифрования предлагается сервером и выбирается клиентом - 3DES (по умолчанию) или Blowfish (быстрее).

SSH2: Каждый хост имеет DSA/RSA ключ хоста. Ключ сервера не генерируется. Аутентификация сервера и ключ сессии обеспечивается с помощью алгоритма Diffie-Hellman. Остаток сессии шифруется симметричым алгоритмом шифрования: Blowfish, 3DES, CAST128, Arcfour, AES (128, 192, 256 бит). Целостность сессии обеспечивается hmac-sha1 или hmac-md5. Аутентификация клиента происходит с помощью публичных ключей или "обычных" паролей или интерактивного обмена с помощью дополнительных средств.

III. Туннелирование

SSH-туннелирование очень мощная возможность, предоставляемая SSH. Если возможно установить SSH-соединении от одного хоста до другого, можно туннелировать и другие TCP-подключения по безопасному зашифрованному SSH-сеансу. Это позволяет защищать другие не шифруемые протоколы. Есть два вида туннелирования - LocalForwards и RemoteForwards. Рассмотрим первый, используемый чаще.

LocalForwards - туннель с вашей машины на удаленный ssh-сервер. Проще показать работу SSH – туннелирования на примере:

Туннель можно создать из командной строки. добавляя к ssh следующей параметры:

-L local_port : destination_host : destination_port

Где поля определены следующим образом:

-L - указывает на то, что необходимо создать локальный порт

Local_port - номер порт на локальной машине, который будет слушать ssh-демон. Это может быть номер порта или сервисное название например http, pop3, или mysql.

Destination_host - удаленный хост адресата (имя или IP-адрес) с которым происходит ssh-соединение.

Destination_port - порт на удаленной машине, к которому происходит соединение.

Например, если использовать следующую команду

Homebox$ ssh mail.my_isp.net -L 2525:localhost:25 sleep 99999

Тогда, когда соединение с портом 2525 на вашей машине будет установленно , это будет туннелировано на smtp-порт хоста с именем mail.my_isp.net.

Если в другом окне дать команду:

homebox$ telnet -v localhost 2525

homebox [127.0.0.1] 2525 open

220 mail.my_isp.net ESMTP ReegenMail

Видно что подключение к локальному порту 2525 привело к подключению к smtp-порту (25) на удаленном сервере. Так как это подключение было туннелировано внутри существующего ssh сеанса, то оно полностью зашифровано, при этом не потребовалось больших усилий по настройке системы шифрования, и знания криптографических алгоритмов вообще.

Возможно одновременно осуществить многократные подключения, и каждое из них – будет туннелировано через надежно зашифрованный ssh-канал.

Применяя сжатие совместно с туннелированием можно получить существенный выигрышь в скорости HTTP, FTP, POP3 и других соединений.

IV.Примеры реализации протокола.

SSH Secure Shell - коммерческая реализация протокола для UNIX-систем. Так же выпушена версия под Windows.

OpenSSH - бесплатная версия SSH для OpenBSD. Портирован под другие OS. В частности, удалось запустить его под Solaris 2.5 (чего не удалось сделать с оригинальной

версией SSH) и Linux 2.2/2.4. Поддерживает протоколы 1.3, 1.5 и 2.0. При желании поддерживает Entropy Gathering Daemon (egd) или PRNGd, PAM и Gnome passphrase. Для установки требуется zlib 1.1.3 и OpenSSL 0.9.5a/0.9.6b.

Кроме этого сушествует ряд других менее популярных ssh клиентов и серверов под различные плаформы :

win32

- **cigaly:** http://www.doc.ic.ac.uk/~ci2/ssh/
- **f-secure:** http://www.datafellows.com/products/cryptography/f-sshtt.htm
- **secure crt:** http://www.vandyke.com/products/SecureCRT/
- **ttssh:** http://www.zip.com.au/~roca/ttssh.html
- **therapy:** http://guardian.htu.tuwien.ac.at/therapy/ssh/
- **chaffee:** http://bmrc.berkeley.edu/people/chaffee/winntutil.html
- **sergey okhapkin:** http://miracle.geol.msu.ru/sos/
  http://www.lexa.ru/sos/
- **putty:** http://www.chiark.greenend.org.uk/~sgtatham/putty.html
- **fissh:** http://www.massconfusion.com/ssh/

cygnus win32 (http://sourceware.cygnus.com/cygwin/)

- **Client and Server**
  ftp://ftp.franken.de/pub/win32/develop/gnuwin32/cygwin/porters/Mathur_Raju/

beos

- http://www.be.com/beware/Network/ssh.html

wince

- mov: http://www.movsoftware.com/sshce.htm

palm pilot

- **top gun:** http://www.isaac.cs.berkeley.edu/pilot/

java

- **java-applet:** http://www.cl.cam.ac.uk/~fapp2/software/java-ssh/
- **mindterm: client:** http://www.mindbright.se/mindterm
- **mindtunnel: server:** http://www.mindbright.se/mindtunnel.html

os2

- ftp://hobbes.nmsu.edu/pub/os2/apps/internet/telnet/client/sshos203.zip

macintosh

- **niftytelnet+ssh:** http://www.lysator.liu.se/~jonasw/freeware.html
- **f-secure:** http://www.datafellows.com/f-secure/fclintp.htm

openvms

- **server:** http://www.er6.eng.ohio-state.edu/~jonesd/ssh/
- **client:** http://www.free.lp.se/fish/

V. Уязвимость

Теоретически SSH протокол обеспечивыющий очень надежное соединение, не подверженное большинству типов атак, таких как

- "IP spoofing'a"[ip-подмены], когда удаленный[атакующий] компьютер высылает свои пакеты симулируя якобы они пришли с другого компьютера, с которого разрешен доступ. Ssh защищает от подмены даже в локальной сети, когда кто-то например, решил подменить ваш маршрутизатор "собой".
- "IP source routing"[ip исходный маршрутизатор], когда компьютер может симулировать что IP-пакеты приходят от другого, разрешенного компьютера.
- "DNS spoofing", когда атакующий фальсифицирует записи "name server'a"
- Прослушивания нешифрованных паролей и других данных промежуточными компьютерами.
- Манипуляций над вашими данными людьми управляющими промежуточными компьютерами.
- Атак основанных на прослушивании "X authentication data" и подлога соединения к X11 серверу.

Стоит отдельно отметить что SSH1 уязвим к атакам типа “man in the middle”, в отличие от SSH2.

Вопрос безопасности больше относится не к протоколу а программам реализующим SSH. Известные ошибки "безопасности" в различных версиях ssh, OpenSSH можно найти здесь

http://www.openssh.com/ru/security.html

http://faqs.org.ru/softw/inetsoft/ssh_faq.htm

http://www.freebsd.org/cgi/query-pr.cgi?pr=14749

http://www.opennet.ru/opennews/art.shtml?num=1293

http://www.psc.ru/sergey/bgtraq/UNIX/SSH/SSHarhiv.html

Большинство уязвимостей связанно, как обычно, с переполнением буфера.

VII. Список используемой литературы

Официальный сайт производителя Ssh Secure Shell

http://www.ssh.fi/

http://www.atrunet.ru/modules.php?name=News&file=article&sid=22

http://www.hackinglinuxexposed.com/articles/20030228.html

http://www.opennet.ru/docs/HOWTO-RU/mini/Compressed-TCP.html

http://www.opennet.ru/base/sec/lavr-ssh.2.txt.html

http://old.softerra.ru/freeos/18314/page1.html

http://old.softerra.ru/freeos/18337/page1.html

http://www.cisco.com/global/RU/win/solutions/sec/mer_tech_cel-ssh.shtml

http://www.bog.pp.ru/work/ssh.html

http://www.openssh.com/ru/security.html

http://www.openssh.com/ru/manual.html

http://www.psc.ru/sergey/bgtraq/UNIX/SSH/SSHarhiv.html

http://www.opennet.ru/opennews/art.shtml?num=1293

http://www.freebsd.org/cgi/query-pr.cgi?pr=14749

http://www.nevis.columbia.edu/cgi-bin/man.sh?man=ssh

http://faqs.org.ru/softw/inetsoft/ssh_faq.htm