Содержание

openssh

https://habr.com/ru/companies/skillfactory/articles/503466/

ssh-keygen

Программа-генератор RSA/DSA ключей

Синтаксис

```
ssh-keygen [-q] [-b <bits>] [-t <type>] [-N <new_passphrase>] [-C <comment>]
[-f <output_keyfile>]
ssh-keygen -p [-P <old_passphrase>] [-N <new_passphrase>] [-f <keyfile>]
```

Опции

- `-f <~/.ssh/id_srv2>` - указывает имя/путь файла ключа.
- `-t <технология шифрования>` - выбор технологии (`rsa1`, `dsa`).
- `-b <размер ключа в битах>` - задаем размер ключа.
- `-N <_new_passphrase_>` - обеспечивает ввод новой ключевой фразы.
- `-P <passphrase>` - обеспечивает ввод (старой) ключевой фразы.
- `-p` - запрашивает изменение ключевой фразы приватного ключа вместо создания нового приватного ключа. Предложит указать имя файла содержащего приватный ключ, старую ключевую фразу и, дважды, новую ключевую фразу.
- `-C <comment>` - обеспечивает ввод нового комментария.
- `-D <reader>` - копировать публичный RSA-ключ на smartcard в устройстве _reader_.

sshd

`sshd` - это демон, который ожидает соединения от клиентов. Обычно он запускается при загрузке системы из `/etc/rc`. Он создает нового демона для каждого нового соединения. Ответвленный демон обрабатывает обмен ключами, шифрование, аутентификацию, выполнение команд и обмен данными. Эта реализация sshd поддерживает обе версии протокола SSH, 1 и 2, одновременно.

Синтаксис команды:

```
sshd [-deiqtDL46] [-b <битов>] [-f <файл_конфигурации>] [-g
<время_задержки_регистрации>] [-h <файл_ключа_машины>] [-k
<время_генерации_ключа>] [-p <порт>] [-u <величина>] [-V
<идентификатор_протокола_клиента>]
```

Опции

- `-b` <биты> - Определяет число бит в ключе сервера эфемерного протокола версии 1 (по умолчанию 768).
- `-d` - Режим отладки. Сервер посылает отладочную информацию в файл регистрации событий системы и не переходит в фоновый режим работы. Также сервер не создает дочерних процессов и обрабатывает только одно соединение. Этот параметр предназначен только для настройки сервера. Несколько параметров -d, указанных один за другим, повышают уровень отладки. Максимум это 3.
- -e - Если указан этот параметр, sshd направит вывод в консоль вместо механизма регистрации событий системы. - `-f` <файл_конфигурации> - Определяет имя файла конфигурации. По умолчанию это `/etc/openssh/sshd_config`. sshd отказывается работать не имея файла конфигурации.
- `-g` <время_задержки_регистрации> - Позволяет клиенту растянуть время в течении которого клиент должен себя аутентифицировать (по умолчанию 600 секунд). Если клиент не смог аутентифицировать себя в течение этого времени, сервер отключается. Значение равное нулю означает неограниченное время ожидания.
- `-h` <файл_ключа_машины> - Определяет файл, из которого будет считан ключ машины (по умолчанию это `/etc/openssh/ssh_host_key`). Этот параметр должен быть указан, если sshd запущен не от имени суперпользователя. - `-i` - Определяет, что sshd должен быть запущен из inetd.
- `-k` <время_генерации_ключа> - Определяет, как часто будет регенерирован ключ сервера эфемерного протокола версии 1 (по умолчанию 3600 секунд).
- `-p` <порт> - Определяет порт, на котором сервер будет ожидать соединения (по умолчанию 22).
- `-q` - Режим молчания. В системный журнал регистрации событий не будет занесено никакой информации.
- `-t` - Режим тестирования. Только проверяет соответствие файла конфигурации и безопасность ключей.
- `-u` <величина> - Определяет размер полей в структуре utmp хранящей имя удаленной машины.
- `-D` - Когда указан этот параметр, sshd не будет переведен в режим работы демона.
- `-V` <идентификатор_протокола_клиента> - SSH-2 совместимый режим.
- `-4` - Принуждает sshd использовать только IPv4 адреса.
- `-6` - Принуждает sshd использовать только IPv6 адреса.

Конфиг

`sshd` настраивается через `/etc/ssh/sshd.conf` Для применения настроек нужно дёрнуть службу `sshd` Пример:

```
# слушать 22ой порт
Port 22

# использовать вторую версию протокола
Protocol 2

# запрет логинится под root@*
PermitRootLogin no
```

```
# доступ к ssh только для определенного пользователя или группы
AllowUsers User1, User2, User3
AllowGroups Group1, Group2, Group3

# резрешить запуск X11 приложений
X11Forwarding yes
```

Интересный факт: приветствие, выводимое при подключение

ssh

Мощный инструмент, позволяющий как выполнять удаленные команды, так и вертеть трафиком.

Синтаксис команды выглядит так:
`ssh [опции] [имя пользователя@]<сервер> [команда]`

Tip: В `.ssh/config` можно задать псевдонимы в следующем формате:

```
Host radi0ss
    HostName 188.227.87.62
    User root
    Port 22
    ProxyCommand ssh -W %h:%p root@188.227.87.62
```

Опции

Общие параметры
- `-f` - перевести ssh в фоновый режим;
- `-g` - разрешить удалённым машинам обращаться к локальным портам;
- `-l <имя>` - имя пользователя, под которым собираемся подключаться к удаленной машине;
- `-n` - перенаправить стандартный вывод в /dev/null;
- `-N` - указывает, что не запускать удалённый сеанс shell;
- `-p <n>` - указать порт для подключения на удалённой машине;
- `-q` - не показывать сообщения об ошибках;
- `-v` - режим отладки, указанный несколько раз (max 3), сделает ssh более многословным;

Перенаправление X11
- `-X` - включить перенаправление X11;
- `-x` - отключить перенаправление X11;

Перенаправление портов и туннелирование
- `-L [локальный_адрес:]локальный_порт:удаленный_адрес:удаленный_порт` - перенаправление локального порта на порт удаленной машины;
- `-R [удаленный_адрес:]удаленный_порт:локальный_адрес:локальный_порт` - перенаправление удаленного порта с удаленной машины на локальный порт;
- `-D [bind_address:]порт` - создание локального SOCKS-прокси-сервера, перенаправляющего трафик с определенного сокета по зашифрованному каналу на удалённый

сервер;
- `-w local_tun[:remote_tun]` - создает туннель между локальным и удаленным виртуальными сетевыми интерфейсами (TUN или TAP) с номерами `local_tun` и `remote_tun`. Если указать `any` - выбирается любой. Если не указать `remote_tun` - подразумевается `any`. честно говоря, не получалось настроить туннель через tun интерфейсы

Аутентификация и ключи
- `-i <path>` - явно указать приватный ключ для подключения;
- `-A` - включить переадресацию агента

Протоколы и адреса
- `-4` - принудительное использование только IPv4 адресов;
- `-6` - принудительное использование только IPv6 адресов;
- `-1` - принудительное использование SSH протокола только версии 1;
- `-2` - принудительное использование SSH протокола только версии 2;

Дополнительные параметры
- `-C` - включить сжатие данных;
- `-t` - переназначить псевдо-терминал для выполнения программ;
- `-T` - послать на выполнение команду, не подключаясь к shell;

Примеры:

```
# выполнить команду, не заходя в полноценный сеанс
ssh user@host ls

# перенаправление бекапа прямо на удалённый сервер
sudo dd if=/dev/sda | ssh user@host 'dd of=sda.img'
# восстановление бекапа с удаленного сервера
ssh user@host 'dd if=sda.img' | dd of=/dev/sda

# взятие пароля из файла
ssh user@host < local_file.txt

# выполнять на сервере прогу "eclipse" и транслировать её окно нам
ssh -XC user@remotehost "eclipse"

# создание ssh туналя (проброс порта с удалённой машины на локальную)
ssh -L локальный_порт:удаленный_адрес:удаленный_порт пользователь@сервер
```

scp

Утилита для копирования файлов на удаленный сервер.

```
scp /адрес/локального/файла пользователь@хост:адрес/папки
```

Примеры использования:

```
scp ~/test.txt user@host:documents
```

```
# scp непосредственно копирует файл, в то время как передавать файл можно при помощи ssh, но он принимает поток, а не файл.
cat localfile | ssh user@host "cat > remotefile"
# аналогично:
ssh user@host "cat > remotefile" < localfile

# Пойдем еще дальше, вы можете сжимать файлы перед передачей с помощью tar, а потом их сразу же на лету распаковывать:
tar czf - /home/user/file | ssh user@host tar -xvzf -C /home/remoteuser/
```

ssh-agent

По простому:
\- это «связка с ключами». Сначала добавляешь ключи туда через `ssh-add`, а потом эта чудо-связка сама ищет подходящий для сервера, к которому пытаемся подключится, ключ.
Подробнее:
\- это менеджер ключей для SSH, работающий независимо от последнего, запускаясь при первом включении `ssh`. Хранит в памяти ключи, избавляя от необходимости вводить ключевую фразу каждый раз при подключении куда то. Он не записывает ключи на диск и не позволяет экспортировать их.
Вот как проверяется ключ пользователя во время SSH-соединения, с точки зрения сервера:
- Клиент предоставляет серверу публичный ключ.
- Сервер генерирует и отправляет короткое случайное сообщение, прося клиента подписать его с помощью приватного ключа.
- Клиент просит **агента** SSH подписать сообщение и пересылает результат обратно на сервер.
- Сервер проверяет подпись, используя публичный ключ клиента.
- Теперь у сервера есть доказательство того, что клиент владеет приватным ключом.

Позже в процессе соединения генерируется набор новых, эфемерных и симметричных ключей, которые используются для шифрования трафика сеанса SSH. Эти ключи могут даже не длиться весь сеанс; событие «`rekey`» происходит через регулярные промежутки времени.

ssh-add

команда ssh-agent для добавления ключей к «ключнице», которую ssh-agent автоматически использует для ssh. При запуске без параметров ищет и добавляет ключи со стандартными путями:
- `~/.ssh/id_rsa`
- `~/.ssh/id_ed25519`
- `~/.ssh/id_dsa`
- `~/.ssh/id_ecdsa`

переадресация агента

Когда для соединения включена переадресация агента (обычно с использованием `ssh -A`), в фоновом режиме открывается второй канал для переадресации любых запросов агента

обратно на ваш локальный компьютер. Для ssh нет разницы между локальным и удаленным ssh-agent, так как ssh смотрит лишь на $SSH_AUTH_SOCK. При подключении к удаленному хосту с включенной переадресацией агента SSHD создаст удаленный доменный сокет Unix, связанный с каналом переадресации агента, и экспортирует $SSH_AUTH_SOCK, указывающий на него.

> [!warning]
> Когда вы переадресовываете доменный сокет `ssh-agent` Unix на удаленный хост, это создает угрозу безопасности: любой человек с root доступом на удаленном хосте может незаметно получить доступ к вашему локальному SSH-agent'у через сокет. Они могут использовать ваши ключи, чтобы <u>выдавать себя за вас</u> на других машинах в сети.
>
> Совет:
> Заблокируйте свой ssh-агент, когда вы используете переадресацию агента. `ssh-add -x` блокирует агент паролем, а `ssh-add -X` разблокирует его. Когда вы подключены к удаленному хосту с переадресацией агента, никто не сможет проникнуть в ваш агент без пароля.