Администрирование отечественных операционных систем
Практическая работа № 2. Настройка удаленного доступа по SSH

ДИСЦИПЛИНА	Администрирование отечественных операционных систем
ИНСТИТУТ	Институт перспективных технологий и индустриального программирования
КАФЕДРА	Цифровая кафедра
ВИД УЧЕБНОГО МАТЕРИАЛА	Практическая работа
ПРЕПОДАВАТЕЛЬ	Макиевский Станислав Евгеньевич
СЕМЕСТР	1 семестр, 2023-2024

Цель работы: получить навыки работы по следующим направлениям:
- настраивать службу sshd и клиент SSH, работать с SSH;
- использовать SSH для проксирования и туннелирования.

Задание:

0. Теоретическая информация по SSH.

SSH или Secure Shell – это зашифрованный протокол, который часто используется для взаимодействия и удаленного управления серверами. Если вы захотите что-либо сделать на удаленном сервере, скорее всего, вам придется воспользоваться SSH и работать через терминал.

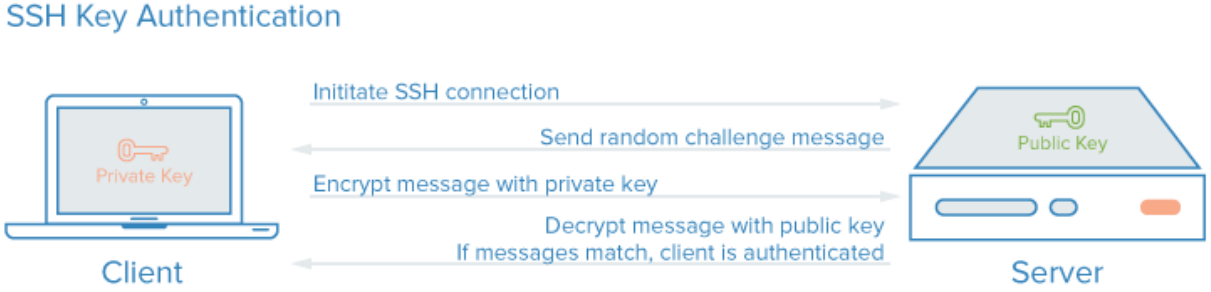
В SSH существует несколько способов авторизации. Вы можете каждый раз вводить пароль пользователя или использовать более безопасный и надежный способ – ключи SSH. Он более удобен для применения, вам даже не нужно будет вводить пароль. В данной практической работе будет рассмотрено, как настраивается авторизация по ключу SSH.

При этом, самый популярный способ аутентификации – это аутентификация по паролю. Он достаточно прост, но не очень безопасный. Пароли передаются по безопасному каналу, но они недостаточно сложны для противостояния попыткам перебора. Вычислительная мощность современных систем в сочетании со специальными скриптами делают перебор очень простым. Конечно, существуют другие способы дополнительной безопасности, но аутентификация по ключу SSH более надежна.

Каждая пара ключей состоит из открытого и закрытого ключа. Секретный ключ сохраняется на стороне клиента и не должен быть доступен кому-либо еще. Утечка ключа позволит злоумышленнику войти на сервер, если не была настроена дополнительная аутентификация по паролю.

Открытый ключ используется для шифрования сообщений, которые можно расшифровать только закрытым ключом. Это свойство и используется для аутентификации с помощью пары ключей. Открытый ключ загружается на удаленный сервер, к которому необходимо получить доступ. Его нужно добавить в специальный файл `~/.ssh/authorized_keys`.

Когда клиент попытается выполнить проверку подлинности через этот ключ, сервер отправит сообщение, зашифрованное с помощью открытого ключа, если клиент сможет его расшифровать и вернуть правильный ответ – аутентификация пройдена.



1. Настройка службы ssh

1.1. Установите службу `ssh` на клиенте и сервере (если она не была установлена ранее).

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

1.2. Сгенерируйте ключи SSH на клиенте и сервере.

`ssh-keygen`

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

Примечание: по умолчанию команда создает пару 2048 битных RSA ключей, которая подойдет не только для SSH, но и для большинства других ситуаций.

Утилита предложит вам выбрать расположение ключей. По умолчанию ключи располагаются в папке `~/.ssh/`. Здесь лучше ничего не менять, чтобы ключи автоматически подгружались из нужных файлов. Секретный ключ будет называться `id_rsa`, а публичный `id_rsa.pub`.

Затем утилита предложит ввести пароль для дополнительного шифрования ключа на диске (его можно не указывать). Использование дополнительного шифрования имеет только один минус – необходимость вводить пароль, и несколько преимуществ:

Пароль никогда не попадет в сеть, он используется только на локальной машине для расшифровки ключа. Это значит, что перебор по паролю больше невозможен.

Секретный ключ хранится в закрытом каталоге и у клиента SSH нет к нему доступа, пока вы не введете пароль.

Если злоумышленник хочет взломать аутентификацию по ключу SSH, ему понадобится доступ к вашей системе. И даже тогда ключевая фраза может стать серьезной помехой на его пути.

Если мы ничего не введем, тогда доступ по ключу SSH будет выполняться автоматически и вам не нужно будет что-либо вводить.

Теперь у нас есть открытый и закрытый ключи SSH, и мы можем использовать их для проверки подлинности.

1.3. Загрузите ключ на сервер.

```
ssh-copy-id username_сервера@ip_адрес_сервера
```

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

Примечание: утилита подключится к удаленному серверу, а затем использует содержимое ключа `id.rsa.pub` для загрузки его на сервер в файл `~/.ssh/authorized_keys`. Дальше вы можете выполнять аутентификацию с помощью этого ключа.

Если такой способ по какой-либо причине для вас не работает, вы можете скопировать ключ по SSH вручную. Здесь создается каталог `~/.ssh`, а затем помещается ключ в файл `authorized_keys` с помощью символа >>, это позволит не перезаписывать существующие ключи:

```
cat ~/.ssh/id_rsa.pub | ssh username@remote_host
"mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys"
```

1.4. Подключитесь к серверу по SSH.

```
ssh username_сервера@ip_адрес_сервера
```

Примечание: если вы не захотели создать SSH ключ с доступом по паролю, то вы сразу же будете авторизованы, что очень удобно. Иначе, сначала придется ввести фразу-пароль для расшифровки ключа.

```
student@prac-work-question:~# Предоставьте ответ в
виде скриншота(-ов), где каждый шаг (действие)
сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может
содержать ссылку на скрипт решения (только git):
```

1.5. Отключение проверки пароля. Если пароль больше не будет использоваться, то для увеличения безопасности системы лучше его вовсе отключить. Сначала нужно убедиться, что ключ надежно сохранен и вы его не потеряете, потому что по паролю вы больше не войдете. Авторизуйтесь на сервере, затем откройте конфигурационный файл

/etc/ssh/sshd_config и найдите там директиву PasswordAuthentication. Нужно установить ее значение в no.

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

1.6. Установите нестандартный порт для SSH (директива Port).

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

1.7. Сохраните файл и перезапустите службу ssh. Дальше будет возможно только подключение по ключу SSH, пароль не будет приниматься.

```
sudo service ssh restart
```

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

2. SSH для проксирования и туннелирования

Переадресация портов SSH может осуществляться по трём типам:

1) Перенаправление локального порта. Соединение перенаправляется с клиентского хоста на SSH-сервер и впоследствии на порт хоста назначения.

2) Перенаправление удаленного порта. Соединение перенаправляется с хоста сервера на клиентский хост и впоследствии на порт хоста назначения.

3) Динамическая переадресация портов. В данном случае создается прокси-сервер SOCKS, который в свою очередь обеспечивает связь через ряд портов.

Предварительно: разрешить TcpForwarding и GatewayPorts. В файле `/etc/ssh/sshd_config` добавить следующую конфигурацию:

```
AllowTcpForwarding yes
GatewayPorts yes
```

2.1. Переадресация локального порта.

Перенаправление порта SSH-клиента (на локальном компьютере), на порт SSH-сервера (удаленного компьютера) и далее на порт, расположенный на конечном компьютере. Тот в свою очередь может являться удаленным SSH-сервером или каким-либо иным компьютером.

Перенаправление локальных портов требуется для подключения к удаленной службе во внутренней сети. Это может быть, к примеру, база данных или системы удалённого доступа. В Linux, MacOS и прочих

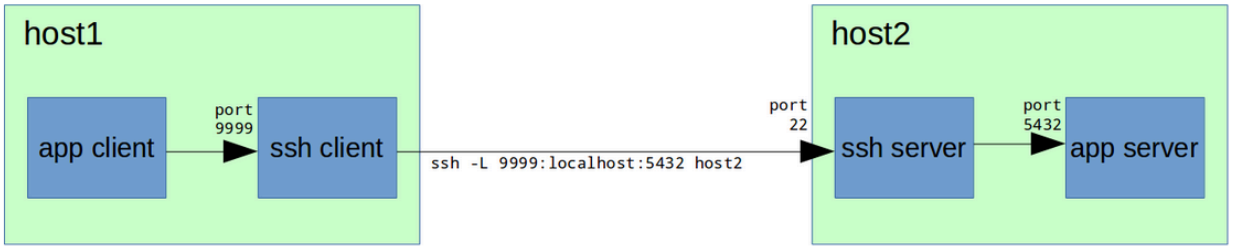
Unix-системах, для установки локальной переадресации портов нужно использовать следующую команду:

```
ssh -L [LOCAL_IP:]LOCAL_PORT:DESTINATION:DESTINATION_PORT [USER@]SSH_SERVER
```

`[LOCAL_IP:]LOCAL_PORT` – IP-адрес и номер порта локального компьютера. Если локальный IP не указан, SSH-клиент привязывается к локальному хосту.

`DESTINATION:DESTINATION_PORT` – IP/имя хоста и порт машины назначения.

`[USER@]SERVER_IP` – удаленный пользователь и IP-адрес сервера.



Примечание: в качестве `LOCAL_PORT` вы можете использовать любой номер порта больше, чем 1024. Значения меньше 1024 являются привилегированными портами и могут использоваться только пользователем `root`. Если SSH-сервер прослушивает не 22-й порт, который стоит по умолчанию, используйте опцию `-p [PORT_NUMBER]`.

Имя хоста назначения должно иметь разрешение с сервера SSH.

Создайте туннель с клиентской машины для доступа к серверу. Используйте локальный порт `2222`, порт сервера – `22`.

```
ssh -L 192.168.1.200:2222:192.168.1.10:80 \
username_сервера@ip_адрес_сервера #создание туннеля с подключением
```

student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.

Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):

2.2. Переадресация удаленного порта.

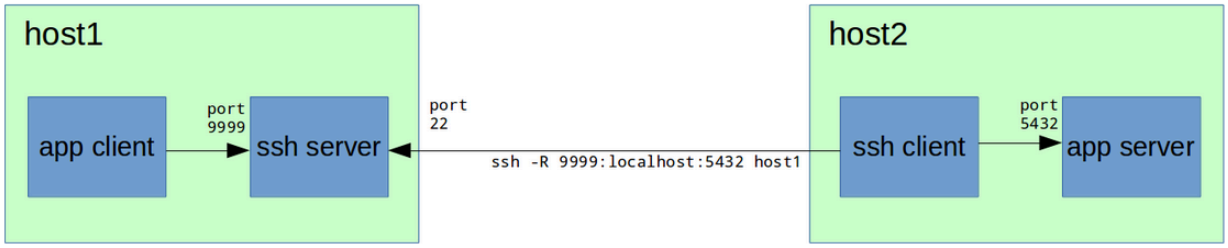
Перенаправление удаленного порта является противоположным случаю с портом локальным. Здесь можно перенаправить порт на удаленном компьютере (SSH-сервер) на порт локального компьютера (SSH-клиент), а затем на порт конечного компьютера. Выполняется с помощью команды:

```
ssh -R [REMOTE:]REMOTE_PORT:DESTINATION:DESTINATION_PORT [USER@]SSH_SERVER
```

`[REMOTE:]REMOTE_PORT` – IP-адрес и номер порта удаленного SSH-сервера. Если значение `REMOTE` не выставлено, удаленный SSH-сервер свяжется сразу со всеми интерфейсами.

`DESTINATION:DESTINATION_PORT` – IP/имя хоста и порт машины назначения.

`[USER@]SERVER_IP` – удаленный пользователь SSH и IP-адрес сервера.



Удаленная переадресация портов в основном используется, чтобы предоставить кому-то извне доступ к одной из внутренних служб.

student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.

Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):

2.3. Динамическая переадресация портов.

Динамическая переадресация портов дает возможность создать сокет на локальном компьютере (SSH-клиент), который выступает в качестве прокси-сервера SOCKS. Когда клиент подсоединяется к данному порту, соединение перенаправляется на удаленный компьютер (SSH-сервер), который далее идет на динамический порт компьютера назначения. Выполняется с помощью команды:

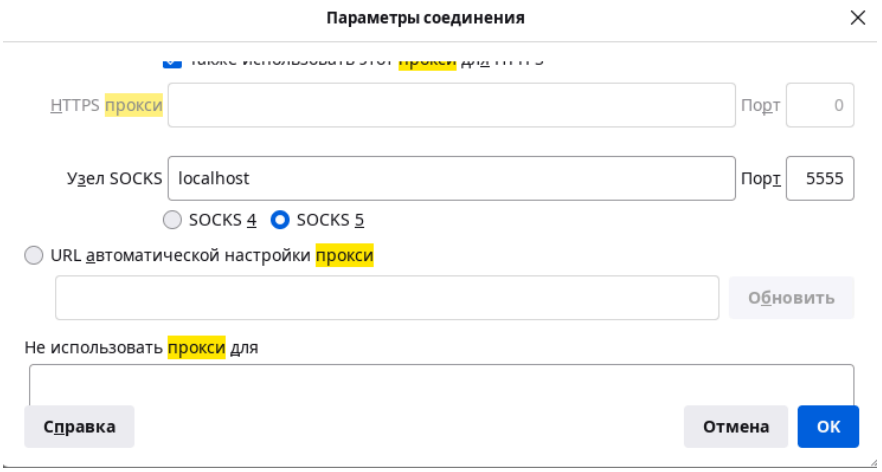
```
ssh -D [LOCAL_IP:]LOCAL_PORT [USER@]SSH_SERVER
```

`[LOCAL_IP:]LOCAL_PORT` – IP-адрес и номер порта локального компьютера. Если `LOCAL_IP` не указан, SSH-клиент привязывается к `localhost`.

`[USER@]SERVER_IP` – Удаленный пользователь SSH и IP-адрес сервера.

Создайте динамическую переадресацию портов, с помощью которой на локальном хосте можно открыть порта прокси-сервера, работающего по протоколам SOCKS4/5. В настройках proxy в операционной системе укажите SOCKS-прокси `127.0.0.1:8080`.

```
ssh -D 8080 -N -f user@192.168.1.10 #создание туннеля без подключения
```



После установки туннелирования можно настроить приложение для его применения. Перенаправление портов нужно настраивать отдельно для каждого приложения, траффик с которого вы хотите перенаправить по SSH-туннелю.

Типовым примером динамической переадресации будет туннелирование трафика браузера через SSH-сервер.

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

2.4. Скопируйте любой файл или директорию с сервера на клиентскую машину.

```
touch file_for_scp
scp file_for_scp username@192.168.1.200
mkdir dir_for_scp && cp file_for_scp dir_for_scp/
scp -r dir_for_scp username@192.168.1.200:~/
```

Примечание: если используется нестандартный номер порта, то после команды `scp` необходимо дописать ключ `-P`.

```
student@prac-work-question:~# Предоставьте ответ в виде скриншота(-ов), где каждый шаг (действие) сопровождается письменным описанием.
   Если необходимо предоставить скрипт, то ответ может содержать ссылку на скрипт решения (только git):
```

Администрирование отечественных операционных систем