МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Московский государственный технический университет
им. Н.Э. Баумана
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра «Информационная безопасность» (ИУ8)

Отчет по лабораторной работе №2

По дисциплине:

«Операционные системы»

По теме: «Сетевой доступ в ОС Linux»

Преподаватель: Климцов В.Е.

Студент: Отдельнов К.Р.

Группа: ИУ8-65

Основная часть

1. Password authentication

Настройка сервера sshd происходит в файле /etc/ssh/sshd_config или же для удобства можно вынести свою конфигурацию в директорию /etc/ssh/sshd_config.d, там как раз и создадим файл ssh01_config.conf .

Ниже будет показана конфигурация нашего сервера:

```
root@lab02-OS:/etc/ssh/sshd_config.d# cat sshd01_config.conf
Port 2142
PermitRootLogin no
PasswordAuthentication yes
PermitEmptyPasswords no
AllowUsers ssh01-user@192.168.100.3
DenyUsers ssh01-user2
LogLevel INFO
root@lab02-OS:/etc/ssh/sshd_config.d# _
```

В этом файле мы задали подключение к серверу по нестандартному порту (**Port 2142**)

PasswordAuthentication yes - вход по паролю на сервер.

PermitEmptyPasswords no - запрещает заходить на сервер через пользователей, у которых не задан пароль.

AllowUsers - указывает пользователей, которые могут заходить на сервер. Также можно указать, чтобы эти пользователи могли получить доступ к серверу только с определенного IP-адресса.

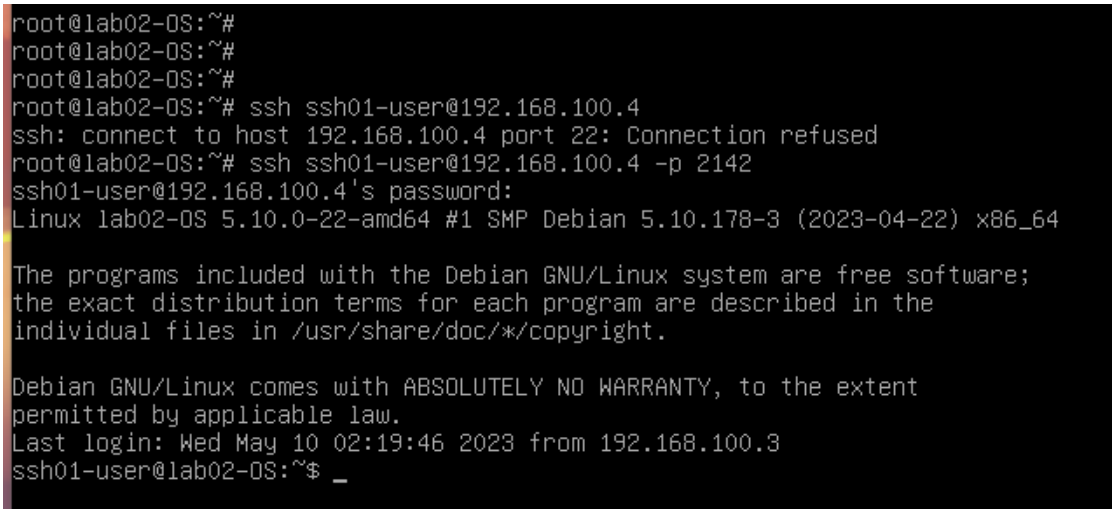
DenyUsers - запрещает подключаться к серверу через указанных пользователей.

PermitRootLogin no - запрещает заходить на сервер под root пользователем. Думаю более правильным было бы создать отдельного пользователя или группу с максимальными привилегиями sudo и заходить уже через него вместо обычного рута.

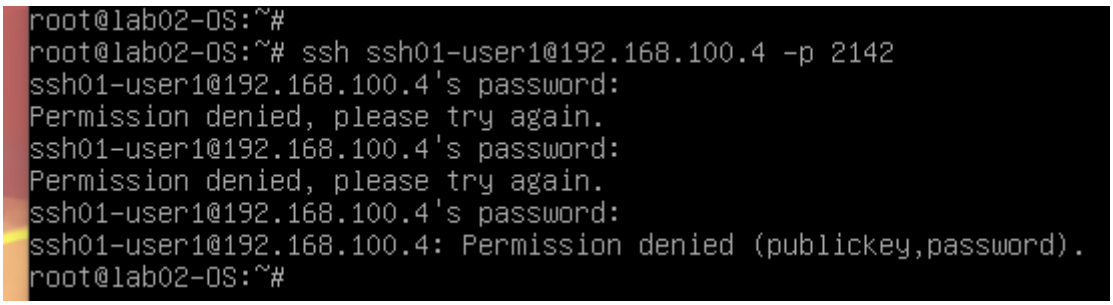
```
root@lab02-OS:/etc/ssh/sshd_config.d#
root@lab02-OS:/etc/ssh/sshd_config.d#
root@lab02-OS:/etc/ssh/sshd_config.d#
root@lab02-OS:/etc/ssh/sshd_config.d# ss -tlpn
State   Recv-Q  Send-Q   Local Address:Port     Peer Address:Port  Process
LISTEN  0       128            0.0.0.0:2142          0.0.0.0:*      users:(("sshd",pid=1511,fd=3))
LISTEN  0       128              [::]:2142             [::]:*      users:(("sshd",pid=1511,fd=4))
root@lab02-OS:/etc/ssh/sshd_config.d#
```

Ниже продемонстрировано подключение через пользователя ssh01-user и ssh01-user1.

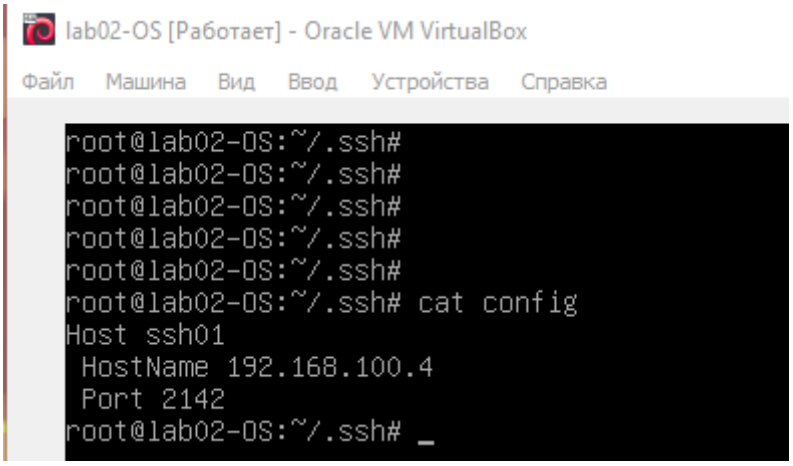
Теперь, чтобы подключиться по порту, который мы указывали в файле ssh01_config.conf нужно явно указывать порт «-p 2142», иначе подключение будет осуществляться по стандартному порту, который имеет значение 22.



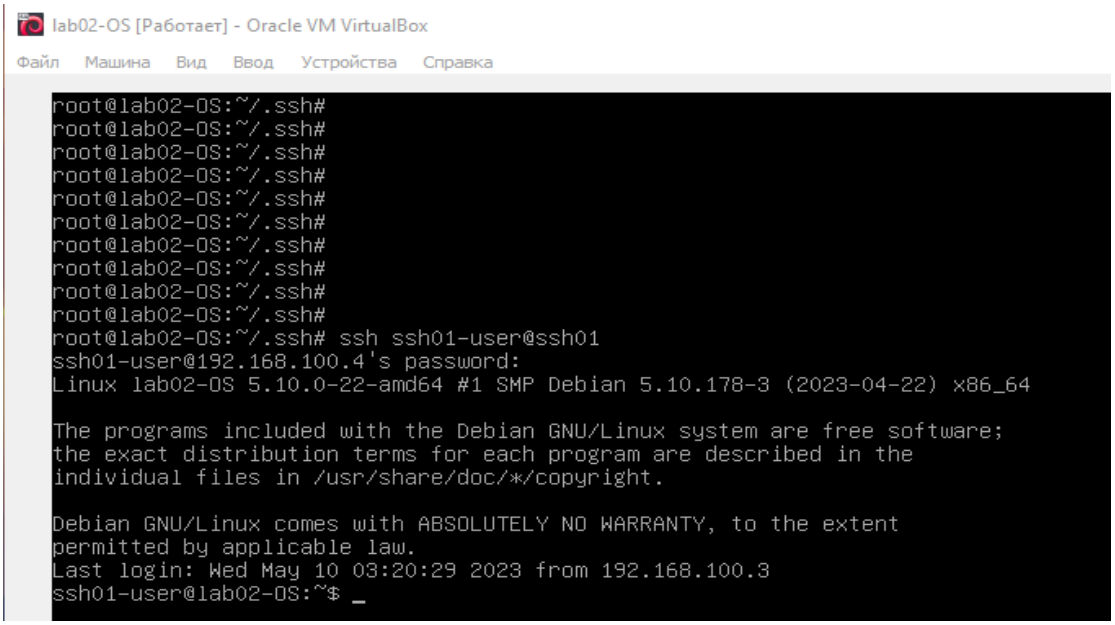
Под другим же пользователем, который указан в DenyUsers зайти, очевидно, не получится.



Настройка клиента происходит в файле ~/.ssh/config.

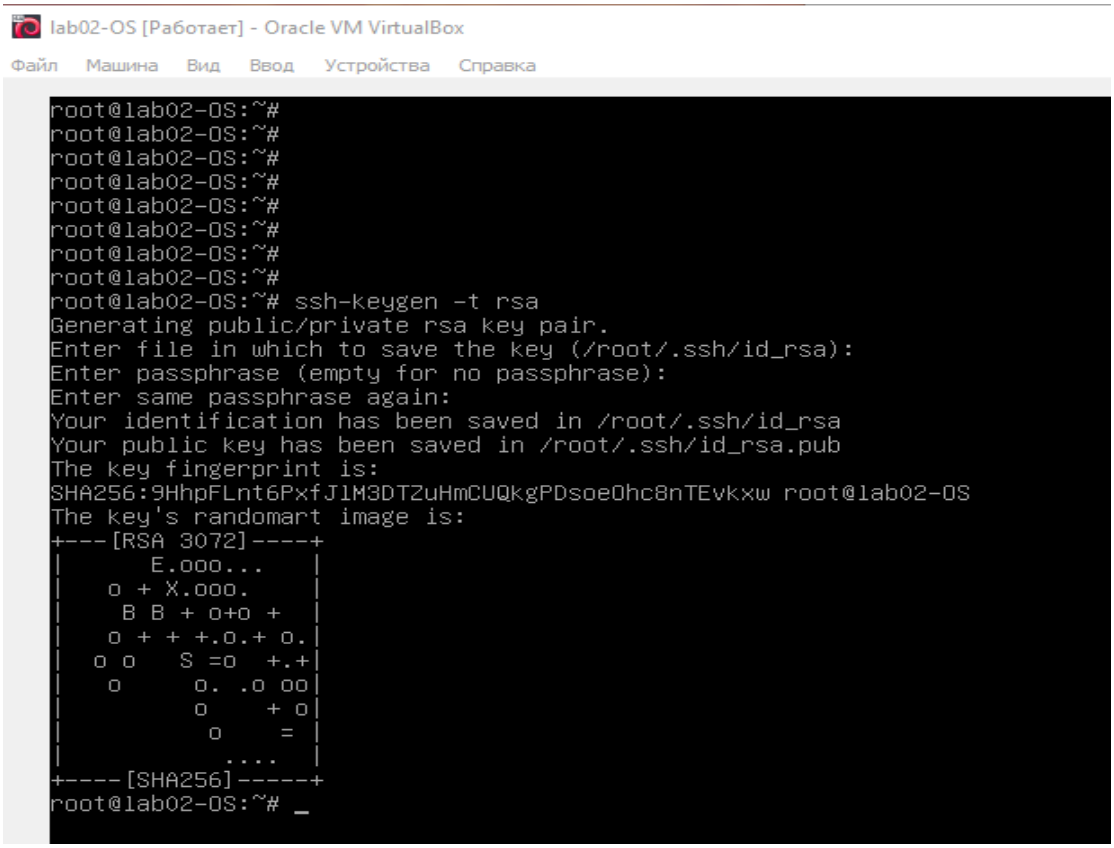


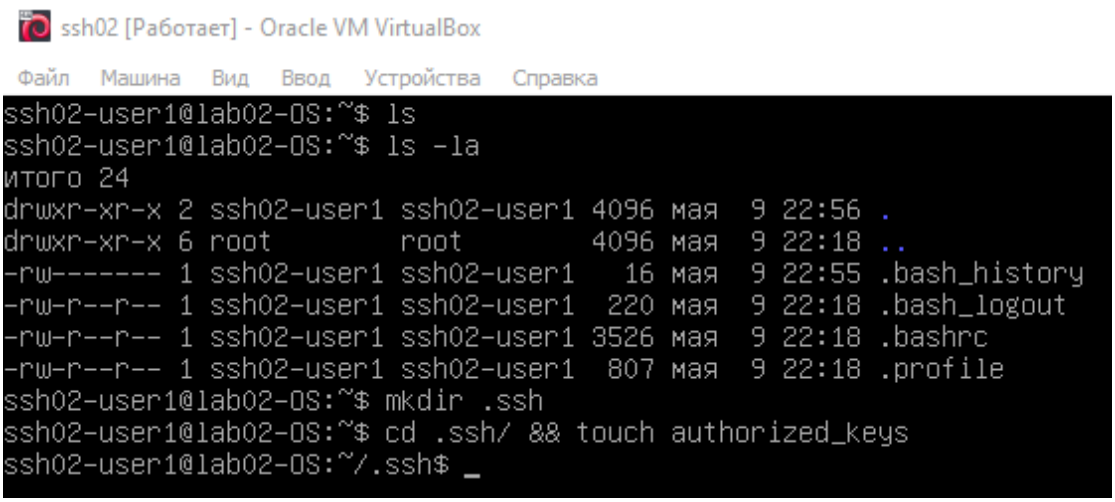
Теперь, чтобы каждый раз не вводить IP-адресс и порт куда мы хотим подключиться достаточно просто указать имя хоста для подключения.



2. Public key authentication

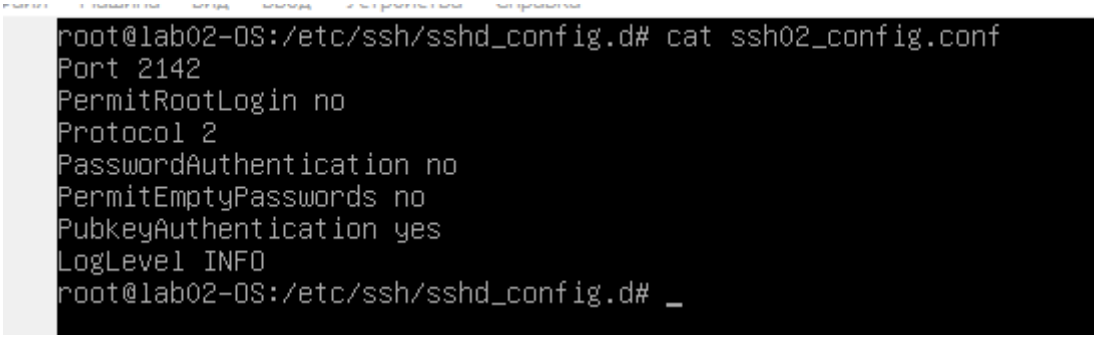
Для осуществления этого метода авторизации нужно сгенерировать ключ на клиенте с помощью команды ssh-keygen -t rsa, где с помощью параметра -t указываем тип ключа (rsa).





Теперь нужно передать публичный ключ id_rsa.pub на удаленный хост с помощью следующей команды (или же есть специальная утилита ssh-copy-id).

Настройка конфигурационных файлов на сервере

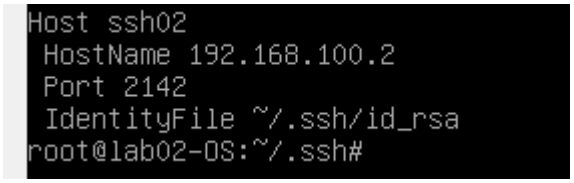


Protocol 2 - указываем, что можно использовать только вторую версию протокола ssh (т.к. старая версия протокола может иметь известные уязвимости).

PasswordAuthentication no - Т.к. мы пробросили ключи, то можно отключить авторизацию по паролю, для обеспечения большей безопасности.

PubkeyAuthentication yes - разрешает авторизацию по ключам.

Настройка клиента, от которого будем подключаться к нашему серверу:



IdentityFile ~/.ssh/id_rsa - указывает путь до приватных ключей.

Проверка подключения:

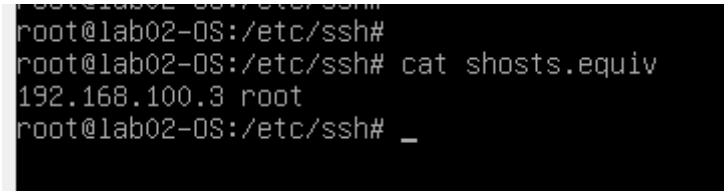
3. Host based authentication

Для начала проведем настройку сервера в отдельном файле /etc/ssh/sshd_config.d/ssh03_config.conf

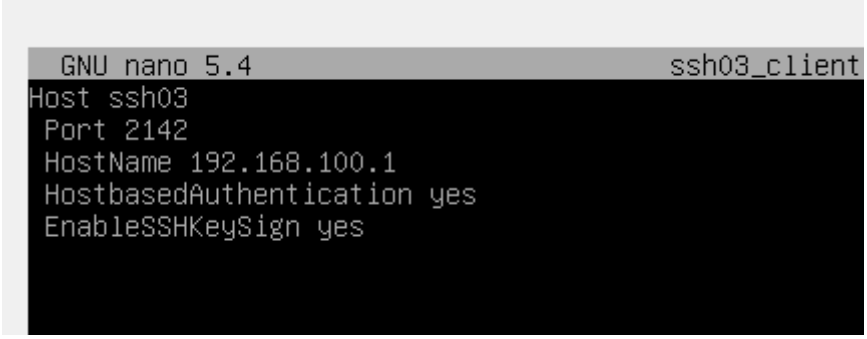
Отключаем авторизацию по ключам и по паролю и оставляем только авторизацию по хосту.

Далее сканируем ключи с клиента на сервер, с помощью утилиты ssh-keyscan, и записываем их в файл /etc/ssh/ssh_known_hosts.

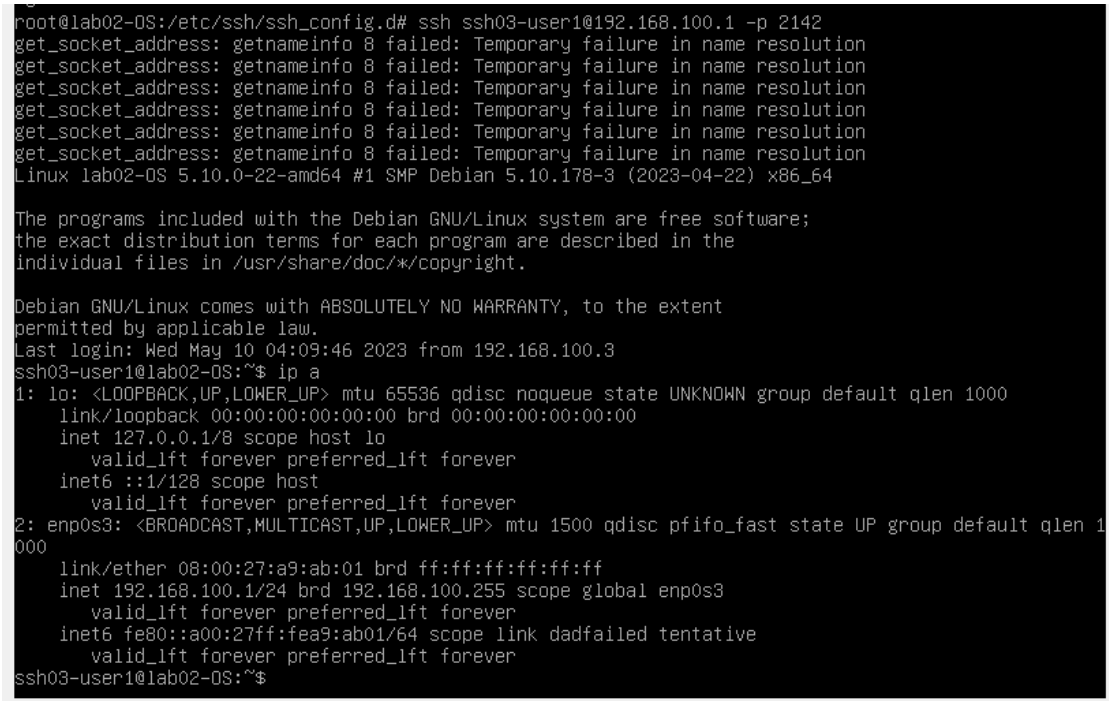
Затем создаем и настраиваем файл /etc/ssh/shosts.equiv, который содержит список клиентских систем, которым разрешена аутентификация на основе хоста.



И последним шагом в настройке hostbased подключения будет изменения файла /etc/ssh/ssh_config, в который нужно также прописать разрешение авторизации на основе хоста, а также настроить включение **EnableSSHKeySign yes** (который по умолчанию выключен), который нужен для разрешения доступа к ключам клиента.



Проверка подключения


root@lab02-OS:/etc/ssh/ssh_config.d# ssh ssh03-user1@192.168.100.1 -p 2142
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
get_socket_address: getnameinfo 8 failed: Temporary failure in name resolution
Linux lab02-OS 5.10.0-22-amd64 #1 SMP Debian 5.10.178-3 (2023-04-22) x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed May 10 04:09:46 2023 from 192.168.100.3
ssh03-user1@lab02-OS:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
000
link/ether 08:00:27:a9:ab:01 brd ff:ff:ff:ff:ff:ff
inet 192.168.100.1/24 brd 192.168.100.255 scope global enp0s3
valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:fea9:ab01/64 scope link dadfailed tentative
valid_lft forever preferred_lft forever
ssh03-user1@lab02-OS:~$

Вывод

В ходе выполнения лабораторной работы были получены навыки в конфигурировании удаленного доступа к серверу. Были разобраны различные методы авторизации: по паролю, по ключу и по хосту.

1. Авторизация по паролю - т.к. пароли могут быть украдены или подобраны, то этот способ подключения не обеспечивает надежную защиту против взлома, не смотря на то, что существуют инструменты, которые могут усилить безопасность (например fail2ban).

2. Авторизация по ключу - более надежная и безопасная альтернатива. Пары ключей SSH представляют собой два защищенных шифрованием ключа, которые можно использовать для аутентификации клиента на сервере SSH. Каждая пара ключей состоит из открытого ключа и закрытого ключа. Закрытый ключ хранится клиентом и должен быть абсолютно защищен. Любое нарушение безопасности закрытого ключа позволит злоумышленникам входить на серверы с соответствующим открытым ключом без дополнительной аутентификации. В качестве дополнительной меры предосторожности ключ можно зашифровать на диске с помощью парольной фразы.

3. Авторизация по хосту - этот метод авторизации не сильно отличается от аутентификации с открытым ключом, и клиент использует пару ключей для аутентификации себя на сервере. Однако подключение должно осуществляться с устройства, указанного в списке разрешенных хостов на сервере, поэтому авторизация по ключу хоста подразумевает доверие хосту, тем самым задача авторизации переносится на хост клиента. За хостом клиента могут выступать устройства, которым не следует

безукоризненно доверять, а значит авторизация по ключу хоста для таких устройств не является безопасной.