

ЛАБОРАТОРНАЯ РАБОТА №11

НАСТРОЙКА СЛУЖБЫ SSH

НАСТРОЙКА СЛУЖБЫ SSH

Цель работы: Научиться устанавливать и настраивать серверную и клиентскую части службы SSH в различных конфигурациях, в соответствии с требованиями безопасности. А также организовывать безопасный вход в систему используя ключи.

Задание:

1. Проверьте, установлены ли в системе какие либо пакеты openssh:

```
$ aptitude search openssh
```

2. Если серверная или клиентская часть не установлены, установите их

```
# aptitude install openssh-client
```

```
# aptitude install openssh-server
```

3. Проверьте, запущен ли sshd:

```
# ps ax | grep sshd
```

4. Если не запущен, запустите его

```
# service ssh start
```

5. Проверьте состояние сервера ssh:

НАСТРОЙКА СЛУЖБЫ SSH

/etc/init.d/ssh status

При подключении по SSH используются три вида аутентификации: аутентификация по паролю, по ключу хоста и по открытому ключу. Аутентификация по паролю в чистом виде, является наименее безопасной. Рассмотрим разницу между аутентификацией по ключу хоста и аутентификацией по открытому ключу. При первом подключении с использованием аутентификации по ключу хоста открытый ключ хоста (сервера) копируется на компьютер-клиент в профиль пользователя, инициировавшего удалённое подключение, в файл `/.ssh/known_hosts`. При последующих подключениях копия открытого ключа на клиенте сравнивается с открытым ключом на сервере и далее следует запрос пароля для пользователя. При использовании аутентификации по открытому ключу, пара ключей генерируется не на хосте (сервере), а на клиенте и полностью его идентифицирует. Открытый ключ шифруется и копируется на сервер. При подключении происходит сравнение копии открытого ключа и оригинала и если ключи совпали, происходит вход на удалённый компьютер.

НАСТРОЙКА СЛУЖБЫ SSH

6. Осуществить подключение к серверу SSH можно с помощью команды `ssh` :

```
$ ssh 192.168.2.131
```

```
$ ssh -l user 192.168.2.231
```

```
$ ssh user@example.com
```

При первом подключении будет выдан следующий запрос:

The authenticity of host 'example (192.168.2.231)' can't be established.

RSA key fingerprint is ee:ba:b8:48:ef:96:93:a8:3d:7b:3f:fa:85:8c:a7:ad.

Are you sure you want to continue connecting (yes/no)?

Предлагается принять открытый ключ удалённого сервера `ssh` и продолжить подключение. Для обеспечения безопасности `ssh` генерирует пару ключей: открытый и закрытый. Закрытый ключ никому не показывается, открытыми ключами компьютеры обмениваются между собой.

Соглашаемся. Получаем сообщение о добавлении сервера **example.com192.168.2.231** в список известных хостов и приглашение для ввода пароля пользователя. Вводим пароль и получаем доступ к удалённому компьютеру. В результате, открытый ключ удалённого сервера

НАСТРОЙКА СЛУЖБЫ SSH

`example.com` был скопирован на локальный компьютер в файл `known_hosts`, в директорию `~/.ssh` (домашнюю директорию пользователя). Теперь при последующих подключениях к серверу `example.com` не будут выдаваться предупреждения, будет выводиться только запрос пароля.

Настройка SSH сервера

Настройки `sshd` находятся в файле `/etc/ssh/sshd_config`.

10. Откройте этот файл для редактирования и отредактируйте его:

Port-по умолчанию используется 22 порт. Изменим его на нестандартный порт 2203 -это избавит сервер от сетевых роботов, которые автоматически сканируют интернет в поиске открытых портов и пытаются через них подключиться.

Port 2203

11. Перезапустите службу `ssh` и попробуйте подключиться к серверу (нужно явно указать порт).

```
$ ssh -l user -p 2203 192.168.2.231
```

или

НАСТРОЙКА СЛУЖБЫ SSH

```
$ ssh user@example.com-p 2203
```

12. По умолчанию сервер «слушает» (принимает подключения) на всех сетевых интерфейсах. Если это устраивает, оставьте значение по умолчанию.

```
#ListenAddress ::
```

```
#ListenAddress 0.0.0.0
```

Если нужно оставить подключение только через внешний интерфейс, то раскомментировав строку укажите:

```
ListenAddress 192.168.2.0
```

13. Следующий параметр отвечает за версию протокола SSH. Значение по умолчанию 2. Не меняйте на первую версию -она не безопасна

```
Protocol 2
```

14. Строки HostKey необходимы для второй версии протокола SSH и отвечают за названия файлов ключей и их расположение. Первая строка отвечает за пару ключей RSA, вторая за пару ключей DSA. К названиям открытых (публичных) ключей добавляется .pub. Эти ключи

НАСТРОЙКА СЛУЖБЫ SSH

используются при аутентификации с ключом хоста. Отключите ключи DSA, будем пользоваться RSA:

```
HostKey /etc/ssh/ssh_host_rsa_key
```

```
#HostKey /etc/ssh/ssh_sunup_dsa_key
```

15. Privilege Separation указывает, должен ли демон sshd разделять привилегии. Если да - сначала будет создан непривилегированный дочерний процесс для входящего сетевого трафика. После успешной авторизации будет создан другой процесс с привилегиями вошедшего пользователя. Основная цель разделения привилегий - предотвращение превышения прав доступа. Оставляем yes.

```
UsePrivilegeSeparation yes
```

16. Следующие строки отвечают за временный ключ и его длину при работе с первой версией протокола SSH. Не используем SSH-1, поэтому закомментируем строки.

```
# KeyRegenerationInterval 3600
```

НАСТРОЙКА СЛУЖБЫ SSH

ServerKeyBits 768

17. Группа параметров, отвечающая за журналирование. События, связанные с доступом по SSH записываются в `/var/log/auth`. Первый параметр определяет, список событий в журнале. Доступны значения: DAEMON, USER, AUTH, LOCAL0, LOCAL1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7. Нас интересует авторизация, поэтому оставляем AUTH.

SyslogFacility AUTH

18. Второй параметр определяет уровень детализации событий. Доступны: SILENT, QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG, DEBUG1, DEBUG2, DEBUG3. Оставляем уровень детализации по умолчанию.

LogLevel INFO

19. Следующая группа параметров относится к аутентификации. Первый означает, что соединение будет разорвано через указанное количество секунд, если пользователь не войдёт в систему. Уменьшим это время в два раза.

НАСТРОЙКА СЛУЖБЫ SSH

LoginGraceTime 60

20. Второй параметр разрешает/запрещает вход под root-ом. Запрещаем вход суперпользователю.

PermitRootLogin no

21. Третий параметр включает проверку демоном ssh прав и владение домашним каталогом пользователя, который пытается получить удалённый доступ к компьютеру. Оставляем yes.

StrictModes yes

22. Добавляем параметр AllowUsers, которого нет в конфигурационном файле по умолчанию. Этот параметр разрешает доступ к серверу по протоколу SSH только для перечисленных пользователей.

AllowUsers student

НАСТРОЙКА СЛУЖБЫ SSH

В примере разрешение только у пользователя student. Значениями параметра могут быть имена пользователей, отделённые друг от друга пробелами. Можно использовать запись пользователя в виде student@example.com-доступ разрешён пользователю student с компьютера example.com. Существует параметр -DenyUsers, который запрещает доступ пользователям, перечисленным в значении этого параметра. Кроме параметров связанных с доступом отдельных пользователей существуют соответствующие параметры для групп: AllowGroups и DenyGroups.

23. Оставляем включенной аутентификацию RSA, в параметре RSAAuthentication:
RSAAuthentication yes

24. Оставляем включенной аутентификацию по открытому ключу, в дальнейшем настроим аутентификацию таким способом.
PubkeyAuthentication yes

НАСТРОЙКА СЛУЖБЫ SSH

25. Параметр `AuthorizedKeysFile` определяет файл, в котором содержатся публичные ключи, используемые для аутентификации пользователей по открытому ключу. В записи могут присутствовать переменные, например `%h` означает домашний каталог пользователя, а `%u` - имя пользователя. В дальнейшем мы планируем использование аутентификации по открытому ключу -раскомментируем эту строку.

```
AuthorizedKeysFile .ssh/authorized_keys
```

26. Следующие два параметра отвечают за включение совместимости с программой `rhosts`. Нам такая совместимость только повредит, поэтому оставляем значения по умолчанию.

```
IgnoreRhosts yes
```

```
RhostsRSAAuthentication no
```

27. Аутентификация `hostbased` не нужна и она уже отключена, поэтому оставляем существующее значение

```
HostbasedAuthentication no
```

28. Следующая строка нужна для совместимости с `rhost` -оставляем закомментированной.

НАСТРОЙКА СЛУЖБЫ SSH

#IgnoreUserKnownHosts yes

29. Параметр PermitEmptyPasswords разрешает или запрещает вход с пустым паролем. Естественно, запрещаем вход с пустым паролем -оставляем no.

PermitEmptyPasswords no

30. ChallengeResponseAuthentication включает PAM интерфейс. Если yes, для всех типов аутентификации помимо обработки модуля сессии и аккаунта PAM, будет использоваться аутентификация на основе запроса-ответа (ChallengeResponseAuthentication и PasswordAuthentication) Т.к. аутентификация запросов-ответов в PAM обычно выполняет ту же роль, что и парольная аутентификация, следует отключить, либо PasswordAuthentication, либо ChallengeResponseAuthentication.

ChallengeResponseAuthentication no

31. Следующий параметр отвечает за аутентификацию по паролю. Сейчас используется аутентификация по ключу хоста -просто раскомментируем строку.

PasswordAuthentication yes

НАСТРОЙКА СЛУЖБЫ SSH

32. Группу из четырёх параметров, отвечающих за аутентификацию Kerberos, оставляем без изменений -не раскомментированными.

```
#KerberosAuthentication no
```

```
#KerberosGetAFSToken no
```

```
#KerberosOrLocalPasswd yes
```

```
#KerberosTicketCleanup yes
```

33. Следующие два закомментированных параметра отвечают за то, разрешена ли аутентификация пользователя на основе GSSAPI или нет. обычно GSSAPI не нужна -оставляем без изменений.

```
#GSSAPIAuthentication no
```

```
#GSSAPICleanupCredentials yes
```

34. Параметры, начинающиеся с X11, отвечают за проброс иксов через ssh туннель. Если сервер не имеет иксов, комментируем эти опции

```
# X11Forwarding yes
```


НАСТРОЙКА СЛУЖБЫ SSH

```
# X11DisplayOffset 10
```

35. Опция PrintMotd выводит при подключении к sshd так называемое сообщение дня, что на самом деле является содержимым файла /etc/motd. Опция PrintLastLog очень полезна, так как она включает отображение информации о том, когда вы последний раз и с какого компьютера заходили на сервер.

```
PrintMotd no
```

```
PrintLastLog yes
```

36. Установим параметру TCPKeepAlive значение no. Этот параметр важен для поддержания соединения со стороны сервера, но мы реализуем те же функции, но более безопасней.

```
TCPKeepAlive no
```

Для этого добавим два следующих параметра:

```
ClientAliveCountMax 3
```

```
ClientAliveInterval 20
```

НАСТРОЙКА СЛУЖБЫ SSH

Первый из параметров определяет количество запросов, которое ssh-сервер отправляет ssh-клиентам через определённые промежутки времени. Как только установленное значение запросов достигнуто, а клиент так и не ответил, соединение будет разорвано. Если не посылать такие запросы, то сессии на сервере придётся закрывать вручную, так как они будут длиться бесконечно, отбирая часть ресурсов. Второй параметр определяет интервал отправки запросов в секундах. В нашем примере, соединение будет разорвано, если между клиентом и сервером не будет связи в течение одной минуты.

37. Параметр `UseLogin` оставляем закомментированным. Его значение по умолчанию и так `no`.
`#UseLogin no`

38. Раскомментируем параметр `MaxStartups` и выставим ему следующее значение:
`MaxStartups 3:30:9`

НАСТРОЙКА СЛУЖБЫ SSH

По умолчанию значение параметра 10 -количество неавторизованных подключений к серверу ssh. Длительность такого подключения определяется параметром LoginGraceTime. Перенесём параметр MaxStartups под LoginGraceTime и запишем в виде start:rate:full, где start - имеющееся количество неавторизованных подключений, rate -процент вероятности отклонения попытки подключения, full -максимальное количество неавторизованных соединений. Т.е. если имеется 3 неавторизованных подключения, то следующее подключение будет отклонено с вероятностью 30%, а если количество неавторизованных подключений достигнет 9, все последующие попытки подключения -отвергнуты.

39. Опция Banner определяет место положения файла-баннера, который будет выведен на экран, при попытке подключиться к sshd. Предлагается файл /etc/issue.net, но можно использовать свой баннер, поместив его, например в /etc/ssh. По умолчанию выводится содержимое файла /etc/motd.tail.

```
#Banner /etc/issue.net
```

НАСТРОЙКА СЛУЖБЫ SSH

40. Параметр `AcceptEnv` указывает, какие переменные окружения, переданные клиентом, будут приняты.

`AcceptEnv LANG LC_*`

41. Следующий параметр включает внешнюю подсистему (например, FTP). В качестве параметров понимает, имя подсистемы и команду, которая будет выполнена при запросе подсистемы. Команда `sftp-server`, реализует протокол передачи файлов через SSH -SFTP.

`Subsystem sftp /usr/lib/openssh/sftp-server`

42. Параметр `UsePAM` оставляем без изменений. Если директива `UsePAM` включена, то запустить `sshd` можно будет только от имени `root`.

`UsePAM yes`

НАСТРОЙКА СЛУЖБЫ SSH

43. Сохраняем изменения и перезапускаем `sshd`. Подключаемся, проверяем, если всё в порядке, то настройка сервера `ssh` с аутентификацией по ключу хоста закончена. Для разрешения проблем и вывода отладочной информации можно подключаться с ключами: `-v`, `-vv`, `-vvv`

Настройка SSH клиента

1. Глобальные клиентские настройки находятся в файле `/etc/ssh/ssh_config` и применяются ко всем пользователям.

Пользовательские настройки могут находиться в домашнем каталоге пользователя, в `~/.ssh/config` и применяются к одному пользователю. Файл пользовательских настроек не создаётся автоматически в отличие от файла глобальных настроек клиентской части `ssh`.

НАСТРОЙКА СЛУЖБЫ SSH

2. Параметр **Host**. Ограничивает множество хостов, к которым применяются последующие (до ближайшей новой директивы Host) директивы, по указанным шаблонам (хост должен соответствовать хотя бы одному шаблону). Шаблон, состоящий из одного символа *, соответствует любому хосту. Под хостом в данном контексте понимается аргумент имя_хоста передаваемый в командной строке (т.е. никаких преобразований перед сравнением не выполняется).
3. Параметр **HostName**. Устанавливает соответствие между псевдонимами, сокращениями и настоящими именами хостов. По умолчанию используется имя, передаваемое в командной строке. Допустимо непосредственное указание IP-адресов.
4. Параметр **Port**. Порт на удалённой машине, к которому следует подключаться. По умолчанию -22
5. Параметр **User**. Имя пользователя, которое следует использовать при регистрации в удалённой системе.

НАСТРОЙКА СЛУЖБЫ SSH

6. В качестве примера создадим файл пользовательских настроек `/home/selifan/.ssh/config` следующего содержания:

```
Host example
```

```
HostName example.com
```

```
Port 2203
```

```
User student
```

```
Host host1
```

```
HostName host1.example.com
```

```
Port 2280
```

```
User user2
```

```
Host 212.177.65.1
```

```
HostName 212.177.65.1
```

```
Port 2222
```

```
User user5
```

НАСТРОЙКА СЛУЖБЫ SSH

Теперь при подключении к перечисленным компьютерам не нужно вспоминать имя пользователя или ни ssh порт подключения, достаточно после ssh набрать имя сервера.

Генерация ключей

1. При подключении с использованием аутентификации с ключом хоста открытый ключ сервера копируется на компьютер-клиент. На сервере ключи лежат в директории `/etc/ssh`. При установке OpenSSH создаются две пары ключей RSA и DSA, которые представляют собой отдельные файлы: `ssh_host_rsa_key`, `ssh_host_rsa_key.pub`, `ssh_host_dsa_key` и `ssh_host_dsa_key.pub`. Файлы, которые оканчиваются на `.pub` являются открытыми (публичными) ключами, а те, что оканчиваются на `key` - закрытыми. В процессе администрирования сервера может понадобится сгенерировать новые ключи. Рассмотрим генерацию ключей на следующем примере. Изменим параметр `HostKey`:

```
HostKey /etc/ssh/ssh_sunup_rsa_key
```

2. Удалим старые ключи:

НАСТРОЙКА СЛУЖБЫ SSH

```
# rm /etc/ssh/ssh_host*
```

3. Сгенерируем новую пару RSA ключей, для чего выполним следующую команду:

```
# ssh-keygen -t rsa -f /etc/ssh/ssh_sunup_rsa_key
```

В запросе о пароле оставляем поля пустыми. В результате будет создана новая пара RSA ключей с длиной шифрования 2048 бит.

4. Перезапускаем sshd:

```
# /etc/init.d/ssh restart
```

5. Теперь, если мы попробуем подключиться к серверу со старым ключом, то получим следующее предупреждение:

НАСТРОЙКА СЛУЖБЫ SSH

```
WARNING: HOST IDENTIFICATION HAS CHANGED! @
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
2b:f3:c6:42:29:f2:26:6c:6f:28:dc:16:39:9f:bb:e7.
Please contact your system administrator.
....
Host key verification failed.
```

6. При попытке подключения не совпали слепки ключей (fingerprint). Решение проблемы содержится в сообщении - нужно просто удалить указанную строку, в нашем случае 3, в файле `known_hosts`, снова подключиться и принять новый ключ сервера.

```
$ vi ~/.ssh/known_hosts
```

Подключение с использованием открытого ключа

1. Для подключения с авторизацией по открытому ключу нужно сгенерировать секретный ключ на стороне клиента. Делаем это с правами обычного пользователя:

НАСТРОЙКА СЛУЖБЫ SSH

\$ ssh-keygen -t rsa

В процессе генерации пары ключей сначала будет предложено ввести желаемое название файла ключа, а так же ввести и подтвердить пароль. Вместо имени файла нажимаем Enter - будут созданы файлы ключей с именами по умолчанию. Секретную фразу не вводим, особенно если планируем удалённо запускать скрипты. В противном случае нужно будет каждый раз вводить секретную фразу или работать с помощью **ssh-agent**. Пароль для файла ключа может оказаться полезным в случае попадания ключа в чужие руки, но такие вещи как секретные ключи надо беречь, даже если они защищены хорошим паролем. Файлы закрытых ключей должны быть недоступны для чтения/записи всем кроме их владельца (режим 600). Открытые ключи для чтения должны быть доступны всем, но право на запись должен иметь только владелец (режим 644). В идеале лучше хранить ключи на Token.

2. И так, были сгенерированы два RSA ключа с именами **id_rsa** и **id_rsa.pub**. Теперь нужно скопировать на сервер открытый ключ в директорию, указанную в файле

НАСТРОЙКА СЛУЖБЫ SSH

`/etc/ssh/sshd_config`, в параметре `AuthorizedKeysFile`, но сначала на сервере в домашней директории нужно создать скрытую поддиректорию `.ssh`, если её нет:

```
$ mkdir ~/.ssh
```

3. Теперь с клиента копируем сгенерированные ключи на сервер, воспользовавшись утилитой `scp`, входящей в пакет `OpenSSH`:

```
$ scp -P 2203 ~/.ssh/id_rsa.pub student@example.com:~/.ssh/authorized_keys
```

4. Совсем отключаем аутентификацию по паролю. Для этого на сервере в файле `/etc/ssh/sshd_config` меняем параметр `PasswordAuthentication`:

```
PasswordAuthentication no
```

5. Подключаемся к серверу:

```
$ ssh student@example.com -p 2203
```

6. Теперь можно копировать открытый ключ на сервер