Linux

- Вход по ключу в SSH
- Запрет доступа с УЗ root и служебные УЗ
- Настройка fail2ban (защита от брутфорса)
- Разрешение доступа по гео с iptables
- Команды по Linux Hardening (Харденинг)
- Блокировка источников динамическим листом IP с UFW

???? ?? ????? ? SSH

Помимо использования SSH на не стандартном порту (не панацея), лучше усилить защиту используя вход по ключу. Пару ключей SSH сложнее взломать по сравнению с обычным паролем. Содержимое ключей генерируется с использованием алгоритмов, что затрудняет его перебор. Доступ может получить только машина, на которой находится закрытый ключ.

Аутентификация с открытым ключом никогда не показывает серверу содержимое закрытого ключа. В случае компрометации сервера локальная машина останется в безопасности.

Тк в течении 2023 года появлялись научные статьи о возможной компрометации алгоритма RSA, для генерации ключа мы будем использовать алгорим использующий эллиптические кривые. На машине Linux (Хост А или Б) выполните следующую команду:

```
ssh-keygen -t ecdsa -b 521
```

Далее нажимаем просто клавишу Enter. Либо можно задать имя для ключевой пары или добавить пароль.

> Добавление пароля к закрытому ключу добавляет многофакторную аутентификацию.

Выполняем действия на машине на которую хотим заходить по SSH (Хост Б), в корневой директории пользователя если нет папки `.ssh`, то ее необходимо создать:

```
cd ~
mkdir .ssh
```

Копируем содержимое публичного ключа (`my_key.pub`) и добавляем его содержимое в файл домашней директории пользователя root `authorized_keys` в папке `.ssh` пример команды ниже, для удобства можно использовать удобный вам текстовый редактор (в нашем случае публичный ключ был скопирован на Хост Б):

```
cat my_key.pub >> .ssh/authorized_keys
```

Либо операцию выше можно выполнить с помощью другой команды:

```
ssh-copy-id -i /root/.ssh/my_key root@<Хост Б>
```

Теперь на Хосте Б нужно поменять конфиг сервиса sshd, чтобы по SSH по паролю вход был запрещен:

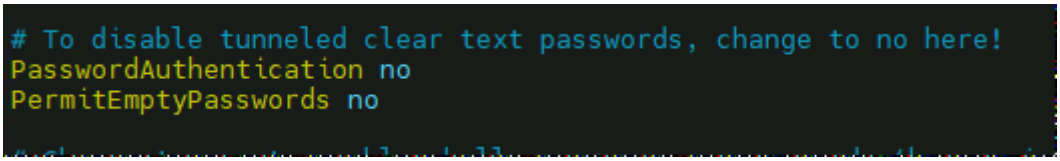
```
vi /etc/ssh/sshd_config
```

Раскомментируем/Добавляем следующие строки:

```
PasswordAuthentication no
PermitEmptypasswords no
LoginGraceTime 0

# if ssh tunnels need
AllowTcpForwarding yes
GatewayPorts yes
```

Получем следующее:



Выходим из редактирования с сохранением (для vi потребуется отдельная статья, чтобы узнать как оттуда выйти - шутка).

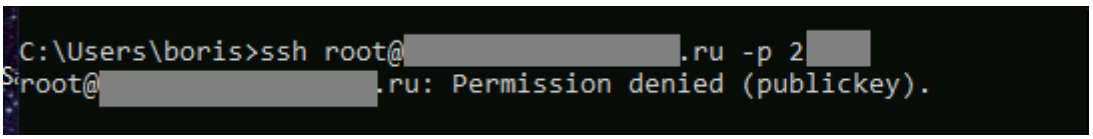
Перезапускаем службу сервиса SSH на Хосте Б:

```
systemctl restart ssh.service
```

Теперь на Хост Б можно зайти из машины, где присутсвует закрытый ключ, пример команды подключения с ключом на Linux системах:

```
ssh -i my_key root@<Хост Б>
```

А при попытке входа по паролю на Хост Б будет возникать подобная ошибка:



?????? ??????? ? ?? root ? ????????? ??

Прежде чем запретить доступ для root создадим отдельного пользователя `admin` с домашней директорией и с доступом по SSH по ключу (ключ необходимо предварительно сгенерировать см. **эту** статью):

```
useradd -m admin
usermod -aG sudo admin
passwd admin
su admin
mkdir /home/admin/.ssh
cat ssh_key.pub >> /home/admin/.ssh/authorized_keys
chown -R admin:admin /home/admin/.ssh
chmod 700 /home/admin/.ssh
chmod 600 /home/admin/.ssh/authorized_keys
```

Проверяем возможность входа пользователем admin.

Запретим вход по SSH пользователю root, для этого нужно поменять конфиг сервиса sshd:

```
vi /etc/ssh/sshd_config
```

Находим или добавляем (если такой строки нет) следующие строки:

```
PermitRootLogin no
```

Выходим из редактирования с сохранением.

Перезапускаем службу сервиса SSH:

```
systemctl restart sshd.service
```

Далее сбрасываем пароль для УЗ root:

```
sudo passwd -l root
```

????????? ??

?? ??? ??????????? ????? ? ???????

Такие УЗ могут потребоваться например для проброса портов или SSH туннелирования и т.д. Ниже пример создания такой УЗ:

```
useradd -m portfwd
passwd portfwd
usermod -s /sbin/nologin portfwd
```

???????? ?? ??? ????????? ??????? ?? SCP

На исходной Linux машине, откуда будут забираться бекапы, необходимо создать отдельную специальную УЗ с доступом в определенную папку, в нашем случае это папка `/backup` куда складываются локальные бекапы:

```
adduser --home /backup user_back; chown user_back:user_back /backup; chmod 744 /backup
```

Далее необходимо создать пару ключей для беспарольного входа по этой УЗ:

```
sudo -u user_back ssh-keygen
# Добавить значение из созданного файла *pub в authorized_keys в домашней директории
пользователя user_back
nano /backup/.ssh/authorized_keys
```

Закрытый ключ (без расширения pub) необходимо скопировать на удаленный хост, который будет использоваться для доступа к исходной машине. Далее для копирования можно использовать следующую команду (где 2222 порт SSH):

```
chmod 700 /root/.ssh/user_back_key
/usr/bin/scp -i /root/.ssh/user_back_key -P 2222 user_back@source.server:/backup/local-
backup.tar.gz /source_server_backup/source-server-backup_$(date +"%d%m%Y").tar.gz
```

Для автоматизации можно добавить эту команду в планировщик задач CRON:

```
crontab -e
```

Для бекапа каждое воскресенье в 00:00 нужно добавить следующую запись в конец файла:

```
0 0 * * 0 /usr/bin/scp -i /root/.ssh/user_back_key -P 2222
user_back@source.server:/backup/local-backup.tar.gz /source_server_backup/source-server-
backup_$(date +"%d%m%Y").tar.gz
```

Для собственного расписания можно использовать этот ресурс https://crontab.guru/

Чтобы удалять архивы старше 30 дней, можно в планировщик также добавить следующую команду:

```
find /source_server_backup/*.gz -type f -mtime +30 -delete
```

????????? fail2ban (?????? ?? ?????????)

Fail2ban служба в Linux которая по log–файлам приложений может обнаружить злоумышленника и заблокировать его IP адрес. Программа умеет бороться с различными атаками на все популярные *NIX–сервисы, такие как Apache, Nginx, Guacamole, sshd, Exim, Postfix и другие. В данной статье мы будем защищать службу SSH.

Установка:

```
apt-get -y install fail2ban
```

Включаем автозапуск и запускаем:

```
systemctl enable fail2ban
systemctl status fail2ban
systemctl start fail2ban
```

Правим конфиг файл:

```
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
vi /etc/fail2ban/jail.local
```

Содержимое файла jail.local:

```
[DEFAULT]
bantime  = 1d
findtime  = 4h
maxretry = 2

[sshd]
enabled = true
mode = aggressive
port = 2222
# incremental banning:
bantime.increment = true
# default factor (causes increment - 1h -> 1d 2d 4d 8d 16d 32d ...):
bantime.factor = 24
```

```
# max banning time = 6 week:
bantime.maxtime = 6w
logpath = %(sshd_log)s
backend = %(sshd_backend)s
```

В конфиге выше fail2ban смотрит порт 2222 службы ssh, будет блокировка на 1 день источника подключения в случае 2 неверных попыток ввода в течение 4 часов, блокировка инкрементальная, в случае если этот же источник повторит неверные действия - блок на 2 дня и так до максимального срока блокировки до 6 недель.

Перезапуск службы для применения новой конфигурации:

```
systemctl restart fail2ban
```

Проверка журнала fail2ban:

```
less /var/log/fail2ban.log
```

Подсчет количества блокирововк по журналу:

```
cat /var/log/fail2ban.log | grep "\[sshd\] Ban" | wc -l
```

Просмотр заблокированных IP-адресов:

```
fail2ban-client status | grep "Jail list" | sed -E 's/^[^:]+:[ \t]+//' | sed 's/,//g'
```

Разблокировка адреса:

```
fail2ban-client set YOURJAILNAMEHERE unbanip IPADDRESSHERE
```

?????????? ??????? ?? ??? ? iptables

Устанавливаем необходимые пакеты:

```
apt-get -y install iptables-persistent
apt-get -y install ipset
```

Для постоянности работы ipset необходимо создать службу `/etc/systemd/system/ipset-persistent.service`:

```
[Unit]
Description=ipset persistent configuration
Before=network.target

# ipset sets should be loaded before iptables
# Because creating iptables rules with names of non-existent sets is not possible
Before=netfilter-persistent.service
Before=ufw.service

ConditionFileNotEmpty=/etc/iptables/ipset

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/sbin/ipset restore -exist -file /etc/iptables/ipset
# Uncomment to save changed sets on reboot
# ExecStop=/sbin/ipset save -file /etc/iptables/ipset
ExecStop=/sbin/ipset flush
ExecStopPost=/sbin/ipset destroy

[Install]
WantedBy=multi-user.target
RequiredBy=netfilter-persistent.service
RequiredBy=ufw.service
```

Включаем нашу службу:

```
systemctl daemon-reload
systemctl enable ipset-persistent.service
systemctl restart ipset-persistent.service
```

Создаем новую зону содержащую российские подсети:

```
ipset create "RU_zone" hash:net
for IP in $(wget -O - https://www.ipdeny.com/ipblocks/data/countries/ru.zone); do ipset add
"RU_zone" $IP; done
```

Проверяем, что добавились подсети в зону:

```
ipset -L RU_zone | less
```

Для персистентности подсетей сохраняем и подгружаем их из файла:

```
ipset save > /etc/iptables/ipset
ipset restore < /etc/iptables/ipset
```

Сохраняем текущие правила iptables:

```
iptables-save > /root/rules.v4
```

Добавляем разрешение в нужном месте (определите его самостоятельно) доступа по гео в сохраненном файле /root/rules.v4:

```
-A INPUT -p tcp -m set --match-set RU_zone src -m tcp --dport 22000 -j ACCEPT
```

Для персистентности правил сохраняем их:

```
cp /root/rules.v4 /etc/iptables/rules.v4
```

Применение новых правил iptables:

```
iptables-restore < /etc/iptables/rules.v4
```

> Полезная утилита для мониторинга работы iptables - **iptstate**

??????? ?? Linux Hardening (?????????)

> Статья со временем будет пополняться

Запрос пароля при каждом выполнении `sudo`:

```
echo "Defaults timestamp_timeout=0" >> /etc/sudoers
```

Вход в систему под УЗ root запрещен:

```
sudo passwd -l root
```

Планировщик задач `crontab` недоступен для всех, кроме владельца:

```
chown root:root /etc/crontab
chmod og-rwx /etc/crontab
chown root:root /etc/cron.hourly/
chmod og-rwx /etc/cron.hourly/
chown root:root /etc/cron.daily/
chmod og-rwx /etc/cron.daily/
chown root:root /etc/cron.weekly/
chmod og-rwx /etc/cron.weekly/
chown root:root /etc/cron.monthly/
chmod og-rwx /etc/cron.monthly/
chown root:root /etc/cron.d/
chmod og-rwx /etc/cron.d/
```

Защищаем системные разделы `/home`, `/tmp`, `/var/tmp`, `/dev/shm` от хранения и выполнения вредоносных программ с них, отредактируйте или добавьте строчки в файле /`etc/fstab`:

```
tmpfs /tmp tmpfs defaults,rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs defaults,noexec,nodev,nosuid,seclabel 0 0
```

Отключите сбор телеметрии ОС:

```
sudo apt purge -y ubuntu-report popularity-contest apport whoopsie
sudo apt autoremove -y
sudo apt-get autoclean
```

?????????? ?????????? ???????????? ?????? IP ? UFW

Устанавливаем необходимые пакеты:

```
apt-get -y install ipset
```

Создаем блок лист:

```
ipset create "IP_ipsum4_block" hash:ip
```

Добавляем туда значения от общедоступных фидов IPsum (агрегационный фид level 4 - very low false positives) и сохраняем в файл `/etc/ipset.rules`:

```
ipset flush "IP_ipsum4_block"; for IP in $(wget -O -
https://raw.githubusercontent.com/stamparm/ipsum/master/levels/4.txt); do ipset add
"IP_ipsum4_block" $IP; done; ipset save > /etc/ipset.rules
```

Добавляем ежедневное обновление фидов с crontab (для редактирования планировщика: `crontab -e`):

```
0 0 * * * ipset flush "IP_ipsum4_block"; for IP in $(wget -O -
https://raw.githubusercontent.com/stamparm/ipsum/master/levels/4.txt); do ipset add
"IP_ipsum4_block" $IP; done; ipset save > /etc/ipset.rules
```

Для восстановления ipset при перезагрузке зайдите в CRON:

```
crontab -e
```

Добавьте запись и выйдите с сохранением:

```
@reboot ipset restore < /etc/ipset.rules; ufw reload
```

Переустановка UFW

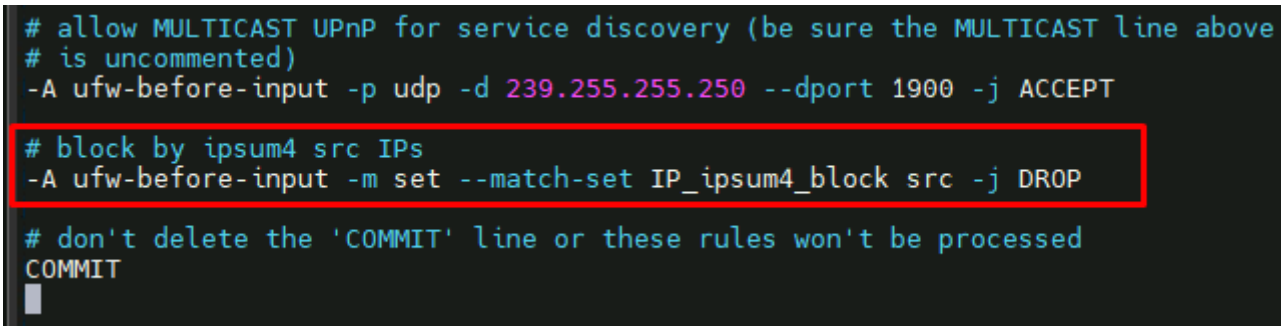
```
sudo apt-get purge --auto-remove ufw
sudo apt-get install ufw
sudo ufw allow 22/tcp
sudo ufw enable
sudo ufw reload
```

Далее добавим правило в UFW:

```
sudo nano /etc/ufw/before.rules
```

Добавьте следующее значение до COMMIT:

```
# block by ipsum4 src IPs
-A ufw-before-input -m set --match-set IP_ipsum4_block src -j DROP
```



Перезапустите UFW:

```
sudo ufw reload
```

Убедитесь, что правило применяется (добавленная запись должна появиться в списке вывода):

```
sudo iptables -S ufw-before-input
```

Полезные команды UFW

```
ufw status verbose
ufw allow in on eth0 from 203.0.113.102
ufw allow from 203.0.113.103 proto tcp to any port 22
ufw allow proto tcp from any to any port 80,443
ufw allow from 203.0.113.0/24 to any port 3306
sudo ufw allow 13647:13650/tcp
```

```
ufw deny out 25
ufw status numbered
ufw delete 13
ufw --force delete 13
ufw reset
```