# Born2beroot Part 2: Configure VirtualMachine
(up until before ftp)

- Introduction to virtualization
- Create first machine in VirtualBox
- Set up own operating system while implementing strict rules.
- Submit a signature.txt file at the root of my repository - paste in it the signature of my machine's virtual disk.

Reference github/javiff8
Reference github/cabartell
Reference github/GuillaumeOz
Reference medium/baigal
Reference reposhub/YOPII
Video Linux Directories 100 secs
Video Linux File System

Hostname: evong42
Host/Root password: Lev##20202
New User: Evangelene
New User ID*: evong
Password: Cayo##5782
Encryption passphrase: Asher#5782

* All print screens, User ID evong22 have been replaced with evong since subject paper requested use of intra login as user.
** All print screens in black are taken from the VirtualMachine and print screens in white are taken from local terminal.

## Hostname
From root, check with <**hostnamectl**>
Change <**hostnamectl set-hostname new_hostname**>
Change /etc/hosts file <**sudo nano /etc/hosts**>
Change old hostname with new
<**127.0.0.1 localhost**>
<**127.0.0.1 new_hostname**>
Reboot to check change <**sudo reboot**>

## Check system requirements
(Show requirements as stated in subject paper)
Type <**lsblk**> to show VM partitions
Type <**head -n 2 /etc/os-release**> or
<**head /etc/os-release**> to see operating system
(-n 2 shows first 2 lines of information block)
Type <**sudo /usr/sbin/aa-status**> to show apparmor module.
Type <**ss -tunlp**> to see ports
Type <**sudo /usr/sbin/ufw status**> to see ufw status

## General
- Use <**Insert-Control-Break**> found at the bottom right hand corner of the VM screen to get out of action.
- Use <**nano**> instead of <**vi**> if having problems with vi.
- To check ip address on Mac, get to Finder with <**command**> + <**space**> then write <**Terminal**>
- To logout from Terminal, <**sudo poweroff**>
- monitoring.sh can be created in <**/usr/local/bin**> (root) or </**home/evong**> (local home directory)
- To change directory for user <**sudo usermod -d /home/evong -m evong**>
- To list <**sudo usermod -l evong evong22**>

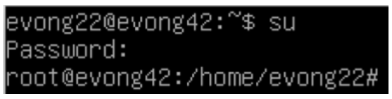
## Sudo
Sudo (Super-user do) is a program designed to let system administrators allow some users to execute some commands as root (another user). Sudo is also an effective way to log who ran which command and when.

**Why sudo?**
Using sudo is better (safer) than opening a session as root for a number of reasons, including:

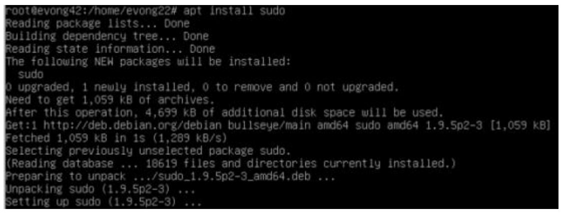
1. Nobody needs to know the root password (sudo prompts for the current user's password). Extra privileges can be granted to individual users temporarily, and then taken away without the need for a password change.
2. It's easy to run only the commands that require special privileges via sudo; the rest of the time, you work as an unprivileged user, which reduces the damage that mistakes can cause.
3. Auditing/logging: when a sudo command is executed, the original username and the command are logged.

For the reasons above, switching to root using **sudo -i** (or **sudo su**) is usually deprecated because it cancels the above features.

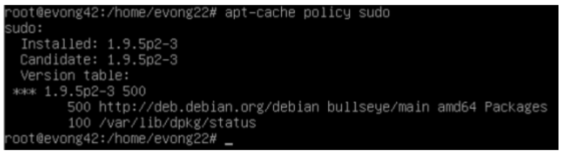
### Install sudo
Get to root with <**su**> or <**su -**>and by using host/root password <**Lev##20202**>
Update sudo <**apt-get update -y**> or <**apt-get upgrade -y**>
Install sudo with <**apt install sudo**>.
Verify with <**apt-cache policy sudo**> or
<**dpkg -l | grep sudo**>.



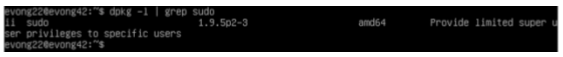
```
evong22@evong42:~$ su
Password:
root@evong42:/home/evong22#
```



```
root@evong42:/home/evong22# apt install sudo
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  sudo
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,059 kB of archives.
After this operation, 4,699 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main amd64 sudo amd64 1.9.5p2-3 [1,059 kB]
Fetched 1,059 kB in 1s (1,289 kB/s)
Selecting previously unselected package sudo.
(Reading database ... 18619 files and directories currently installed.)
Preparing to unpack .../sudo_1.9.5p2-3_amd64.deb ...
Unpacking sudo (1.9.5p2-3) ...
Setting up sudo (1.9.5p2-3) ...
```

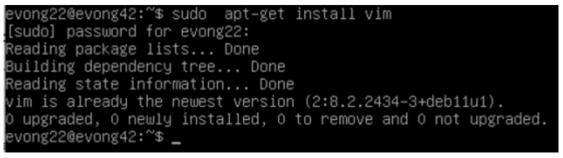


```
root@evong42:/home/evong22# apt-cache policy sudo
sudo:
  Installed: 1.9.5p2-3
  Candidate: 1.9.5p2-3
  Version table:
 *** 1.9.5p2-3 500
        500 http://deb.debian.org/debian bullseye/main amd64 Packages
        100 /var/lib/dpkg/status
root@evong42:/home/evong22# _
```



```
evong22@evong42:~$ dpkg -l | grep sudo
ii  sudo                          1.9.5p2-3                    amd64        Provide limited super u
ser privileges to specific users
evong22@evong42:~$
```

### Install vim
From root, type <**sudo apt-get install vim**>.

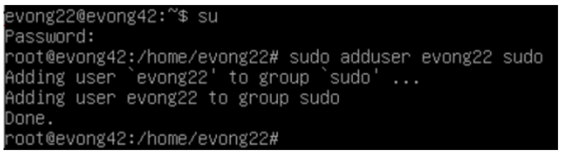


```
evong22@evong42:~$ sudo  apt-get install vim
[sudo] password for evong22:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
vim is already the newest version (2:8.2.2434-3+deb11u1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
evong22@evong42:~$ _
```

## Enable sudo for a user on Debian

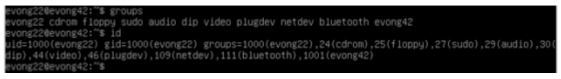
**Users and sudo**
Debian's default configuration allows users in the sudo group to run any command via sudo.

From root, add user to group sudo (so user could use sudo command) <**sudo adduser evong sudo**>.
Alternatively, for Debian 11, use <**/sbin/adduser evong sudo**>
Alternatively, <**usermod -aG sudo evong**>
<**Reboot**> or <**log out and log in again**> for change to take effect.

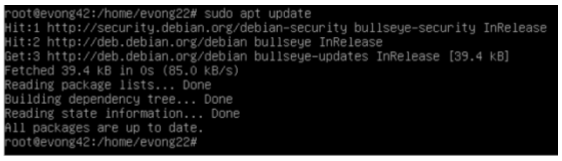


```
evong22@evong42:~$ su
Password:
root@evong42:/home/evong22# sudo adduser evong22 sudo
Adding user `evong22' to group `sudo' ...
Adding user evong22 to group sudo
Done.
root@evong42:/home/evong22#
```

Verify if user belongs to group sudo with <**groups**> or <**id**> or <**getent group sudo**>
Verify sudo powers with  <**sudo -v**>
From root, update all packages of the system with <**sudo apt update**>

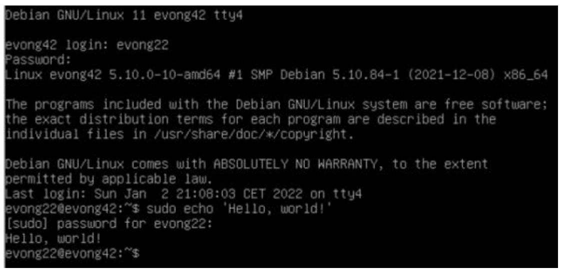


```
evong22@evong42:~$ groups
evong22 cdrom floppy sudo audio dip video plugdev netdev bluetooth evong42
evong22@evong42:~$ id
uid=1000(evong22) gid=1000(evong22) groups=1000(evong22),24(cdrom),25(floppy),27(sudo),29(audio),30(
dip),44(video),46(plugdev),109(netdev),111(bluetooth),1001(evong42)
evong22@evong42:~$
```



```
root@evong42:/home/evong22# sudo apt update
Hit:1 http://security.debian.org/debian-security bullseye-security InRelease
Hit:2 http://deb.debian.org/debian bullseye InRelease
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [39.4 kB]
Fetched 39.4 kB in 0s (85.0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@evong42:/home/evong22#
```

<u>To check if all goes well</u>
Log out and log in with the same user.
Run <**sudo echo 'Hello, world!'**> with password.



```
Debian GNU/Linux 11 evong42 tty4

evong42 login: evong22
Password:
Linux evong42 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-08) 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: Sun Jan  2 21:08:03 CET 2022 on tty4
evong22@evong42:~$ sudo echo 'Hello, world!'
[sudo] password for evong22:
Hello, world!
evong22@evong42:~$
```

Open sudoers file <**sudo visudo**>
Add this line into file <**your_username ALL=(ALL)  ALL**>

<u>Install git</u>
Type <**apt-get update -y**> or <**apt-get upgrade -y**> or <**apt-get install git -y**>
Check git version <git --version>

<u>Install wget</u>
wget is a free and open source tool for downloading files from web repositories.
Type <**sudo apt-get install wget**>

<u>Install Vim</u>
Type <**sudo apt-get install wget**>

<u>Install Oh my zsh (easier to use)</u>
<**sudo apt-get install zsh**>
<**zsh --version**>
<**sh -c "$(wget https://raw.github.com/robbyrussell/oh-my-zsh/master/tools/install.sh 0 -)"**>

## Create new users with sudo

You can also create new users with sudo membership:

**Create new user while installing OS**
As of DebianSqueeze, if you give root an empty password during installation, sudo will be installed and the first user will be able to use it to gain root access (currently, the user will be added to the sudo group). The system will also configure gksu and aptitude to use sudo. You should still verify group membership after logging in as the installed user.

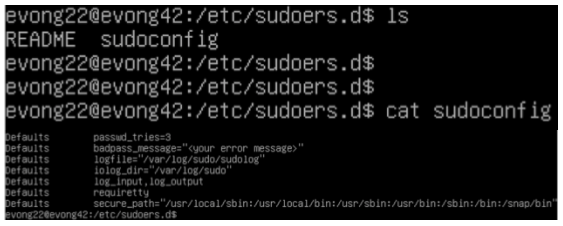
**Create new user from command line**
A user which already has sudo can create another user with sudo group membership:

From command line, run <**sudo adduser vangie22 -G sudo**>.
Log in as vangie22 and verify group membership.

## Configuring sudo

Vim or vi is a text editor default in Linux.
Change directory to <**/etc/sudoers.d/**>
Then type <**sudo vi sudoconfig**>
While inside vim, use <**i**> for insert; <**d**> for delete; <**:q!**> to quit.
Problems with vim editor!!! Use below instead.

To create file, use <**sudo nano sudoconfig**>
<**^**> is equivalent to <**control**> key
Type <**ls**> to list file.
Type <**cat sudoconfig**> to view file.
Type <**su**> + password to check if it is working.
Type <**sudo --help**> + enter to see help menu.
Type <**sudo -v**> to verify sudo powers.



```
evong22@evong42:/etc/sudoers.d$ ls
README  sudoconfig
evong22@evong42:/etc/sudoers.d$
evong22@evong42:/etc/sudoers.d$
evong22@evong42:/etc/sudoers.d$ cat sudoconfig
Defaults        passwd_tries=3
Defaults        badpass_message="<your error message>"
Defaults        logfile="/var/log/sudo/sudolog"
Defaults        iolog_dir="/var/log/sudo"
Defaults        log_input,log_output
Defaults        requiretty
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"
evong22@evong42:/etc/sudoers.d$
```

## SSH

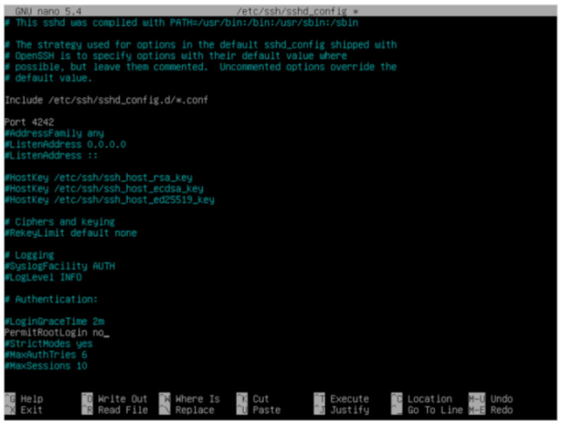
<u>Install openssh-server</u>
Type <**sudo apt install openssh-server**>
Verify with <**apt-cache policy openssh-server**> or <**dpkg -l | grep ssh**>

<u>Configure SSH</u>
Type <**sudo nano /etc/ssh/sshd_config**> or
<**sudo vi /etc/ssh/sshd_config**>

<u>Set up SSH using Port 4242</u>
Only allow connections through port 4242
Replace <**line15 #Port 22**>
With <**line15 Port 4242**>

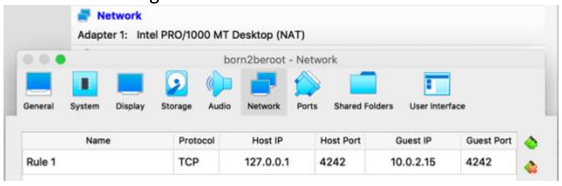
<u>Show  that ssh only uses port 4242</u>
Type <**cat /etc/ssh/sshd_config**>

<u>Disable SSH login as root</u>
(regardless of authentication mechanism)
Replace <**line34 #PermitRootLogin prohibit-password**>
With <**line34 PermitRootLogin no**>



```
  GNU nano 5.4                    /etc/ssh/sshd_config *
# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 4242
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no_
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10


^G Help      ^O Write Out  ^W Where Is  ^K Cut       ^T Execute   ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line M-E Redo
```

Add  forward rule for VirtualBox
VirtualBox -> Choose VM -> Network -> Adapter 1 -> Advanced -> Port Forwarding -> Enter values

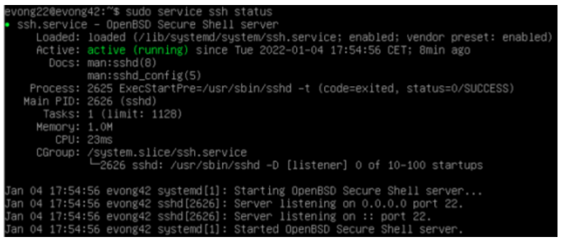


| Name | Protocol | Host IP | Host Port | Guest IP | Guest Port |
|---|---|---|---|---|---|
| Rule 1 | TCP | 127.0.0.1 | 4242 | 10.0.2.15 | 4242 |

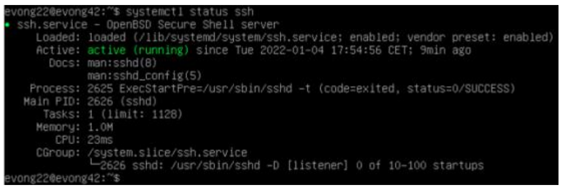
Restart SSH server with
<sudo systemctl restart ssh>

Check SSH status
Type <sudo service ssh status> or
Type <sudo service sshd status> or
Type <sudo systemctl status ssh>

Connect with <ssh evong@127.0.0.1 -p 4242>
To quit <exit>



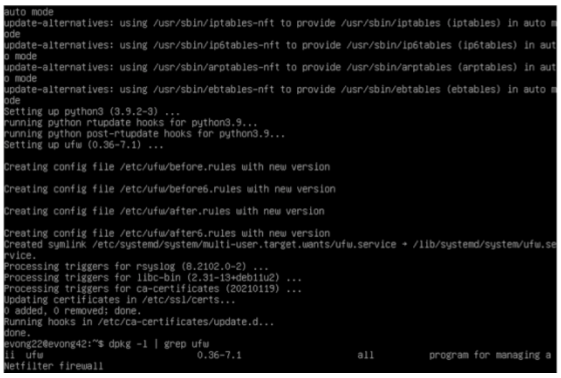
```
evong22@evong42:~$ sudo service ssh status
• ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-01-04 17:54:56 CET; 8min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 2625 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 2626 (sshd)
      Tasks: 1 (limit: 1128)
     Memory: 1.0M
        CPU: 23ms
     CGroup: /system.slice/ssh.service
             └─2626 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

Jan 04 17:54:56 evong42 systemd[1]: Starting OpenBSD Secure Shell server...
Jan 04 17:54:56 evong42 sshd[2626]: Server listening on 0.0.0.0 port 22.
Jan 04 17:54:56 evong42 sshd[2626]: Server listening on :: port 22.
Jan 04 17:54:56 evong42 systemd[1]: Started OpenBSD Secure Shell server.
```



```
evong22@evong42:~$ systemctl status ssh
• ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-01-04 17:54:56 CET; 9min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 2625 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 2626 (sshd)
      Tasks: 1 (limit: 1128)
     Memory: 1.0M
        CPU: 23ms
     CGroup: /system.slice/ssh.service
             └─2626 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
evong22@evong42:~$
```

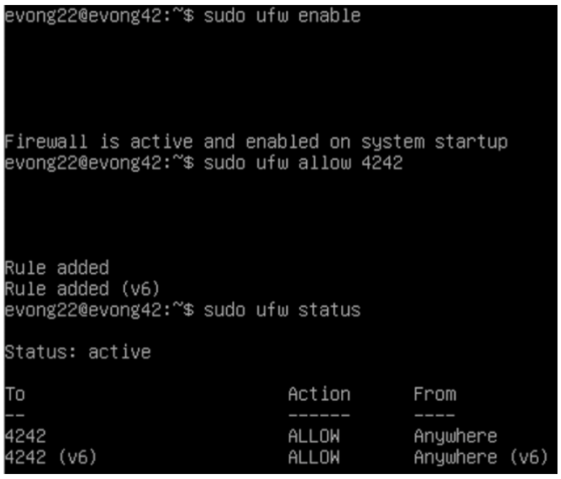
## UFW Uncomplicated Firewall

Install UFW
Type <sudo apt install ufw>
Verify with <dpkg -l | grep ufw>
Enable with <sudo ufw enable>



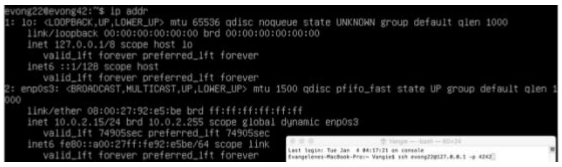
```
auto mode
update-alternatives: using /usr/sbin/iptables-nft to provide /usr/sbin/iptables (iptables) in auto m
ode
update-alternatives: using /usr/sbin/ip6tables-nft to provide /usr/sbin/ip6tables (ip6tables) in aut
o mode
update-alternatives: using /usr/sbin/arptables-nft to provide /usr/sbin/arptables (arptables) in aut
o mode
update-alternatives: using /usr/sbin/ebtables-nft to provide /usr/sbin/ebtables (ebtables) in auto m
ode
Setting up python3 (3.9.2-3) ...
running python rtupdate hooks for python3.9...
running python post-rtupdate hooks for python3.9...
Setting up ufw (0.36-7.1) ...

Creating config file /etc/ufw/before.rules with new version

Creating config file /etc/ufw/before6.rules with new version

Creating config file /etc/ufw/after.rules with new version

Creating config file /etc/ufw/after6.rules with new version
Created symlink /etc/systemd/system/multi-user.target.wants/ufw.service → /lib/systemd/system/ufw.se
rvice.
Processing triggers for rsyslog (8.2102.0-2) ...
Processing triggers for libc-bin (2.31-13+deb11u2) ...
Processing triggers for ca-certificates (20210119) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
evong22@evong42:~$ dpkg -l | grep ufw
ii  ufw                          0.36-7.1                     all          program for managing a
Netfilter firewall
```

Configure UFW
Allow SSH connections through Port 4242
Type <sudo ufw allow 4242>
Type <sudo ufw allow ssh>
Verify with <sudo ufw status verbose>
List numbers associated with ports <sudo ufw status numbered>
Delete with <sudo ufw delete number>
Alternate way to delete port: include rule and number <sudo ufw delete rule number>
List help <sudo ufw --help> or <sudo ufw ls>



```
evong22@evong42:~$ sudo ufw enable



Firewall is active and enabled on system startup
evong22@evong42:~$ sudo ufw allow 4242



Rule added
Rule added (v6)
evong22@evong42:~$ sudo ufw status

Status: active

To                         Action      From
--                         ------      ----
4242                       ALLOW       Anywhere
4242 (v6)                  ALLOW       Anywhere (v6)
```

Connecting to Server via SSH
Check ip with <ip> or <ip addr> or <Hostname -I>
SSH into VM using Port 4242 via
<ssh evong@ip-address -p 4242> ->
<ssh evong@localhost -p 4242>



```
evong22@evong42:~$ ip addr
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 08:00:27:92:e5:be brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 74905sec preferred_lft 74905sec
    inet6 fe80::a00:27ff:fe92:e5be/64 scope link
       valid_lft forever preferred_lft forever
```

```
Last login: Tue Jan  4 04:17:21 on console
Evangelenes-MacBook-Pro:~ Vangie$ ssh evong22@127.0.0.1 -p 4242
```

Terminate SSH
Type <logout> or <exit>

## User Management: Password Policy

Change pw for root <sudo nano /etc/shadow>
Looking for pw root <sudo chage -l root>

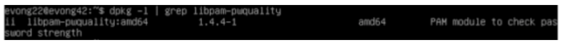
Password Age/Expiration
Change policy with <sudo nano /etc/login.defs> or < sudo vi /etc/login.defs >

To set password to expire every 30 days
Replace <PASS_MAX_DAYS  99999>
With <PASS_MAX_DAYS  30>

To set minimum days between password changes
Replace <PASS_MIN_DAYS  0>
With <PASS_MIN_DAYS  2>

To send user a warning message 7 days before password expiry
Check <PASS_WARN_AGE  7>

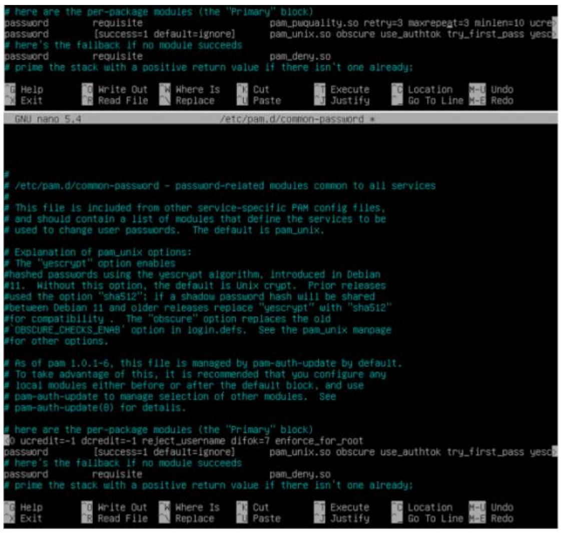
Define Password Strength
Install libpam-pwquality package with <**sudo apt install libpam-pwquality**>
Verify with <**dpkg -l | grep libpam-pwquality**>



```
evong22@evong42:~$ dpkg -l | grep libpam-pwquality
ii  libpam-pwquality:amd64        1.4.4-1                 amd64        PAM module to check pas
sword strength
```

Configure password strength policy
Type <**sudo nano /etc/pam.d/common-password**> or
<**sudo vi /etc/pam.d/common-password**>
Replace
<**password  requisite  pam_pwquality.so  retry=3**>
With
<**password  requisite  pam_ pwquality.so  retry=3 maxrepeat=3 minlen=10 ucredit=-1 dcredit=-1 reject_username difok=7 enforce_for_root**>

Password Strength Policy clarifications
- A maximum of 3 consecutive identical entries.
- Password minimum length to 10 entries.
- Password to contain at least an uppercase entry (ucredit) and a numeric entry (dcredit).
- To reject password if it contains username.
- To set the number of changes required in the new password from the old password to 7.
- To enforce same policy on root.



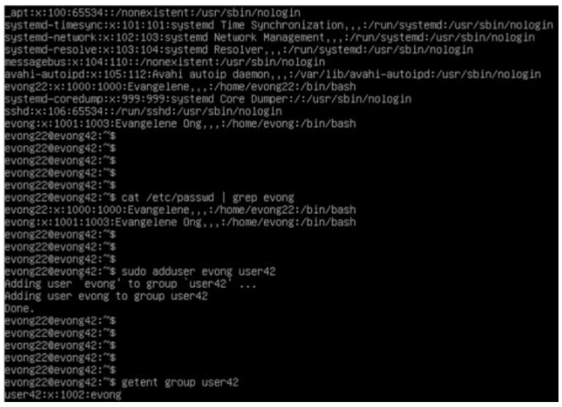
```
# here are the per-package modules (the "Primary" block)
password        requisite                       pam_pwquality.so retry=3 maxrepeat=3 minlen=10 ucre>
password        [success=1 default=ignore]      pam_unix.so obscure use_authtok try_first_pass yesc>
# here's the fallback if no module succeeds
password        requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^_ Go To Line  M-E Redo
  GNU nano 5.4                      /etc/pam.d/common-password *



#
# /etc/pam.d/common-password - password-related modules common to all services
#
# This file is included from other service-specific PAM config files,
# and should contain a list of modules that define the services to be
# used to change user passwords.  The default is pam_unix.

# Explanation of pam_unix options:
# The "yescrypt" option enables
#hashed passwords using the yescrypt algorithm, introduced in Debian
#11.  Without this option, the default is Unix crypt.  Prior releases
#used the option "sha512"; if a shadow password hash will be shared
#between Debian 11 and older releases replace "yescrypt" with "sha512"
#for compatibility .  The "obscure" option replaces the old
#`OBSCURE_CHECKS_ENAB' option in login.defs.  See the pam_unix manpage
#for other options.

# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
<0 ucredit=-1 dcredit=-1 reject_username difok=7 enforce_for_root
password        [success=1 default=ignore]      pam_unix.so obscure use_authtok try_first_pass yesc>
# here's the fallback if no module succeeds
password        requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^_ Go To Line  M-E Redo
```

Check if password rules working in users:
<**chage -l your_new_username**>

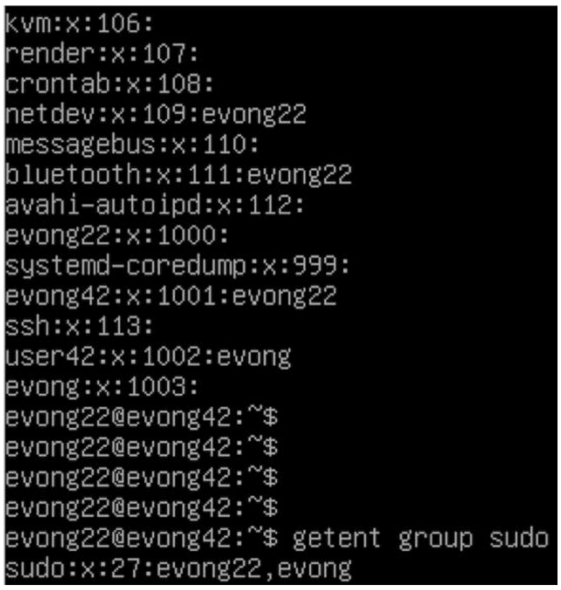
## Setting User Groups

Create a new group
Type <**sudo addgroup user42**> or
<**sudo groupadd user42**>

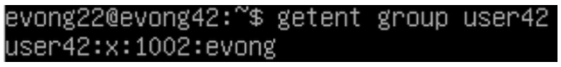
Add a user to group
Type <**sudo adduser evong user42**> or
<**sudo usermod -aG user42 evong**>
Verify group with <**getent group user42**>



```
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:101:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:105:112:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
evong22:x:1000:1000:Evangelene,,,:/home/evong22:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
evong:x:1001:1003:Evangelene Ong,,,:/home/evong:/bin/bash
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$ cat /etc/passwd | grep evong
evong22:x:1000:1000:Evangelene,,,:/home/evong22:/bin/bash
evong:x:1001:1003:Evangelene Ong,,,:/home/evong:/bin/bash
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$ sudo adduser evong user42
Adding user `evong' to group `user42' ...
Adding user evong to group user42
Done.
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$ getent group user42
user42:x:1002:evong
```



```
kvm:x:106:
render:x:107:
crontab:x:108:
netdev:x:109:evong22
messagebus:x:110:
bluetooth:x:111:evong22
avahi-autoipd:x:112:
evong22:x:1000:
systemd-coredump:x:999:
evong42:x:1001:evong22
ssh:x:113:
user42:x:1002:evong
evong:x:1003:
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$ getent group sudo
sudo:x:27:evong22,evong
```



```
evong22@evong42:~$ getent group user42
user42:x:1002:evong
```

## Configure sudo group rules

Create file in /etc/sudoers.d/
Type <**sudo vi /etc/sudoers.d/sudolog**>

Add these rules to file

| | |
|---|---|
| Defaults | passwd_tries=3 |
| Defaults | badpass_message="<your error message>" |
| Defaults | logfile="/var/log/sudo/<filename>" |
| Defaults | iolog_dir="/var/log/sudo" |
| Defaults | log_input,log_output |
| Defaults | requiretty |
| Defaults | |

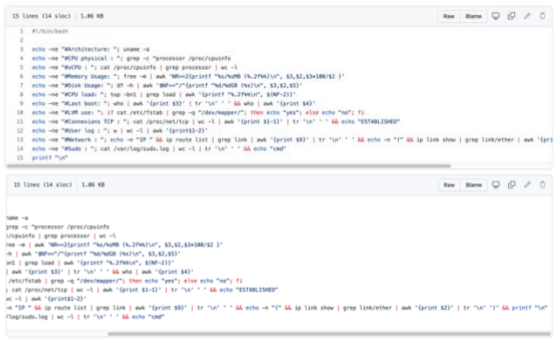
secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

The sudo folder and logfile will automatically be created.

## Write a script for the cron job

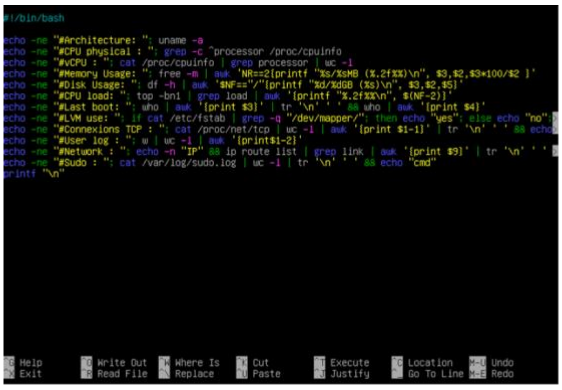
Create a script file with <**nano monitoring.sh**>
(sh means it is a shell script)
Type in contents
Exit and save with <**control**> + <**X**>
Run script with <**bash ./monitoring.sh**>

Dot means look for script in current directory.
Since the script needs to access the sudolog file, we need to update for rwx permission for sudolog file (e.g. chmod 421 for rwx).
List files <**ls -l**>
To amend use <**cat monitoring.sh**>

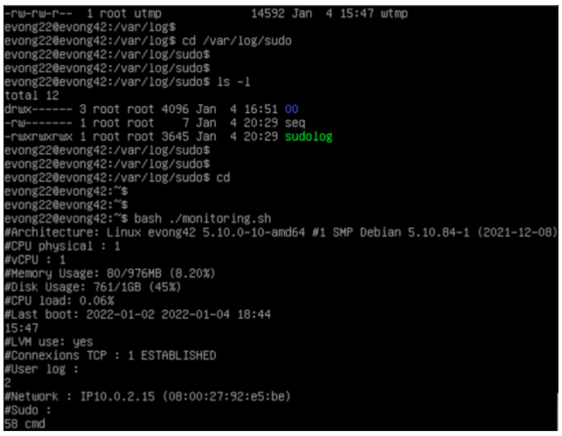


```
#!/bin/bash

echo -ne "#Architecture: "; uname -a
echo -ne "#CPU physical : "; grep -c ^processor /proc/cpuinfo
echo -ne "#vCPU : "; cat /proc/cpuinfo | grep processor | wc -l
echo -ne "#Memory Usage: "; free -m | awk 'NR==2{printf "%s/%sMB (%.2f%%)\n", $3,$2,$3*100/$2 }'
echo -ne "#Disk Usage: "; df -h | awk '$NF=="/"{printf "%d/%dGB (%s)\n", $3,$2,$5}'
echo -ne "#CPU load: "; top -bn1 | grep load | awk '{printf "%.2f%%\n", $(NF-2)}'
echo -ne "#Last boot: "; who | awk '{print $3}' | tr '\n' ' ' && who | awk '{print $4}'
echo -ne "#LVM use: "; if cat /etc/fstab | grep -q "/dev/mapper/"; then echo "yes"; else echo "no"; fi
echo -ne "#Connexions TCP : "; cat /proc/net/tcp | wc -l | awk '{print $1-1}' | tr '\n' ' ' && echo "ESTABLISHED"
echo -ne "#User log : "; w | wc -l | awk '{print$1-2}'
echo -ne "#Network : "; echo -n "IP " && ip route list | grep link | awk '{print $9}' | tr '\n' ' ' && echo -n "(" && ip link show | grep link/ether | awk '{pri
echo -ne "#Sudo : "; cat /var/log/sudo.log | wc -l | tr '\n' ' ' && echo "cmd"
printf "\n"
```

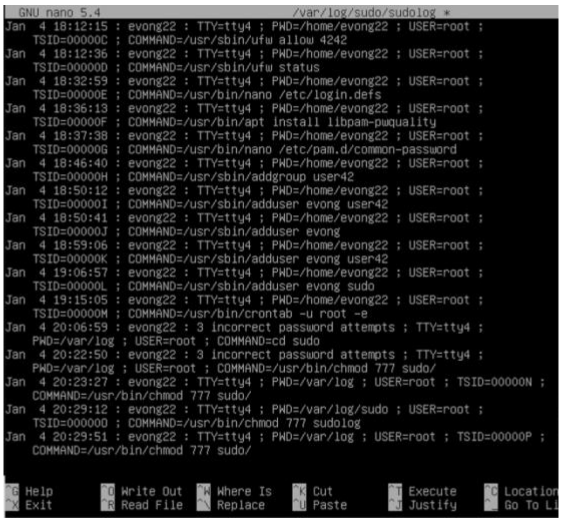
```
name -a
grep -c ^processor /proc/cpuinfo
:/cpuinfo | grep processor | wc -l
ree -m | awk 'NR==2{printf "%s/%sMB (%.2f%%)\n", $3,$2,$3*100/$2 }'
-h | awk '$NF=="/"{printf "%d/%dGB (%s)\n", $3,$2,$5}'
bn1 | grep load | awk '{printf "%.2f%%\n", $(NF-2)}'
| awk '{print $3}' | tr '\n' ' ' && who | awk '{print $4}'
/etc/fstab | grep -q "/dev/mapper/"; then echo "yes"; else echo "no"; fi
; cat /proc/net/tcp | wc -l | awk '{print $1-1}' | tr '\n' ' ' && echo "ESTABLISHED"
wc -l | awk '{print$1-2}'
-n "IP " && ip route list | grep link | awk '{print $9}' | tr '\n' ' ' && echo -n "(" && ip link show | grep link/ether | awk '{print $2}' | tr '\n' ')' && printf "\n"
/log/sudo.log | wc -l | tr '\n' ' ' && echo "cmd"
```



```
#!/bin/bash

echo -ne "#Architecture: "; uname -a
echo -ne "#CPU physical : "; grep -c ^processor /proc/cpuinfo
echo -ne "#vCPU : "; cat /proc/cpuinfo | grep processor | wc -l
echo -ne "#Memory Usage: "; free -m | awk 'NR==2{printf "%s/%sMB (%.2f%%)\n", $3,$2,$3*100/$2 }'
echo -ne "#Disk Usage: "; df -h | awk '$NF=="/"{printf "%d/%dGB (%s)\n", $3,$2,$5}'
echo -ne "#CPU load: "; top -bn1 | grep load | awk '{printf "%.2f%%\n", $(NF-2)}'
echo -ne "#Last boot: "; who | awk '{print $3}' | tr '\n' ' ' && who | awk '{print $4}'
echo -ne "#LVM use: "; if cat /etc/fstab | grep -q "/dev/mapper/"; then echo "yes"; else echo "no";
echo -ne "#Connexions TCP : "; cat /proc/net/tcp | wc -l | awk '{print $1-1}' | tr '\n' ' ' && echo
echo -ne "#User log : "; w | wc -l | awk '{print$1-2}'
echo -ne "#Network : "; echo -n "IP" && ip route list | grep link | awk '{print $9}' | tr '\n' ' '
echo -ne "#Sudo : "; cat /var/log/sudo.log | wc -l | tr '\n' ' ' && echo "cmd"
printf "\n"
```



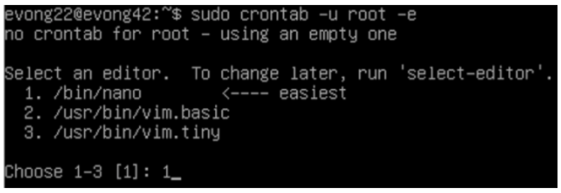
```
-rw-rw-r--  1 root utmp            14592 Jan  4 15:47 wtmp
evong22@evong42:/var/log$
evong22@evong42:/var/log$ cd /var/log/sudo
evong22@evong42:/var/log/sudo$
evong22@evong42:/var/log/sudo$
evong22@evong42:/var/log/sudo$ ls -l
total 12
drwx------ 3 root root 4096 Jan  4 16:51 00
-rw------- 1 root root    7 Jan  4 20:29 seq
-rwxrwxrwx 1 root root 3645 Jan  4 20:29 sudolog
evong22@evong42:/var/log/sudo$
evong22@evong42:/var/log/sudo$
evong22@evong42:/var/log/sudo$ cd
evong22@evong42:~$
evong22@evong42:~$
evong22@evong42:~$ bash ./monitoring.sh
#Architecture: Linux evong42 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-08)
#CPU physical : 1
#vCPU : 1
#Memory Usage: 80/976MB (8.20%)
#Disk Usage: 761/1GB (45%)
#CPU load: 0.06%
#Last boot: 2022-01-02 2022-01-04 18:44
15:47
#LVM use: yes
#Connexions TCP : 1 ESTABLISHED
#User log :
2
#Network : IP10.0.2.15 (08:00:27:92:e5:be)
#Sudo :
58 cmd
```



```
  GNU nano 5.4                            /var/log/sudo/sudolog *
Jan  4 18:12:15 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000C ; COMMAND=/usr/sbin/ufw allow 4242
Jan  4 18:12:36 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000D ; COMMAND=/usr/sbin/ufw status
Jan  4 18:32:59 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000E ; COMMAND=/usr/bin/nano /etc/login.defs
Jan  4 18:36:13 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000F ; COMMAND=/usr/bin/apt install libpam-pwquality
Jan  4 18:37:38 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000G ; COMMAND=/usr/bin/nano /etc/pam.d/common-password
Jan  4 18:46:40 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000H ; COMMAND=/usr/sbin/addgroup user42
Jan  4 18:50:12 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000I ; COMMAND=/usr/sbin/adduser evong user42
Jan  4 18:50:41 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000J ; COMMAND=/usr/sbin/adduser evong
Jan  4 18:59:06 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000K ; COMMAND=/usr/sbin/adduser evong user42
Jan  4 19:06:57 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000L ; COMMAND=/usr/sbin/adduser evong sudo
Jan  4 19:15:05 : evong22 : TTY=tty4 ; PWD=/home/evong22 ; USER=root ;
    TSID=00000M ; COMMAND=/usr/bin/crontab -u root -e
Jan  4 20:06:59 : evong22 : 3 incorrect password attempts ; TTY=tty4 ;
    PWD=/var/log ; USER=root ; COMMAND=cd sudo
Jan  4 20:22:50 : evong22 : 3 incorrect password attempts ; TTY=tty4 ;
    PWD=/var/log ; USER=root ; COMMAND=/usr/bin/chmod 777 sudo/
Jan  4 20:23:27 : evong22 : TTY=tty4 ; PWD=/var/log ; USER=root ; TSID=00000N ;
    COMMAND=/usr/bin/chmod 777 sudo/
Jan  4 20:29:12 : evong22 : TTY=tty4 ; PWD=/var/log/sudo ; USER=root ;
    TSID=00000O ; COMMAND=/usr/bin/chmod 777 sudolog
Jan  4 20:29:51 : evong22 : TTY=tty4 ; PWD=/var/log ; USER=root ; TSID=00000P ;
    COMMAND=/usr/bin/chmod 777 sudo/
```

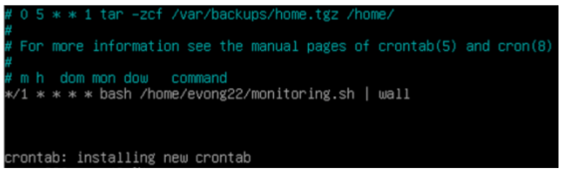
## Set up cron job

<u>Configure a cron job</u>
For this part, check the monitoring.sh file.
Configure cron as root via <**sudo crontab -u root -e**>
(u for user and e for edit)
Select <**1**>



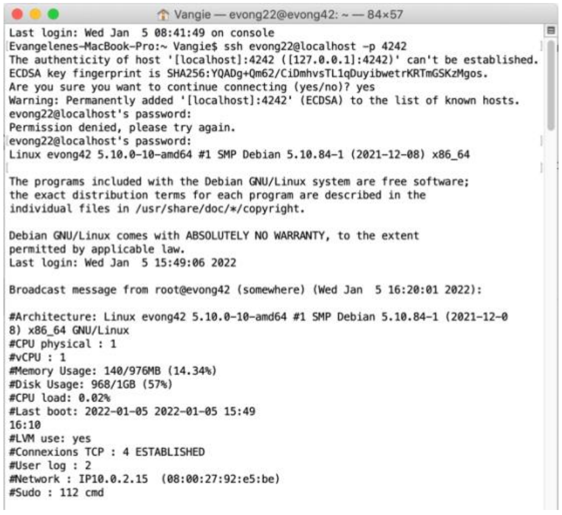
```
evong22@evong42:~$ sudo crontab -u root -e
no crontab for root - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny

Choose 1-3 [1]: 1_
```

<u>To schedule a shell script to run every 10 minutes</u>
Replace <**# m h dom mon dow command**>
With <***/10 * * * * bash /path/to/script | wall**>
i.e. in home drive <***/10 * * * * bash /home/evong22/monitoring.sh | wall**>
or in root drive
<***/10 * * * * bash /usr/local/bin/monitoring.sh | wall**>
wall = write to all
m = minutes
h = hour
dom = day of month
mon = month
dow = day of week

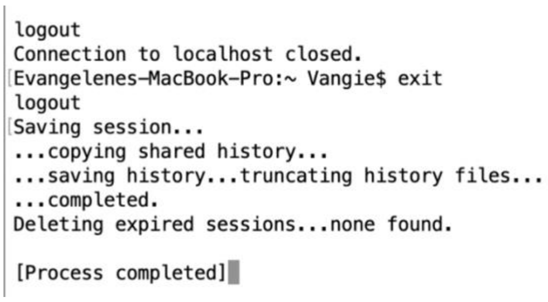


```
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
*/1 * * * * bash /home/evong22/monitoring.sh | wall



crontab: installing new crontab
```

Edit root's scheduled cron jobs via <**sudo crontab -u root -e**>
Verify root's scheduled cron jobs via <**sudo crontab -u root -l**>
Type <**date**> to see current date in VM



```
Vangie — evong22@evong42: ~ — 84×57
Last login: Wed Jan  5 08:41:49 on console
Evangelenes-MacBook-Pro:~ Vangie$ ssh evong22@localhost -p 4242
The authenticity of host '[localhost]:4242 ([127.0.0.1]:4242)' can't be established.
ECDSA key fingerprint is SHA256:YQADg+Qm62/CiDmhvsTL1qDuyibwetrKRTmGSKzMgos.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[localhost]:4242' (ECDSA) to the list of known hosts.
evong22@localhost's password:
Permission denied, please try again.
evong22@localhost's password:
Linux evong42 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-08) 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 Jan  5 15:49:06 2022

Broadcast message from root@evong42 (somewhere) (Wed Jan  5 16:20:01 2022):

#Architecture: Linux evong42 5.10.0-10-amd64 #1 SMP Debian 5.10.84-1 (2021-12-0
8) x86_64 GNU/Linux
#CPU physical : 1
#vCPU : 1
#Memory Usage: 140/976MB (14.34%)
#Disk Usage: 968/1GB (57%)
#CPU load: 0.02%
#Last boot: 2022-01-05 2022-01-05 15:49
16:10
#LVM use: yes
#Connexions TCP : 4 ESTABLISHED
#User log : 2
#Network : IP10.0.2.15  (08:00:27:92:e5:be)
#Sudo : 112 cmd
```



```
logout
Connection to localhost closed.
Evangelenes-MacBook-Pro:~ Vangie$ exit
logout
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
Deleting expired sessions...none found.

[Process completed]
```

## Bonus

### Linux Lighttpd MariaDB PHP (LLMP) Stack

#### Step 1: Installing Lighttpd

Install *lighttpd* via `sudo apt install lighttpd`.

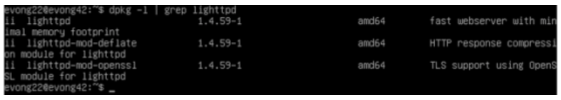
```
&>sudo apt install lighttpd
```

Verify whether *lighttpd* was successfully installed via `dpkg -l | grep lighttpd`.

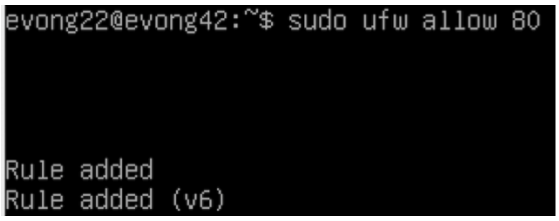
```
$>dpkg -l | grep lighttpd
```

Allow incoming connections using Port 80 via `sudo ufw allow 80`.

```
$>sudo ufw allow 80
```



```
evong22@evong42:~$ dpkg -l | grep lighttpd
ii  lighttpd                   1.4.59-1                   amd64        fast webserver with min
imal memory footprint
ii  lighttpd-mod-deflate       1.4.59-1                   amd64        HTTP response compressi
on module for lighttpd
ii  lighttpd-mod-openssl       1.4.59-1                   amd64        TLS support using OpenS
SL module for lighttpd
evong22@evong42:~$ _
```



```
evong22@evong42:~$ sudo ufw allow 80



Rule added
Rule added (v6)
```

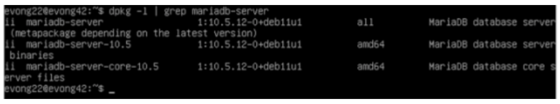
#### Step 2: Installing & Configuring MariaDB

Install *mariadb-server* via `sudo apt install mariadb-server`.

```
$>sudo apt install mariadb-server
```

Verify whether *mariadb-server* was successfully installed via `dpkg -l | grep mariadb-server`.

```
$>dpkg -l | grep mariadb-server
```



```
evong22@evong42:~$ dpkg -l | grep mariadb-server
ii  mariadb-server             1:10.5.12-0+deb11u1        all          MariaDB database server
 (metapackage depending on the latest version)
ii  mariadb-server-10.5        1:10.5.12-0+deb11u1        amd64        MariaDB database server
 binaries
ii  mariadb-server-core-10.5   1:10.5.12-0+deb11u1        amd64        MariaDB database core s
erver files
evong22@evong42:~$ _
```

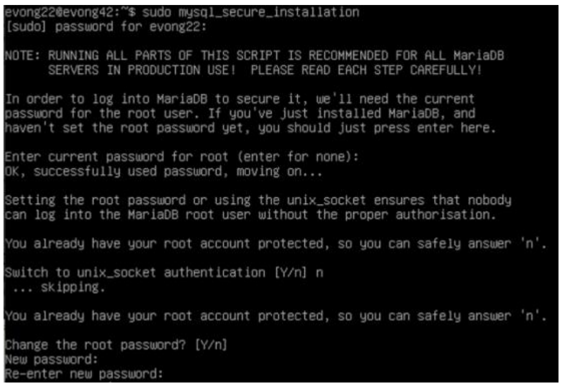
Start interactive script to remove insecure default settings via `sudo mysql_secure_installation`.

```
$>sudo mysql_secure_installation
Enter current password for root (enter for none): #
Set root password? [Y/n] n
Remove anonymous users? [Y/n] Y
Disallow root login remotely? [Y/n] Y
Remove test database and access to it? [Y/n] Y
Reload privilege tables now? [Y/n] Y
```

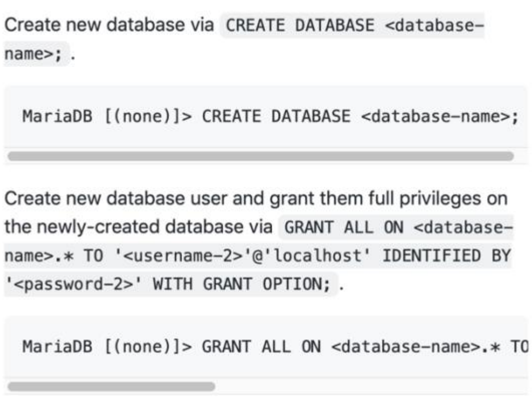
Log in to the MariaDB console via `sudo mariadb`.

```
$>sudo mariadb
MariaDB [(none)]>
```

$>sudo mysql_secure_installation
Enter current password for root (enter for none): #Just press Enter (do not confuse database root with system root)
Set root password? [Y/n] n
Remove anonymous users? [Y/n] Y
Disallow root login remotely? [Y/n] Y
Remove test database and access to it? [Y/n] Y
Reload privilege tables now? [Y/n] Y



```
evong22@evong42:~$ sudo mysql_secure_installation
[sudo] password for evong22:

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
 ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n]
New password:
Re-enter new password:
```



Create new database via `CREATE DATABASE <database-name>;`.

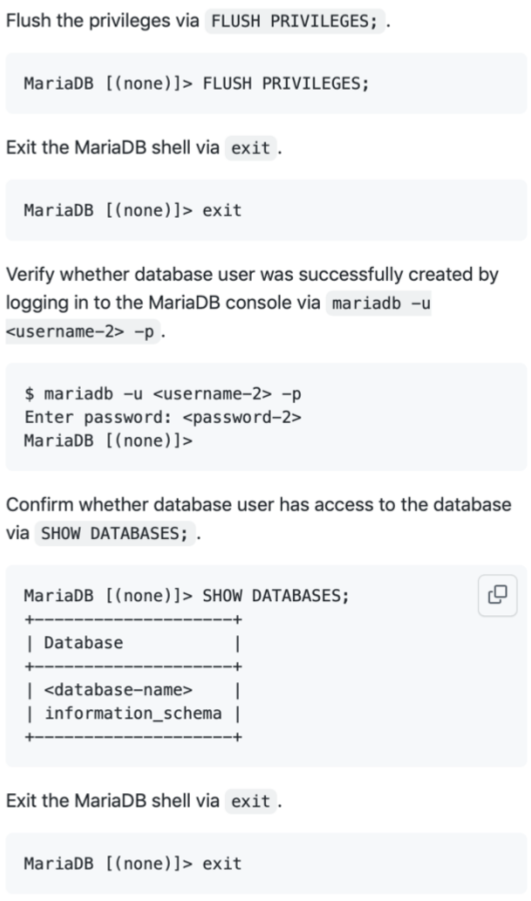
```
MariaDB [(none)]> CREATE DATABASE <database-name>;
```

Create new database user and grant them full privileges on the newly-created database via `GRANT ALL ON <database-name>.* TO '<username-2>'@'localhost' IDENTIFIED BY '<password-2>' WITH GRANT OPTION;`.

```
MariaDB [(none)]> GRANT ALL ON <database-name>.* TO
```

MariaDB [(none)]> CREATE DATABASE <database-name>;

MariaDB [(none)]> GRANT ALL ON <database-name>.* TO '<username-2>'@'localhost' IDENTIFIED BY '<password-2>' WITH GRANT OPTION;



Flush the privileges via `FLUSH PRIVILEGES;`.

```
MariaDB [(none)]> FLUSH PRIVILEGES;
```

Exit the MariaDB shell via `exit`.

```
MariaDB [(none)]> exit
```

Verify whether database user was successfully created by logging in to the MariaDB console via `mariadb -u <username-2> -p`.

```
$ mariadb -u <username-2> -p
Enter password: <password-2>
MariaDB [(none)]>
```

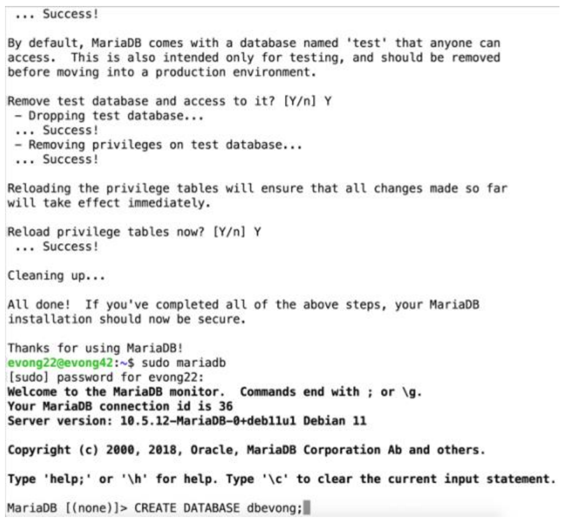
Confirm whether database user has access to the database via `SHOW DATABASES;`.

```
MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| <database-name>    |
| information_schema |
+--------------------+
```

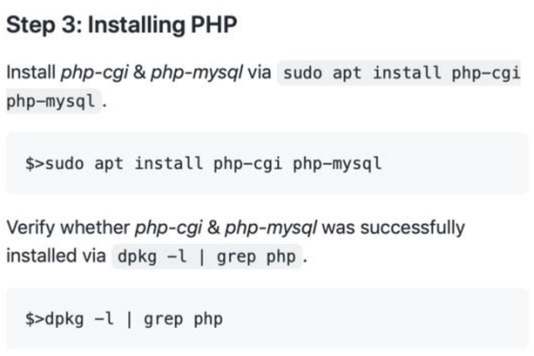
Exit the MariaDB shell via `exit`.

```
MariaDB [(none)]> exit
```

MariaDB username: **dbevong**
MariaDB password: **evongdb**



```
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
evong22@evong42:~$ sudo mariadb
[sudo] password for evong22:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.5.12-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE dbevong;
```



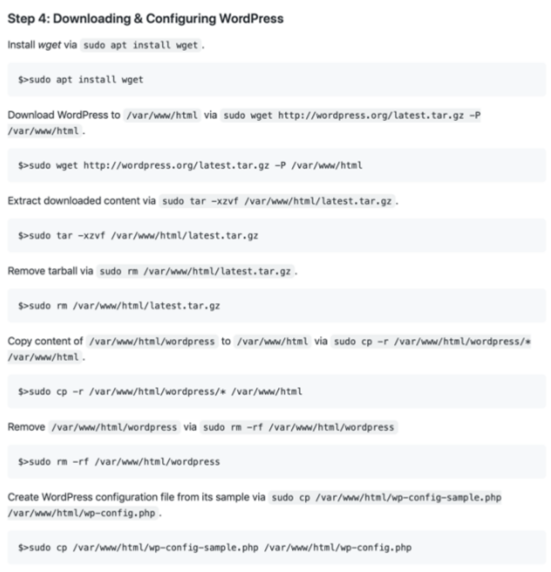
### Step 3: Installing PHP

Install *php-cgi* & *php-mysql* via `sudo apt install php-cgi php-mysql`.

```
$>sudo apt install php-cgi php-mysql
```

Verify whether *php-cgi* & *php-mysql* was successfully installed via `dpkg -l | grep php`.

```
$>dpkg -l | grep php
```



### Step 4: Downloading & Configuring WordPress

Install *wget* via `sudo apt install wget`.

```
$>sudo apt install wget
```

Download WordPress to `/var/www/html` via `sudo wget http://wordpress.org/latest.tar.gz -P /var/www/html`.

```
$>sudo wget http://wordpress.org/latest.tar.gz -P /var/www/html
```

Extract downloaded content via `sudo tar -xzvf /var/www/html/latest.tar.gz`.

```
$>sudo tar -xzvf /var/www/html/latest.tar.gz
```

Remove tarball via `sudo rm /var/www/html/latest.tar.gz`.

```
$>sudo rm /var/www/html/latest.tar.gz
```

Copy content of `/var/www/html/wordpress` to `/var/www/html` via `sudo cp -r /var/www/html/wordpress/* /var/www/html`.

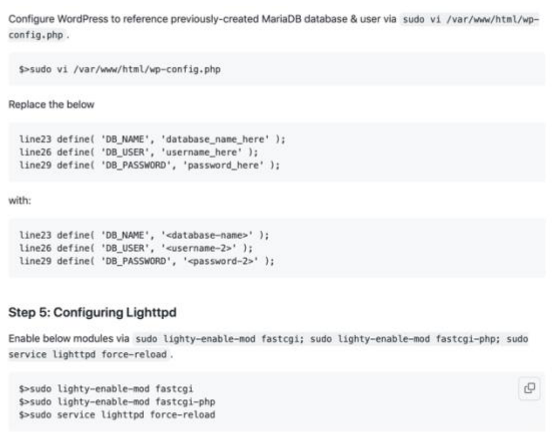
```
$>sudo cp -r /var/www/html/wordpress/* /var/www/html
```

Remove `/var/www/html/wordpress` via `sudo rm -rf /var/www/html/wordpress`

```
$>sudo rm -rf /var/www/html/wordpress
```

Create WordPress configuration file from its sample via `sudo cp /var/www/html/wp-config-sample.php /var/www/html/wp-config.php`.

```
$>sudo cp /var/www/html/wp-config-sample.php /var/www/html/wp-config.php
```



Configure WordPress to reference previously-created MariaDB database & user via `sudo vi /var/www/html/wp-config.php`.

```
$>sudo vi /var/www/html/wp-config.php
```

Replace the below

```
line23 define( 'DB_NAME', 'database_name_here' );
line26 define( 'DB_USER', 'username_here' );
line29 define( 'DB_PASSWORD', 'password_here' );
```

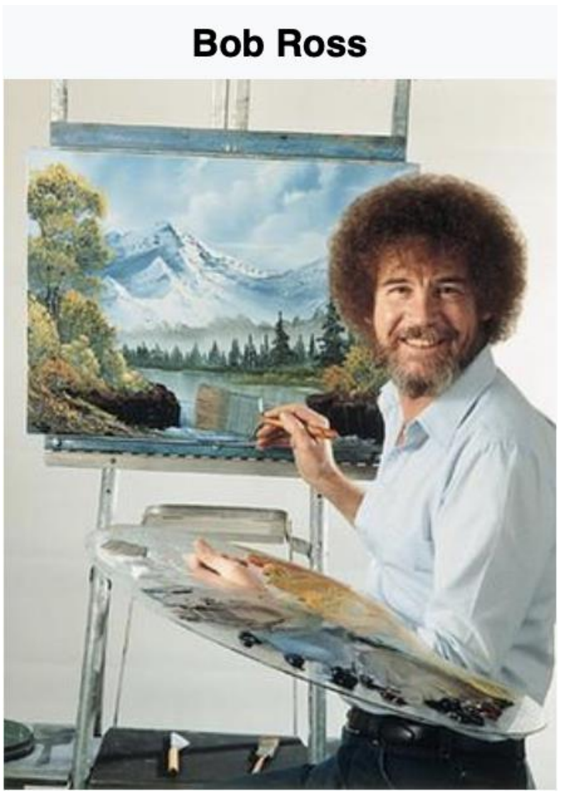
with:

```
line23 define( 'DB_NAME', '<database-name>' );
line26 define( 'DB_USER', '<username-2>' );
line29 define( 'DB_PASSWORD', '<password-2>' );
```

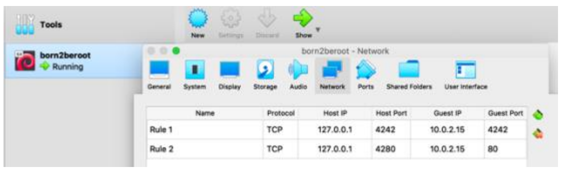
### Step 5: Configuring Lighttpd

Enable below modules via `sudo lighty-enable-mod fastcgi; sudo lighty-enable-mod fastcgi-php; sudo service lighttpd force-reload`.

```
$>sudo lighty-enable-mod fastcgi
$>sudo lighty-enable-mod fastcgi-php
$>sudo service lighttpd force-reload
```

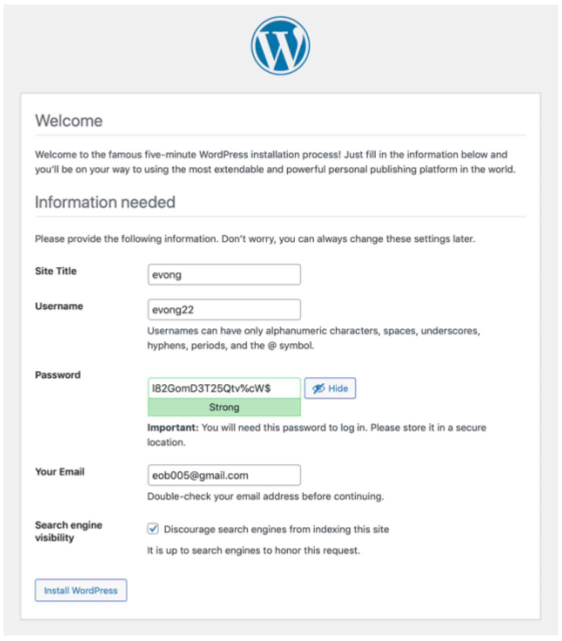


MariaDB name: <**sweden**>
<**evong32@localhost**>
Password: <**ekero**>



| Name | Protocol | Host IP | Host Port | Guest IP | Guest Port |
|---|---|---|---|---|---|
| Rule 1 | TCP | 127.0.0.1 | 4242 | 10.0.2.15 | 4242 |
| Rule 2 | TCP | 127.0.0.1 | 4280 | 10.0.2.15 | 80 |

Web browser: <**127.0.0.1:4280**>
Wordpress password: <**l82GomD3T25Qtv%cW$**>



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

| | |
|---|---|
| Site Title | evong |
| Username | evong22<br>Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol. |
| Password | l82GomD3T25Qtv%cW$ Hide<br>Strong<br>**Important:** You will need this password to log in. Please store it in a secure location. |
| Your Email | eob005@gmail.com<br>Double-check your email address before continuing. |
| Search engine visibility | ☑ Discourage search engines from indexing this site<br>It is up to search engines to honor this request. |