ubuntu

basic configuration

basic configuration

screen

```
vim /etc/screenrc
```

```
hardstatus string "%h%? users: %u%?"
startup_message off
hardstatus alwayslastline "hetzner03: %-Lw%{= BW}%50>%n%f*   %t%{-}%+Lw%<"
bindkey -k k7 prev
bindkey -k k8 next
```

basic configuration

network

The following method allows to change the name of interfaces in ubuntu. The network card below some how is not good recogized by default, one interface is named renameX by default. With this method I assign it the name `enp1s0` by configuration.

First I get all mac addesses of my interfaces

```
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: rename2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:07:43:0c:32:12 brd ff:ff:ff:ff:ff:ff
3: enp130s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen
1000
    link/ether 00:07:43:0c:32:13 brd ff:ff:ff:ff:ff:ff
    inet 10.7.10.70/24 brd 10.7.10.255 scope global dynamic enp130s0
       valid_lft 5639sec preferred_lft 5639sec
    inet6 fe80::207:43ff:fe0c:3213/64 scope link
       valid_lft forever preferred_lft forever
```

Then I enable the feature in the grub configuration to set my own interface names.

```
vim /etc/default/grub
```

```
GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"
```

Apply the new configuration

```
grub-mkconfig -o /boot/grub/grub.cfg
```

```
update-grub
update-initramfs -u
```

Configure the new names per mac address

```
vim /etc/udev/rules.d/70-persistent-net.rules
```

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:07:43:0c:32:12",
NAME="enp1s0"
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:07:43:0c:32:13",
NAME="enp2s0"
```

basic configuration

nameserver

In my LAN I want to make use of the DNS of my OPNsense router, there for I change the setup of ubuntu to use if via the following configuration.

```
systemctl stop systemd-resolved
```

```
systemctl disable systemd-resolved
```

```
vim /etc/resolv.conf
```

```
search home
nameserver 10.7.10.254
```

basic configuration

wipe hdd

```
wipefs -a /dev/sda
```

basic configuration

time synchronization

```
apt install ntp ntpdate -y
```

```
mv /etc/ntp.conf /etc/ntp.conf.orig
vim /etc/ntp.conf
```

```
server 10.8.10.254 prefer iburst
```

```
timedatectl set-ntp no
```

```
service ntp restart
```

```
ntpq -p
ntpdate 10.8.10.254
```

basic configuration

lvm

Check the physical volumes

```
pvs
```

```
  PV         VG        Fmt  Attr PSize  PFree
  /dev/sda3  ubuntu-vg lvm2 a--  <2.73t    0
```

Check the volume groups

```
vgs
```

```
  VG        #PV #LV #SN Attr   VSize  VFree
  ubuntu-vg   1   2   0 wz--n- <2.73t    0
```

Check the logical volumes

```
lvs
```

```
  LV        VG        Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
  plot01    ubuntu-vg -wi-ao----   2.53t
  ubuntu-lv ubuntu-vg -wi-ao---- 200.00g
```

Display volume group details

```
vgdisplay
```

```
  --- Volume group ---
  VG Name               ubuntu-vg
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  3
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
```

```
  Cur LV                2
  Open LV               2
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               <2.73 TiB
  PE Size               4.00 MiB
  Total PE              715140
  Alloc PE / Size       715140 / <2.73 TiB
  Free  PE / Size       0 / 0
  VG UUID               rY8uzZ-2Xrc-TdNR-lOWt-Fr4O-uAKY-H0UsRs
```

Create the additional logical volume

```
lvcreate -l 663940 -n plot01 ubuntu-vg
```

Create filesystem

```
mkfs.ext4 /dev/ubuntu-vg/plot01
```

Get the new UUID

```
blkid
```

```
/dev/mapper/ubuntu--vg-plot01: UUID="3107980c-8ee0-4f63-adeb-2d0f50fe2f5c" TYPE="ext4"
```

Add it to fstab

```
vim /etc/fstab
```

```
UUID=3107980c-8ee0-4f63-adeb-2d0f50fe2f5c /home/chia/chia/chia-blockchain/plot01 ext4
errors=remount-ro        0       1
```

basic configuration

chrony

```
apt install chrony -y
```

```
vim /etc/chrony/chrony.conf
```

```
server 10.8.20.254         iburst  prefer
```

```
systemctl restart chronyd
```

```
chronyc sources
```

```
/sbin/hwclock --systohc
```

ansible

Install basic dependencies for ansible

```
apt update && apt upgrade -y && apt autoremove -y && reboot
apt update
apt install software-properties-common -y
apt-add-repository --yes --update ppa:ansible/ansible
apt install python-argcomplete
vim /etc/ansible/hosts
```

Add the following configuration to your `/etc/asible/hosts` file.

```
[master]
apu03.home

[k8s]
apu[03:06].home

[nodes]
apu[04:06].home

[k8s:vars]
ansible_python_interpreter=/usr/bin/python3
```

Add an `ansible` user

```
useradd -m ansible
```

With group `ansible` and `sudo` allowance

```
usermod -a -G sudo ansible
```

Switch to the user

```
su - ansible
```

Start `bash`

```
bash
```

Create an ssh key

```
ssh-keygen
```

Give the user `ansible` the bash shell as default

```
usermod --shell /bin/bash ansible
```

Allow to sudo without password from the ansible user

```
echo "ansible ALL = (root) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/ansible
sudo chmod 0440 /etc/sudoers.d/ansible
```

Copy the public ssh key of the ansible user

```
vim .ssh/id_rsa.pub
```

Add it on all nodes as authorized key

```
mkdir .ssh
vim .ssh/authorized_keys
```

Find my first playbook on github

icedtea

Install icedtea

```
sudo add-apt-repository ppa:maarten-fonville/ppa
sudo apt-get update
sudo apt-get install icedtea-8-plugin
```

https://askubuntu.com/questions/491995/icedtea-plugin-for-openjdk-8

rclone

```
apt install unzip fuse3 networkd-dispatcher -y
```

```
curl https://rclone.org/install.sh | sudo bash
```

```
vim /etc/fuse.conf
```

```
user_allow_other
```

```
vim /etc/networkd-dispatcher/routable.d/rclone.sh
```

```
#!/bin/bash

echo " Network is up"

runuser -l  cloud -c '/home/cloud/.startup/rclone'
```

chia

chia

plotman

```
sudo apt install python3-pip python3-testresources -y
```

```
pip install --force-reinstall git+https://github.com/ericaltendorf/plotman@main
```

Restart session after, since the binary might not yet be in path.

cuda

Copied from here:
https://www.reddit.com/r/chia/comments/18z4lnp/how_to_install_bladbit_cuda_on_ubuntu/

My steps (I am running ubuntu 22.04, i Had problems with 24.10 due to some dependencies)

I preffer installing the drivers from PPA to get an updated version but "sudo apt install nvidia-driver-535" would also work. the important part is to verify with "nvidia-smi" after reboot

----------Install NVIDIA Drivers on Ubuntu 22.04 via CUDA PPA----------

```
sudo apt install dirmngr ca-certificates software-properties-common apt-transport-https dkms curl -y
```

#import the GPG key

```
curl -fSsL https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/3bf863cc.pub | sudo gpg --dearmor | sudo tee /usr/share/keyrings/nvidia-drivers.gpg > /dev/null 2>&1
```

#add the NVIDIA repository for your system

```
echo 'deb [signed-by=/usr/share/keyrings/nvidia-drivers.gpg] https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/ /' | sudo tee /etc/apt/sources.list.d/nvidia-drivers.list
```

```
sudo apt update
```

Check available Dirver

```
apt search nvidia-driver-*
```

I Chose driver-545 that is a newer driver than the avaiable on the ubuntu repository for 22.04

```
sudo apt install nvidia-driver-545 cuda
```

#Check Driver version

```
nvidia-smi
```

----------Install BaldeBit CUDA----------

#install pre-req

```
sudo apt install -y build-essential cmake libgmp-dev libnuma-dev
```

#Build

```
cd
```

```
git clone https://github.com/Chia-Network/bladebit.git
```

```
cd bladebit
```

```
git checkout cuda-compression
```

```
mkdir -p build-release && cd build-release
```

```
cmake .. && cmake --build . --config Release --target bladebit_cuda -j$(nproc --all)
```

#run

```
cd ~/bladebit/build-release
```

```
./bladebit_cuda -n 1 -f XXX -c XXX --compress XXX cudaplot /media/temp_plots
```

#replace with your farmer key, pool contract, compresion level and save directory

----------Some Other must have ----------

#-----monitoring with a nice terminal GUI---------

```
sudo apt install bpytop
```

```
bytop
```

#---------Moving plots to a final destination---------

```
cd
```

```
git clone https://github.com/lmacken/plow.git
```

#edit plow configuration (most important "SOURCES =" & "DESTS ="

```
sudo nano ./plow/plow.py
```

#install Python

```
sudo apt install python3-pip
```

```
pip install aionotify
```

```
python3 ./plow/plow.py
```