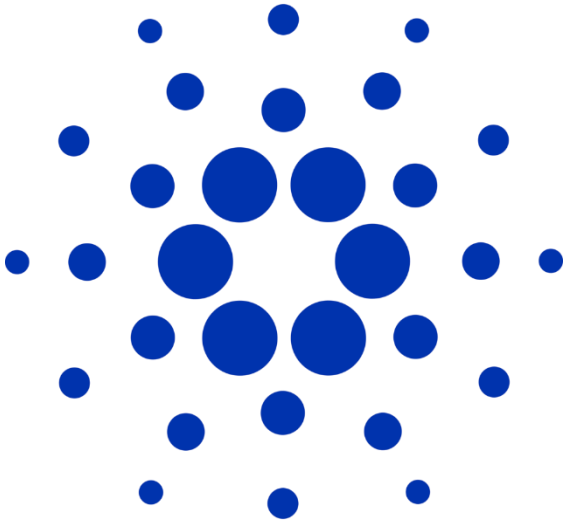
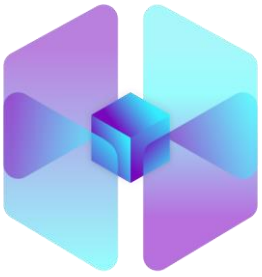


Setup Cheatsheet for Cardano nodes

By [GRANA] GranADA_Staking_Pool



Contents

1 Resources

Cntools [AAA] [CLIO1] [RDLRT] [UNDR] [AHL] [LOVE] [AEON] [BCSH] [EDEN] [STR8] [SMAUG] [PEGA]]

Setup tutorial video [PHNIX]

Security video 2FA [EDEN]

Synchronisation video Chrony [EDEN]

Configuring Grafana [EDEN]

General setup guide [MASTR]

2 First steps

- Create Droplet on data center or install Ubuntu on private server (16 GB RAM for v1.29.0)
- Generate SSH key with PuTTY Keygen (RSA 4096) and save it in your cold storage
- Store credentials and SSH passwords on password manager

[Note: Remember to change all highlighted fields]

3 Creating Non-Root User

```
sudo adduser [username]
sudo adduser [username] sudo
sudo su - [username]
mkdir ~/.ssh
cd .ssh
```

(Login with PuTTY as new user with new SSH key)

```
nano ~/.ssh/authorized_keys
```

(Paste SSH public key -> version displayed by PuTTY keygen)

4 Update Ubuntu

```
sudo apt-get update
sudo apt-get upgrade
```

5 Disable Root Login/Set New SSH Port

```
sudo nano /etc/ssh/sshd_config
```

➔ uncomment port and add number & "PermitRootLogin no"
"PasswordAuthentication no"

```
sudo systemctl restart ssh

sudo systemctl status ssh
-> test that login in with root is no longer possible. Add firewall
rules for droplets
```

6 Server backups (snapshot in private server)

If you are running private servers to enhance your pool's decentralization, you can create snapshots using any of these 2 options:

6.1 TimeShift (GUI)

```
sudo apt-get install timeshift
```

Make sure that your physical volume is formatted in ext4

```
sudo umount /dev/sdb1

sudo mkfs.ext4 -f /dev/sdb1
```

```
sudo timeshift --create --comments [snapshot's name] --tags D --snapshot-
device [target physical volume]
```

➔ This command will create a snapshot of the server and a new config file at this location: /etc/timeshift.json

to restore

```
sudo timeshift --list
```

to restore from a snapshot

```
sudo timeshift --restore --snapshot [snapshot's name]
```

delete a snapshot

```
sudo timeshift --delete  --snapshot [snapshot's name]
```

6.2 LVM (Recommended)

Check if there are any volume groups available

```
sudo vgdisplay
```

You can either configure a LVM partition on a physical volume when installing Ubuntu or just configure a USB stick (min. 8GB) with a LVM partition. First check what volumes are available and select the one to set the LVM partition

```
sudo fdisk -l
```

Check if there are any volume groups available

```
sudo vgdisplay
```

We will assume that we will create a new partition volume in dev/sdb1 (external drive)

```
#This will wipe out all the data in your USB stick and will write the LVM
header to the partition. Just make sure this device doesn't have any
important data and answer "y" to all prompts
sudo pvcreate /dev/sdb1


#Create volume group named backup
sudo vgcreate backup /dev/sdb1

#Create logical volume named snapvol
sudo lvcreate -n snapvol -L 16g backup

#Create snapshot
sudo lvcreate -s -n snap -L 16g backup/snapvol

#Merge snapshot into the origin volume for recovering
sudo lvconvert --merge backup/snap
```

Other useful commands

```
#Resizing partitions
sudo lvextend -L +5g backup/snapvol

#Expand filesystem (after resizing)
sudo resize2fs /dev/backup/snapvol

#Moving partitions
sudo pvmove -n snapvol /dev/sdb1

#Remove logical volume
```

```
sudo lvremove backup/snapvol
```

7 Creating firewall

```
sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw limit proto tcp from any to any port [custom ssh port]
sudo ufw limit proto tcp from any to any port [Cardano port for node]
```

7.1 For Relay Nodes

```
sudo ufw allow [relay port]/tcp
```

7.2 For Producer Node

```
sudo ufw allow proto tcp from [relay 1 ip] to any port [producer port]
sudo ufw allow proto tcp from [relay 2 ip] to any port [producer port]
```

7.3 Finally (both)

```
sudo ufw enable
sudo systemctl restart ssh
sudo ufw status
```

8 Disable Wifi and Bluetooth (Hardware server)

```
systemctl stop bluetooth
systemctl disable bluetooth.service
nmcli radio wifi off
```

9 Disable sleep, hibernation and ping command

```
sudo systemctl mask sleep.target suspend.target hibernate.target
hybrid-sleep.target
sudo sysctl -w net.ipv4.icmp_echo_ignore_all=1
```

10 Configure SWAP for RAM

```
# Swap utilizes diskspace to temporarily handle spikes in memory usage
# Skip this section if you have limited diskspace, (you're running a
raspberry-pi, for instance).

# Show current swap configuration
sudo swapon --show

# Check what swap is currently active, if any
free -h

# Check current disk usage
df -h

# Create swap file (Don't forget the "G")
sudo fallocate -l <SIZE EQUAL TO RAM>G /swapfile

# Verify swap settings
ls -lh /swapfile

# Only root can access swapfile
sudo chmod 600 /swapfile

# Mark the file as swap space
sudo mkswap /swapfile

# Enable swap settings every time we log in
# Make a backup of /etc/fstab
sudo cp /etc/fstab /etc/fstab.bak

# Type this command from the command-line to add swap settings to the
end of fstab
echo '/swapfile none swap sw 00' | sudo tee -a /etc/fstab

# Enable swap
sudo swapon -a

# Verify swap is enabled
free -h
```

11 Security (Google 2FA & Fail2Ban)

```
# Checks logs for ssh logins
sudo journalctl -u ssh


# Checks logs for ssh logins with more details
sudo journalctl -eu ssh

sudo apt install libpam-google-authenticator
```

```
# y to all questions
google-authenticator
sudo nano /etc/ssh/sshd_config
```

➔ `ChallengeResponseAuthentication yes | usePAM yes`

```
sudo systemctl restart ssh

sudo nano /etc/pam.d/sshd
```

➔ Include:

```
#One-time authentication vie Google Authenticator
auth required pam_google_authenticator.so
```

```
sudo nano /etc/ssh/sshd_config
```

➔ Include (at the end of file):

```
AuthenticationMethods publickey,keyboard-interactive
```

```
sudo systemctl restart ssh
sudo apt install fail2ban
sudo systemctl status fail2ban

#checks banned IPs
sudo iptables -L | grep f2b

#if the config file must be changed (defaults are OK)
cd /etc/fail2ban
sudo nano jail.conf
```

12 Synchronisation

```
sudo apt-get install chrony
systemctl status chrony
sudo nano /etc/chrony/chrony.conf
```

➔ Replace:

```
pool time.google.com       iburst minpoll 2 maxpoll 2 maxsources
3 maxdelay 0.3
pool time.facebook.com     iburst minpoll 2 maxpoll 2 maxsources
3 maxdelay 0.3
pool time.euro.apple.com   iburst minpoll 2 maxpoll 2 maxsources
3 maxdelay 0.3
pool time.apple.com        iburst minpoll 2 maxpoll 2 maxsources
3 maxdelay 0.3
pool ntp.ubuntu.com        iburst minpoll 2 maxpoll 2 maxsources
3 maxdelay 0.3

maxupdateskew 5.0
```

```
makestep 0.1 -1


#GET TAI-UTC offset and leap seconds from the system tz database
leapsectz right/UTC
```

➔ Include:

```
# Serve time even if not synchronized to a time source
local stratum 10
```

```
sudo systemctl restart chrony
```

13 Installing Prereqs

```
mkdir "$HOME/tmp"
cd "$HOME/tmp"
curl -sS -o prereqs.sh https://raw.githubusercontent.com/cardano-
community/guild-operators/master/scripts/cnode-helper-
scripts/prereqs.sh

chmod 755 prereqs.sh
./prereqs.sh -f
. "${HOME}/.bashrc"
```

14 Installing cardano-node and cardano-cli

```
cd ~/git
git clone https://github.com/input-output-hk/cardano-node
cd cardano-node
git fetch --tags --all
git pull
git checkout $(curl -s https://api.github.com/repos/input-output-
hk/cardano-node/releases/latest | jq -r .tag_name)

$CNODE_HOME/scripts/cabal-build-all.sh -o

#test installation
cardano-cli version
cardano-node version
```

15 Get mainnet-alonzo-genesis.json file

```
cd $CNODE_HOME/files
curl -sL -f -o alonzo-genesis.json https://hydra.iohk.io/job/Cardano/iohk-
nix/cardano-deployment/latest-finished/download/1/mainnet-alonzo-genesis.json

chmod 755 alonzo-genesis.json
ls -l
nano config.json
```

➔ Include:

```
"AlonzoGenesisFile": "/opt/cardano/cnode/files/alonzo-genesis.json",
 "AlonzoGenesisHash":
"7e94a15f55d1e82d10f09203fa1d40f8eede58fd8066542cf6566008068ed874",

sudo systemctl restart cnode
```

16 Configuring CNODE script to use all available CPU cores (reduces missing slots)

```
cd $CNODE_HOME/scripts
lscpu
#remember number of property "CPU(s)" -> available cores
sudo nano cnode.sh
#uncomment CPU variable and set it to max number of available cores
#save changes and restart node (this avoids missing slots)
```

17 Configuring Env and Starting Node

```
cd $CNODE_HOME/scripts
nano env
```

Change line with CNODE_PORT to:

```
CNODE_PORT=[DESIRED NODE PORT]
```

Press Ctrl + X to exit

Press Y to save modified buffer

Press Enter to keep file name

```
./deploy-as-systemd.sh
[ When Asked About topologyUpdater Select Y for Relays and N for
Producer ]
sudo systemctl start cnode.service
sudo systemctl status cnode.service
./gLiveView.sh #OR ./sLiveView.sh
```

18 Editing topologyUpdater.sh For Relays

```
cd $CNODE_HOME/scripts
nano topologyUpdater.sh
```

Delete # from line with CUSTOM_PEERS and change to:

```
CUSTOM_PEERS = "producer ip:port|relays-new.cardano-
mainnet.iohk.io:3001:2"
```

Press Ctrl + X to exit

Press Y to save modified buffer

Press Enter to keep file name

```
sudo systemctl restart cnode.service
```

19 Editing topology.json for Producer

```
cd $CNODE_HOME/files
nano topology.json
```

Delete Everything and add:

```
{
  "Producers": [
    {
      "addr": "relay 1 ip",
      "port": relay port,
      "valency": 1
    },
    {
      "addr": "relay 2 ip",
      "port": other relay port,
      "valency": 1
    }
  ]
}
```

Press Ctrl + X to exit

Press Y to save modified buffer

Press Enter to keep file name

```
sudo systemctl restart cnode.service
```

20 Creating wallet/Registering pool

```
./cntools.sh
```

21 Monitoring

21.1 Configuring Grafana

21.1.1 Relay (host) part I

```
cd $CNODE_HOME/scripts
./setup_mon.sh #setup script from cntools

#open ports for Prometheus, EKG & Grafana

sudo ufw allow proto tcp from 127.0.0.1 to any port 9090

sudo ufw allow proto tcp from 127.0.0.1 to any port 9091

sudo ufw allow proto tcp from 127.0.0.1 to any port 12798

sudo ufw allow proto tcp from any to any port 5000


#Now you can check the Grafana dashboard in a browser (PC has to be
connected to local network) -> [IP|DNS]:5000

#Check port forwarding in router in case that this doesn't work

#Get JSON config and copy it from this link

#On Grafana Client: Dashboards->manage->add new panel(button)-
>dashboard settings (icon)-> JSON Model->Paste JSON config and save
```

21.1.2 BP

```
cd $CNODE_HOME/files
sudo nano config.json
#change hasPrometheus property (ip: 0.0.0.0) and save -> restart node

sudo systemctl restart cnode.service

cd

cd tmp

wget https://raw.githubusercontent.com/DamjanOstrelic/Cardano-
stuff/master/setup_node_exporter.sh

chmod 755 setup_node_exporter.sh

./setup_node_exporter.sh
```

```
sudo ufw allow proto tcp from [IP host] to any port 9091

sudo ufw allow proto tcp from [IP host] to any port 12798

netstat -tulpn # test that the ports are open and being listened
```

21.1.3 Relay (host) part II

```
cd $CNODE_HOME

cd ..

cd monitoring

cd Prometheus

sudo nano prometheus.yml

# Add in scrape_configs

        - job_name: 'VIMinerII_cardano_node'
          static_configs:
          - targets: ['[Local IP BP]:12798']
            labels:
              instance: "VIMinerII_cnode"
        - job_name: '[BP_NAME]_node_exporter'
          static_configs:
          - targets: ['[Local IP BP]:9091']
            labels:
              instance: "[BP_NAME]_node_exporter"

sudo systemctl restart Prometheus

sudo systemctl status prometheus

sudo systemctl restart node_exporter.service
```

22 Miscellaneous commands

22.1 Check for missing slots

```
curl localhost:12798/metrics | grep
"cardano_node_metrics_slotsMissedNum_int"
```

22.2 Logging (Debug)

```
journalctl -e -f -u cnode.service
```