Masternode Guide #2

Single masternode on Linux VPS (Ubuntu)+ control wallet on local PC (Windows)

Prerequisites:
a - A remote server (Virtual Private Server, VPS) which will be our masternode wallet.
b - A local computer running under Windows 7, 8.1 or 10 which will be our control wallet.
c - PuTTY, which will be used to setup the server (install the dependencies, the wallet itself, and configure everything) after the initial configuration.
d - 1001 COINS as collateral (1000 COINS + 1 COIN to cover the transaction fees)

The server configuration is bare minimum 1 CPU and 1 GB of RAM.
This is enough to run the wallet but might not be enough to compile it.
To compile the wallet you need 2GB of ram or if you have a 1GB RAM server you need to create a Swap file of 1GB.
Detailed instructions on how to do it are provided further on.

Create a Swap file:

```
sudo fallocate -l 2G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile

sudo nano /etc/fstab
```

Add at the end of file

```
/swapfile none swap sw 0 0
```

You don't have to reboot the system for the changes to take effect - the following command will do:

```
mount -o remount /
```

That's it. You can run

```
mount
```

Can check free disk space and free memory (swap also)

```
df -h
free -h
```

Create second user and disable root login:

For better security let’s create second user (non root) with max permissions

```
adduser sparks1 && adduser sparks1 sudo
```

Exit and login with new user

```
exit
```

Diable root login. Edit **sshd_config**:

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

find and change parameter `PermitRootLogin` from **yes** to **no**

Then restart sshd:

```
sudo systemctl restart sshd.service
```

Setup firewall:

Check if firewall installed and enabled:

```
sudo ufw status
```

If not – install firewall (on Debian by default is missing)

```
sudo apt-get install ufw
```

Then disable firewall for further setup:

```
sudo ufw disable
```

Lets allow all needed:

```
sudo ufw default allow outgoing
sudo ufw default deny incoming
sudo ufw allow ssh/tcp
sudo ufw limit ssh/tcp
# sudo ufw allow http/tcp
# sudo ufw allow https/tcp
sudo ufw allow 8890/tcp
sudo ufw logging on
sudo ufw enable
```

Check status

```
sudo ufw status
```

Install node:

Can be done with below commands

```
sudo apt-get update -y
sudo apt-get upgrade -y
sudo apt-get install build-essential libtool autotools-dev autoconf pkg-config
libssl-dev -y
sudo apt-get install software-properties-common -y
sudo add-apt-repository ppa:bitcoin/bitcoin -y
sudo apt-get update -y
sudo apt-get install libdb4.8-dev libdb4.8++-dev -y
sudo apt-get install libboost-all-dev -y
sudo apt-get install libminiupnpc-dev –y
```

Or use one line

```
sudo apt-get update -y && sudo apt-get upgrade -y && sudo apt-get install build-
essential libtool autotools-dev autoconf pkg-config libssl-dev -y && sudo apt-get
install software-properties-common -y && sudo add-apt-repository ppa:bitcoin/bitcoin
-y && sudo apt-get update -y && sudo apt-get install libdb4.8-dev libdb4.8++-dev -y
&& sudo apt-get install libboost-all-dev -y && sudo apt-get install libminiupnpc-dev
-y && sudo apt-get install git automake libevent-dev -y
```

Once we have all dependencies we can download and compile the wallet:

```
sudo apt-get install git automake libevent-dev -y
cd
git clone https://github.com/sparkscrypto/Sparks.git
cd Sparks
sudo ./autogen.sh
sudo ./configure
sudo make
```

Last command can take awhile and some warning messages will be shown it's perfectly normal.

After compilation:

```
cd src
strip Sparks-tx
strip Sparks-cli
strip Sparksd
mv Sparksd Sparks-cli Sparks-tx ~/
cd ~/
ls
sudo rm -r Sparks
```

Lets start daemon

```
cd ~/
./Sparksd -daemon
./Sparks-cli getinfo
./Sparks-cli masternode status
watch ./Sparks-cli getinfo
./Sparks-cli stop
```

If you haven't generate masternode key yet (for some reason – waiting for 1000 on collocation) – can generate on VPS and use then on local wallet

Or generate on local wallet using debug console and command

```
masternode genkey
```

Edit ~/.Sparks/Sparks.conf

```
sudo nano ~/.Sparks/Sparks.conf
```

Paste into Sparks.conf (replace with your data - rpcuser, rpcpassword, externalip and masternodeprivkey)

```
rpcuser=randomusername
rpcpassword=randompass
rpcallowip=127.0.0.1
rpcport=8818
listen=1
server=1
daemon=1
logtimestamps=1
maxconnections=256
masternode=1
externalip=8.8.8.8
masternodeprivkey=7fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx5po
addnode=91.234.32.241:8890
addnode=91.201.42.245:8890
addnode=194.87.237.57:8890
addnode=178.2.46.161:8890
addnode=107.175.59.50:8890
addnode=195.181.243.145:8890
addnode=96.21.125.11:8890
addnode=45.76.252.179:8890
addnode=52.178.24.171:8890
addnode=107.172.196.13:8890
addnode=77.51.183.48:8890
addnode=185.221.152.38:8890
addnode=45.118.151.57:8890
addnode=136.144.167.226:8890
addnode=83.217.206.155:8890
addnode=82.223.43.175:8890
addnode=80.208.228.219:8890
addnode=80.208.225.163:8890
addnode=104.192.102.155:8890
addnode=209.250.235.168:8890
addnode=209.250.235.168:8890
addnode=51.15.84.213:8890
addnode=45.76.89.64:8890
addnode=80.209.230.134:8890
addnode=91.201.42.222:8890
```

```
addnode=213.136.83.147:8890
addnode=195.154.134.142:8890
addnode=163.172.68.78:8890
addnode=62.210.248.84:8890
addnode=37.215.112.178:8890
addnode=188.113.136.160:8890
```

SENTINEL INSTALLATION GUIDE – FOR SPARKS MASTERNODES in Linux

You must install Sentinel. Sentinel is a watchdog for your node which communicates to the network that your node is working properly.
Before starting the steps make sure to restart your masternode with reindex so you are using the fresh blockchain.
Delete mncache.dat and mnpayments.dat from your Sparkscore folder before reindexing.
// Your Sparkscore folder

```
cd
cd .Sparks
~/Sparks-cli stop
rm mncache.dat
rm mnpayments.dat
cd
./Sparksd -daemon -reindex
```

Wait for the full resync to the latest block. You can check regularly with

```
watch ~/Sparks-cli getinfo
```

Once your wallet is fully synced you are now ready to follow the steps below. If you are running your masternodes on Linux you can now skip to Step 1) and start installing prerequisites.
The guide below covers all areas for Linux users in detail, which represents over 90% of total masternode owners.

Step 1) Install the prerequisites

```
sudo apt-get update && sudo apt-get install -y git python-virtualenv
```

Step 2) If you are not already there, navigate to your .Sparks folder

```
cd ~/.Sparks
```

Step 3) Clone sentinel, switch to the sentinel directory

```
git clone https://github.com/sparkscrypto/sentinel.git
cd sentinel
```

Step 4) Create virtual python environment

```
virtualenv venv
```

If this command fails try installing this package:):

```
sudo apt-get install -y virtualenv
```

Step 5) Install sentinel dependencies

```
venv/bin/pip install -r requirements.txt
```

Step 6) Test sentinel is alive and talking to the still syncing wallet
Open sentinel.conf:

```
nano sentinel.conf
```

We have to add path to Sparks.conf. Right now it shows path to dash.
Uncomment the #dash_conf line at the top

```
#dash_conf=/home/evan82/.dashcore/dash.conf
```

and add the path to your MN's Sparks.conf.

```
dash_conf=/home/sparks1/.Sparks/Sparks.conf
```

If you cant find path to Sparks.conf. Go to .Sparks folder and type pwd. You wil get the right path. Copy it and paste in sentinel.conf. Save the file then close it.

If you get a error. Check path first, if path ok. Check do you have rpc user and rpc password in Sparks.conf. If you dont have rpc user with password. Stop Sparks first.

```
~/Sparks-cli stop
```

Check for exist or put (add / change) in Sparks.conf.

```
sudo nano ~/.Sparks/Sparks.conf

rpc=anyusername
rpcpassword=anypassword
server=1
rpcport=8818
rpcallowip=127.0.0.1
```

Then restart a wallet.

```
cd
./Sparksd -daemon
```

Now run:

```
cd ~/.Sparks/sentinel
venv/bin/python bin/sentinel.py
```

You should see: “Sparksd not synced with network! Awaiting full sync before running Sentinel.”
This is exactly what we want to see at this stage

Step 7) Wait until the reindex has complete and the wallet has sync’d

```
cd
~/Sparks-cli masternode status
```

This is what you’re waiting to see:
AssetId 999, all trues, one false, and a FINISHED. Keep issuing

```
watch ~/Sparks-cli mnsync status
```

until it looks like this:

```
{
   “AssetID”: 999,
   “AssetName”: “MASTERNODE_SYNC_FINISHED”,
   “Attempt”: 0,
   “IsBlockchainSynced”: true,
   “IsMasternodeListSynced”: true,
   “IsWinnersListSynced”: true,
   “IsSynced”: true,
   “IsFailed”: false
}
```

At this point, your remote masternode is synchronized and chatting with the network
but is not accepted as a masternode because it hasn’t been introduced to the network
by your collateral.

Step 8) You may now start your masternode! Go back to your local wallet, open the debug console, and run these commands (LABEL is the name you used for your MN in the masternode.conf):

```
walletpassphrase 120
masternode start-alias
```

or just press “Start Alias” Button in “My Masternode” tab (hopefully :)) start your masternode.

Step 9) Test sentinel has nothing bad to say
You're needed back in Sentinel directory!

```
	cd ~/.Sparks/sentinel
```

At this point, running

```
	venv/bin/python bin/sentinel.py
```

should return nothing but silence. This is how you know it's working, and your masternode is working.

Step 10) Create a crontab entry to wake sentinel every five minutes

```
	crontab -e
```

Choose Joe as your editor. Add this line to the end of the file.

```
	* * * * * cd /home/sparks1/.Sparks/sentinel && ./venv/bin/python bin/sentinel.py
2>&1 >> sentinel-cron.log
```

Make sure you:
1) Change USERNAME to your username.
2) Hit enter to create another line at the end after this line, or the file will not work.
Press Control-X to save and exit.

Step 10) Check for start, try, try, again

```
	cd
```

You should be back in .Sparks.

```
	~/Sparks-cli masternode debug

	~/Sparks-cli getinfo

	~/Sparks-cli masternode status
```

Did it work? You should see the message "Masternode successfully started.". Congratulations!
You can find more general documentation on Sentinel here. The simple, beginner-friendly sentinel guide has been originally contributed in the Dash KB by Tao.

https://github.com/dashpay/sentinel