Masternode Setup

Setting up a masternode requires a basic understanding of Linux and blockchain technology, as well as an ability to follow instructions closely. It also requires regular maintenance and careful security, particularly if you are not storing your MOGs on a hardware wallet. There are some decisions to be made along the way, and optional extra steps to take for increased security.

Before you begin

This guide assumes you are setting up a masternode for the first time. You will need:

- 1000 MOGs
- A wallet to store your MOG, either a hardware wallet or Mogwaicoin Core wallet
- A Linux server, preferably a Virtual Private Server (VPS)

This guide also assumes you will be working from a Windows computer. However, since most of the work is done on your Linux VPS, alternative steps for using macOS or Linux will be indicated where necessary.

Set up your VPS

A VPS, more commonly known as a cloud server, is fully functional installation of an operating system (usually Linux) operating within a virtual machine. The virtual machine allows the VPS provider to run multiple systems on one physical server, making it more efficient and much cheaper than having a single operating system running on the "bare metal" of each server. A VPS is ideal for hosting a Mogwaicoin masternode because they typically offer guaranteed uptime, redundancy in the case of hardware failure and a static IP address that is required to ensure you remain in the masternode payment queue. While running a masternode from home on a desktop computer is technically possible, it will most likely not work reliably because most ISPs allocate dynamic IP addresses to home users.

We will use Vultr hosting as an example of a VPS, although DigitalOcean, Amazon EC2, Google Cloud, Choopa and OVH are also popular choices. First create an account and add credit. Then go to the Servers menu item on the left and click + to add a new server. Select a location for your new server on the following screen:
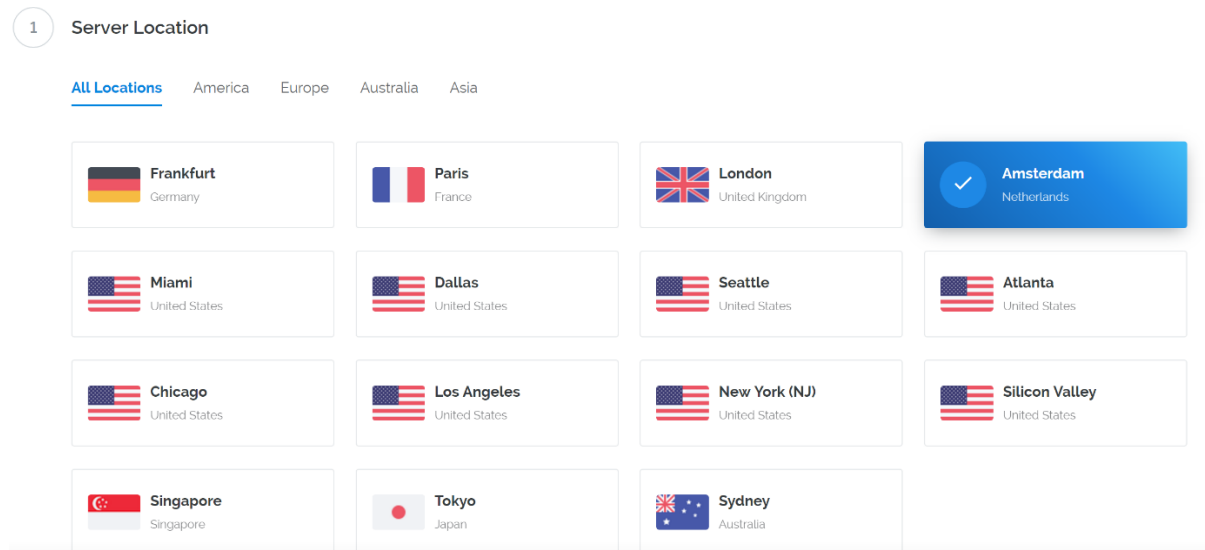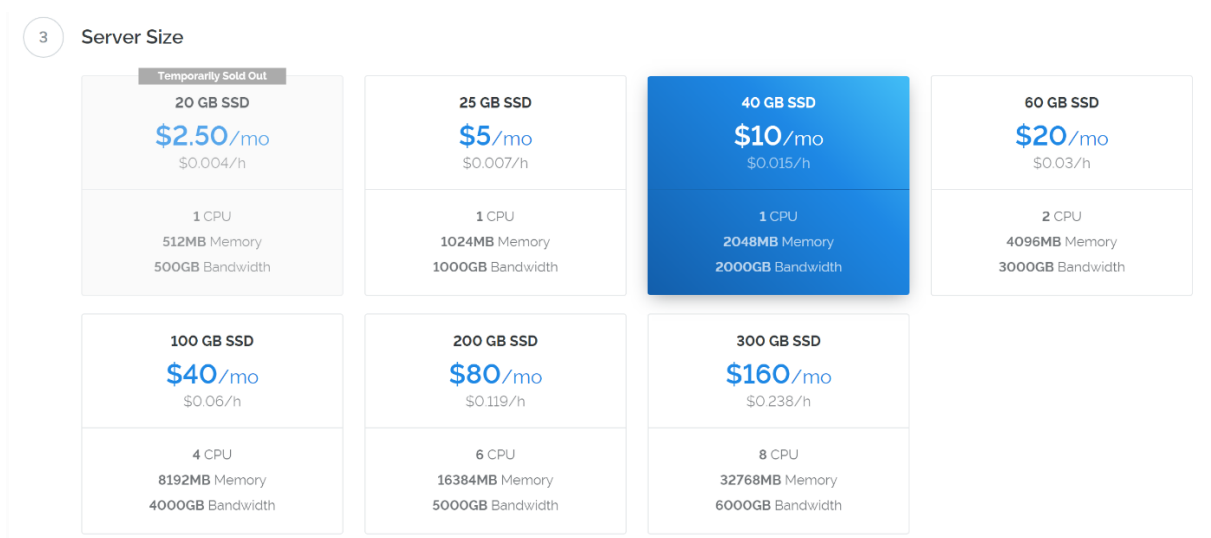

Vultr server location selection screen

Select Ubuntu 16.04 x64 as the server type. We use 16.04 instead of the latest version because 16.04 is an LTS release of Ubuntu, which will be supported with security updates for 5 years instead of the usual 9 months. (18.04 x64 can also be used)
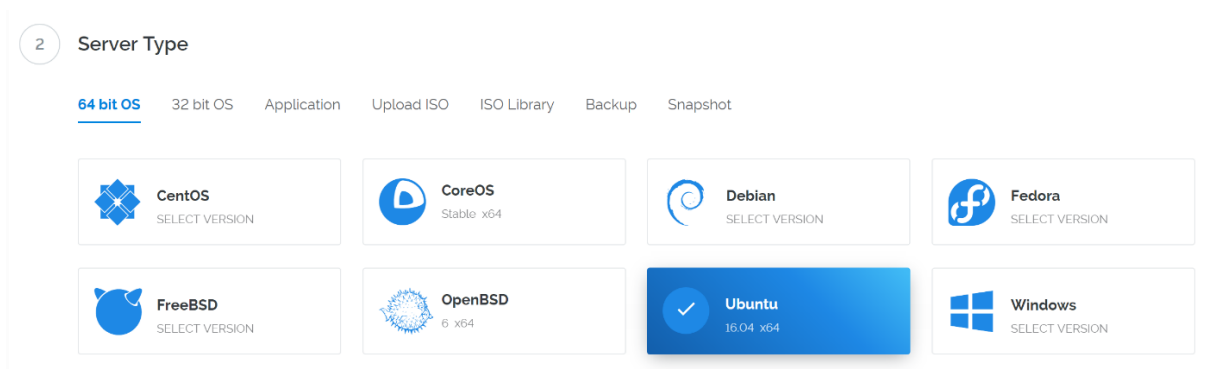
Vultr server type selection screen

Select a server size offering at least 2GB of memory.

Vultr server size selection screen

Enter a hostname and label for your server. In this example we will use `mogwaimn1` as the hostname.

Click **Manage** when installation is complete and take note of the IP address, username and password.

Vultr server management screen

Set up your operating system

We will begin by connecting to your newly provisioned server. On Windows, we will first download an app called PuTTY to connect to the server. Go to the PuTTY download page here and select the appropriate MSI installer for your system. On Mac or Linux you can ssh directly from the terminal - simply type `ssh root@<server_ip>` and enter your password when prompted.

Download PuTTY: latest release (0.69)

Home | FAQ | Feedback | Licence | Updates | Mirrors | Keys | Links | Team
Download: **Stable** · Snapshot | Docs | Changes | Wishlist

This page contains download links for the latest released version of PuTTY. Currently this is 0.69, released on 2017-04-29.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternatively, here is a permanent link to the 0.69 release.

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date version of the code available. If you have a problem with this release, then it might be worth trying out the development snapshots, to see if the problem has already been fixed in those versions.

Package files

You probably want one of these. They include all the PuTTY utilities.

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

MSI ('Windows Installer')

32-bit:	putty-0.69-installer.msi	(or by FTP)	(signature)
64-bit:	putty-64bit-0.69-installer.msi	(or by FTP)	(signature)

Unix source archive

|			
.tar.gz:	putty-0.69.tar.gz	(or by FTP)	(signature)

PuTTY download page

Double-click the downloaded file to install PuTTY, then run the app from your Start menu. Enter the IP address of the server in the Host Name field and click Open. You may see a certificate warning, since this is the first time you are connecting to this server. You can safely click Yes to trust this server in the future.

PuTTY Security Alert ✕

⚠ WARNING - POTENTIAL SECURITY BREACH!

The server's host key does not match the one PuTTY has cached in the registry. This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new ssh-ed25519 key fingerprint is:
ssh-ed25519 256 e1:b2:f6:64:bf:dc:b4:9a:ee:66:78:a5:05:0d:dd:e8
If you were expecting this change and trust the new key, hit Yes to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating the cache, hit No.
If you want to abandon the connection completely, hit Cancel. Hitting Cancel is the ONLY guaranteed safe choice.

| Yes | No | Cancel | Help |

PuTTY security alert when connecting to a new server

You are now connected to your server and should see a terminal window. Begin by logging in to your server with the user `root` and password supplied by your hosting provider.
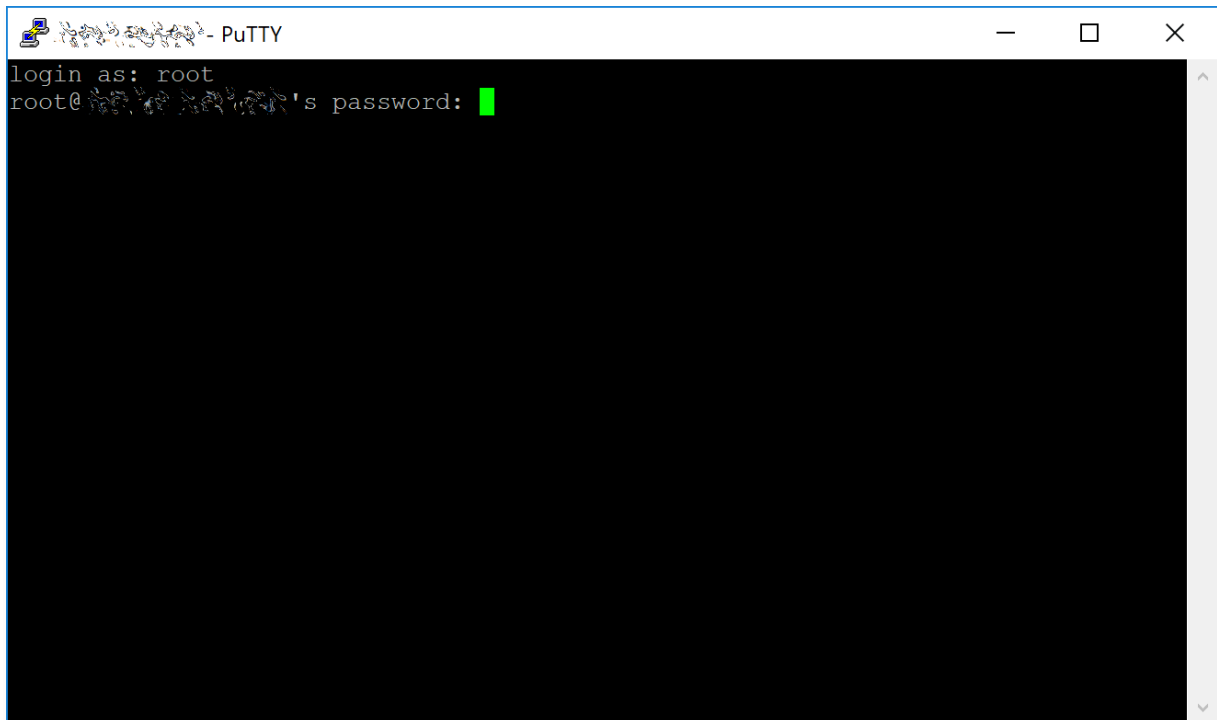
Password challenge when connecting to your VPS for the first time

You should immediately change the root password and store it in a safe place for security. You can copy and paste any of the following commands by selecting them in your browser, pressing Ctrl + C, then switching to the PuTTY window and right-clicking in the window. The text will paste at the current cursor location:

```
passwd root
```

Enter and confirm a new password (preferably long and randomly generated). Next we will create a new user with the following command, replacing `<username>` with a username of your choice:

```
adduser <username>
```

You will be prompted for a password. Enter and confirm using a new password (different to your root password) and store it in a safe place. You will also see prompts for user information, but this can be left blank. Once the user has been created, we will add them to the sudo group so they can perform commands as root:

```
usermod -aG sudo <username>
```

Now, while still as root, we will update the system from the Ubuntu package repository:

```
apt update
apt upgrade
```

The system will show a list of upgradable packages. Press Y and Enter to install the packages. We will now install a firewall (and some other packages we will use later), add swap memory and reboot the server to apply any necessary kernel updates, and then login to our newly secured environment as the new user:

```
apt install ufw python virtualenv git unzip pv
```

(press Y and Enter to confirm)

```
ufw allow ssh/tcp
ufw limit ssh/tcp
ufw allow 9999/tcp
ufw logging on
ufw enable
```

(press Y and Enter to confirm)

```
fallocate -l 4G /swapfile
chmod 600 /swapfile
mkswap /swapfile
swapon /swapfile
nano /etc/fstab
```
Add the following line at the end of the file (press tab to separate each word/number), then press Ctrl + X to close the editor, then Y and Enter save the file.

```
/swapfile none swap sw 0 0
```
Then reboot the server:

```
reboot now
```
PuTTY will disconnect when the server reboots.

While this setup includes basic steps to protect your server against attacks, much more can be done. In particular, authenticating with a public key instead of a username/password combination, installing fail2ban to block login brute force attacks, disabling root login and enabling automatic security updates is advisable. More tips are available here. However, since the masternode does not actually store the keys to any Mogwaicoins, these steps are considered beyond the scope of this guide.

Send the collateral
A Mogwaicoin address with a single unspent transaction output (UTXO) of exactly 1000 MOGs is required to operate a masternode. Once it has been sent, various keys regarding the transaction must be extracted for later entry in a configuration file as proof that the transaction was completed successfully. A masternode can be started from a hardware wallet or the official Mogwaicoin Core wallet, although a hardware wallet is highly recommended to enhance security and protect yourself against hacking. This guide will describe the steps for both hardware wallets and Mogwaicoin Core.

Sending from Mogwaicoin Core wallet
Open Mogwaicoin Core wallet and wait for it to synchronize with the network. It should like this when ready:
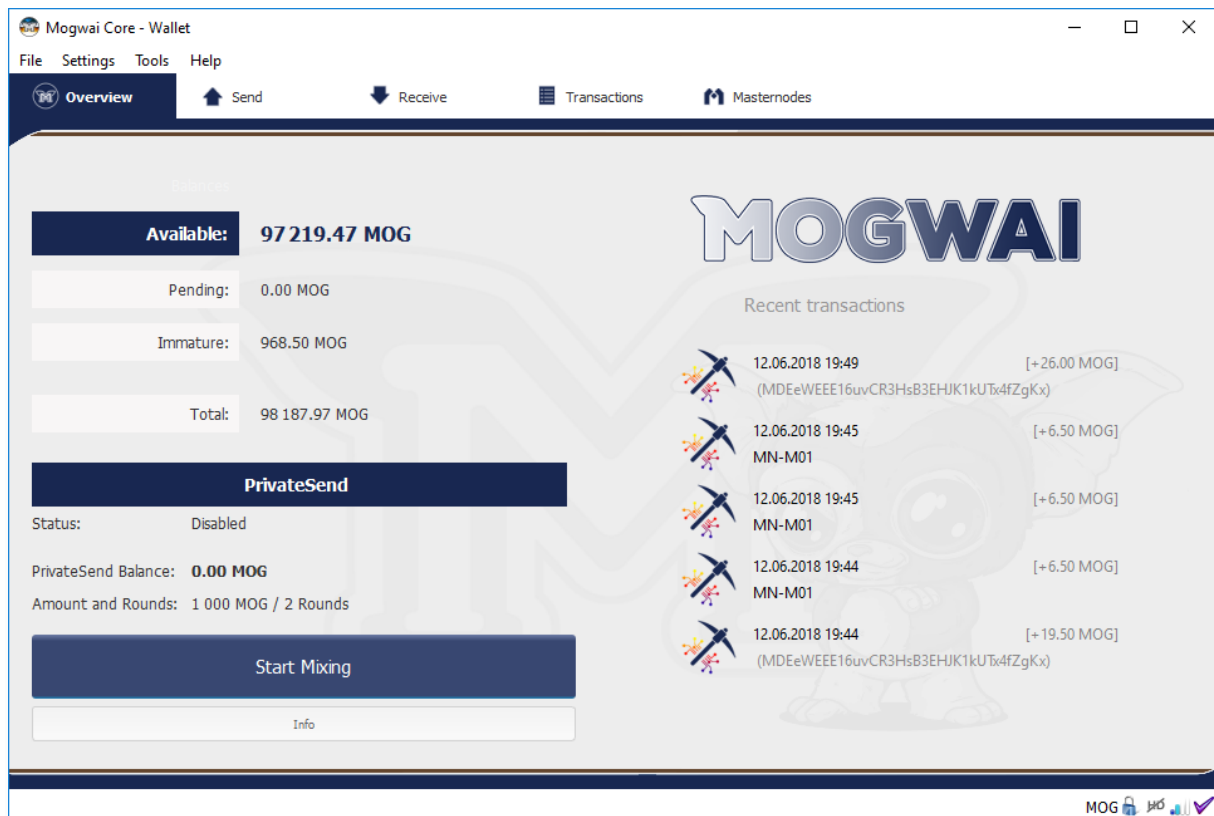
Fully synchronized Mogwaicoin Core wallet

Click **Tools > Debug console** to open the console. Type the following two commands into the console to generate a masternode key and get a fresh address:

```
masternode genkey
getaccountaddress 0
```

Generating a masternode private key in Mogwaicoin Core wallet

Take note of the masternode private key and collateral address, since we will need it later. The next step is to secure your wallet (if you have not already done so). First, encrypt the wallet by selecting **Settings > Encrypt wallet**. You should use a strong, new password that you have never used somewhere else. Take note of your password and store it somewhere safe or you will be permanently locked out of your wallet and lose access to your funds. Next, back up your wallet file by selecting **File > Backup Wallet**. Save the file to a secure location physically separate to your computer, since this will be the only way you can access our funds if anything happens to your

computer. For more details on these steps, see here.
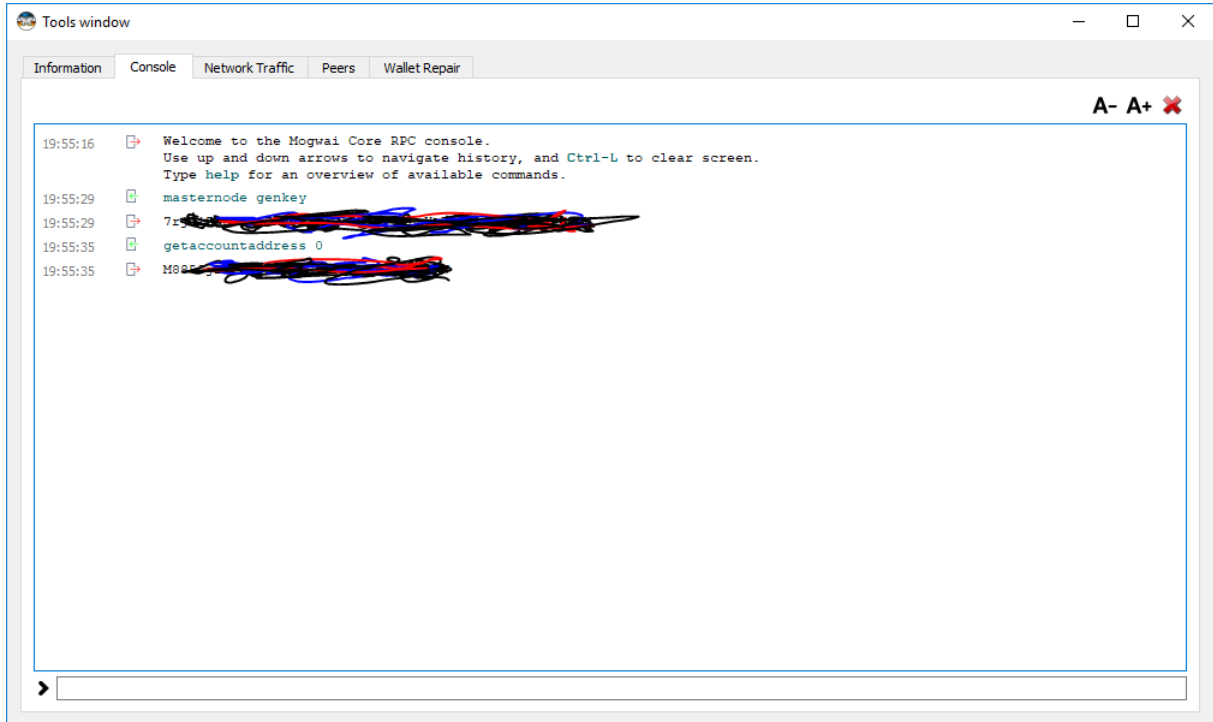
Now send exactly 1000 MOGs in a single transaction to the account address you generated in the previous step. This may be sent from another wallet, or from funds already held in your current wallet. Once the transaction is complete, view the transaction in a blockchain explorer by searching for the address. You will need 15 confirmations before you can start the masternode, but you can continue with the next step at this point already: installing Mogwaicoin Core on your VPS.

Install Mogwaicoin Core

Mogwaicoin Core is the software behind both the Mogwaicoin Core GUI wallet and Mogwaicoin masternodes. If not displaying a GUI, it runs as a daemon on your VPS (mogwaid), controlled by a simple command interface (mogwai-cli).

Open PuTTY or a console again and connect using the username and password you just created for your new, non-root user.

Manual installation

To manually download and install the components of your Mogwaicoin masternode, Login to your VPS server with root user. Make sure you are in root directory:

```
cd /root
```
Download installing script file

```
apt-get install wget -y
wget https://raw.githubusercontent.com/mogwaicoin/sentinel/master/install_masternode.sh
```
Make the script executable

```
chmod 740 install_masternode.sh
```
Run the script to install

```
./install_masternode.sh
```

Now it will take some time. Wait until the following is printed to the console: «Job completed successfully». It will also print masternode private key as: «Masternode private key:». Save that somewhere, you will use it later.

Note, if you happen to see nothing after Masternode private key: then it failed to generate private key. In that case follow these steps: (Skip Step a – d, if you see private key).

a) Generate key:

```
mogwai-cli masternode genkey
```
b) Copy the printed key and open the config file:

```
nano /root/.mogwaicore/mogwai.conf
```
c) Paste the copied key in a newline after masternodeprivkey=

d) Save and close the file

```
CTRL+X → Y → ENTER
```

Remove the script

```
rm install_masternode.sh
```

You can now start running Mogwaicoin on the masternode to begin synchronization with the blockchain:

```
~/mogwai/mogwaid
```

You will see a message reading Mogwaicoin Core server starting.

Starting from Mogwaicoin Core wallet

If you used an address in Mogwaicoin Core wallet for your collateral transaction, you now need to find the txid of the transaction. Click **Tools > Debug console** and enter the following command:

```
masternode outputs
```
This should return a string of characters similar to this:

```
{
"06e38868bb8f9958e34d5155437d009b72dff33fc28874c87fd42e51c0f74fdb" : "0",
}
```

The first long string is your transaction hash, while the last number is the index. We now need to create a file called *masternode.conf* for this wallet in order to be able to use it to issue the command to start your masternode on the network. Open a new text file in Notepad (or TextEdit on macOS, gedit on Linux) and enter the following information:

- **Label**: Any single word used to identify your masternode, e.g. MN1
- IP and port: The IP address and port (usually 9999) configured in the mogwai.conf file, separated by a colon (:)

- **Masternode private key**: This is the result of your masternode genkey command earlier, also the same as configured in the mogwai.conf file
- **Transaction hash**: The txid we just identified using masternode outputs
- **Index**: The index we just identified using masternode outputs

Enter all of this information on a single line with each item separated by a space, for example:

```
MN1 52.14.2.67:9999 XrxSr3fXpX3dZcU7CoiFuFWqeHYw83r28btCFfIHqf6zkMp1PZ4
06e38868bb8f9958e34d5155437d009b72dff33fc28874c87fd42e51c0f74fdb 0
```

Save this file in the MogwaiCore data folder on the PC running the Mogwaicoin Core wallet using the filename *masternode.conf*. You may need to enable View hidden items to view this folder. Be sure to select All files if using Notepad so you don't end up with a *.conf.txt* file extension by mistake. For different operating systems, the MogwaiCore folder can be found in the following locations (copy and paste the shortcut text into the Save dialog to find it quickly):

Platform	Path	Shortcut
Linux	`/home/yourusername/.mogwaicore`	`~/.mogwaicore`
macOS	`/Macintosh HD/Library/Application Support`	`~/Library/Application Support/MogwaiCore`
Windows	`C:\Users\yourusername\AppData\Roaming\Mogwaicoin Core`	`%APPDATA%\MogwaiCore`

Now close your text editor and also shut down and restart Mogwaicoin Core wallet. Mogwaicoin Core will recognize masternode.conf during startup, and is now ready to activate your masternode. Go to Settings > Unlock Wallet and enter your wallet passphrase. Then click Tools > Debug console again and enter the following command to start your masternode (replace MN1 with the label for your masternode):

```
masternode start-alias MN1
```

At this point you can go back to your terminal window and monitor your masternode by entering `~/mogwai/mogwai-cli masternode status`. You will probably need to wait around 30 minutes as the node passes through the PRE_ENABLED stage and finally reaches ENABLED. Give it some time, the final result should appear as follows:

At this point you can safely log out of your server by typing `exit`. Congratulations! Your masternode is now running.