Setting up a Chaincoin Masternode

Introduction

So you want to set up your own Chaincoin Masternode? You've come to the right place!

These instructions are correct as of April, 2017, and relate to version 0.9 of chaincoin.

We're going to be setting the masternode up on a Ubuntu server, and assume you're running Windows.

Here's what you're going to need:

- A server running Ubuntu Linux. In this example, we're going to rent one from Vultr.com. You can use your own, but note that we use a slightly older build of Ubuntu (14.04) for compatibility, so if you're using a newer version you may encounter some issues.
- A static IP address
- Basic UNIX or Linux skills – you're going to be compiling code, potentially dealing with errors, editing files, connecting via SSH etc.

Please also take our security suggestions seriously – you're dealing with currency here, so you want to ensure you're using strong passwords you don't use anywhere else, that your personal computer has an up-to-date antivirus etc.

This is a draft! If you find any issues with this guide, or have suggestions or corrections, please send them to the team on Slack.

1. Setting up a Ubuntu server on Vultr

This is our recommendation for hosting your Masternode server. It's only $5-$6 per month and is hands-off once you've set it up. Vultr is a popular Cloud Computing SSD host for Masternodes.

1. Create an account on vultr.com
2. Choose the option to Deploy a Server, as follows:
   a. Choose a region near your location
   b. Choose a 64 bit OS
   c. Select Ubuntu 14.04 x64
   d. Server Size should be 25GB SSD or higher
   e. We recommend enabling auto backups for an additional $1.00/month. If you want DDOS protection it's an extra $10.00/month – that's up to you.
   f. Give the server a hostname, and a label to identify it in your Vultr account.
3. The server will take a few minutes to deploy and will be available in the "Servers" list on the website.
4. Once your server is set up, you can see it by clicking the Servers menu.
   a. To connect to the server via the console, click the … menu to the right of the Status and you'll see the option to view the Console.
   b. To retrieve the root password, click the server instance and on the Server Information page, under Overview, you'll see the password, which you should make a note of and keep somewhere secure.
5. Make a note of the server's IP address so you can connect to it later.

2. Configuring and securing the server

We recommend you secure your server as soon as possible. In this section, we set up a more secure way to access your server and remove the normal login and password approach for logging in, and we also set up the firewall.

1. Log in to your server from your computer, using SSH:
    a. If you're using Windows, you'll need to download and install a set of tools called PuTTY from http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html These will allow you to connect to your server remotely using a secure protocol known as SSH. Download and install the 32 or 64 bit MSI Windows Installer depending on your computer and set it up. PuTTY is installed by default to C:\Program Files\PuTTY and you might want to make a shortcut to putty.exe on your desktop.
        i. Run putty.exe
        ii. Select the "Session" category on the left, if it's not selected
        iii. In the Host Name field, enter your server IP address and set the port to 22
        iv. Make sure the Connection Type is SSH
        v. Type a name for this connection in the "Saved Sessions" filed and click Save. Now you can load these settings again future by selecting them and clicking Load.
        vi. Click the "Open" button. You should be connected to your server and prompted to login with "root" and the root password you noted when you set the server up.
    b. If you're on Mac or Linux, you should be able to connect to your server from the Terminal or command prompt using the syntax:

        ```
        ssh root@your_server_ip
        ```

2. You may be asked about host authenticity if this is the first time you've logged in, so you can answer 'Yes'. You might also be prompted to change your password after logging in, which is also a good idea.
3. To change your root password, if you are logged in as root, type:

    ```
    passwd
    ```

    And enter your new password twice when prompted. Don't forget it!
4. The root account is too risky for day-to-day use, so let's create a new user who can access root powers as needed but is safer to use.
    a. Add a new user with this command:

        ```
        adduser newusername
        ```

    b. You'll be prompted for a password. Make sure it's strong, you don't use it anywhere else, and you keep a copy of it somewhere safe.
    c. You'll be asked some more questions which you can answer or just skip by pressing enter
    d. Now let's add this user to the **sudo** group so where necessary, they can issue commands with superuser privileges:

        ```
        usermod -aG sudo newusername
        ```

    e. Logout of the server and try logging in as this new user to make sure the account is working correctly.

5. A more secure way to connect to your server than via login and password is to use Public Key Authentication. This involves creating a secure connection between your computer and the server using key files stored on both machines.

   On Mac/UNIX/Linux:

   a. From the Terminal, run this command to create a key pair:

      ```
      ssh-keygen
      ```

   b. Press enter to save the keys in the default location /Users/*username*/.ssh/id_rsa (leave the name as id_rsa unless you need to rename it because you have other keys there)
   c. Next, you will be prompted for a passphrase to secure the key with. You can leave the passphrase blank. If you leave the passphrase blank, you will be able to use the private key for logging in without entering a passphrase. If you enter a passphrase, you will need both the private key and the passphrase to log in.
   d. Now you have a private key, id_rsa, and a public key, id_rsa.pub, in the .ssh directory of your home directory. Remember to keep the private key secure.
   e. Now you need to copy the public key to your Chaincoin Masternode server.
      i. Make a copy of your public key by displaying it like this:

         ```
         cat ~/.ssh/id_rsa.pub
         ```

      ii. Copy the line in the output starting with **ssh-rsa …**
      iii. Log in to the Masternode server as the root user
      iv. Switch to the user you created earlier like this:

         ```
         su - newusername
         ```

      v. Make a directory to hold the key (if it doesn't already exist) like this:

         ```
         mkdir ~/.ssh
         ```

         ```
         chmod 700 ~/.ssh
         ```

      vi. Make a file inside the directory called authorized_keys and paste in the key from your local machine:

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

      vii. Hit CTRL-x to exit the file, then y to save the changes that you made, then ENTER to confirm the file name.
      viii. Now you need to restrict the permissions of the authorized_keys file with this command:

         ```
         chmod 600 ~/.ssh/authorized_keys
         ```

      ix. Then return to the root user:

         ```
         exit
         ```

      x. Logout

         ```
         logout
         ```

      xi. Now try and connect via SSH:

```
ssh newusername@your_server_ip
```

xii. You shouldn't be prompted for your password. However, if you set a passphrase for your key, you should be prompted for that.

On Windows:

a. Run the PuTTYgen utility at C:\Program Files\PuTTY\puttygen.exe
b. At the bottom, in the **Type of key to generate** section, select **RSA**
c. Click the **Generate** button
d. Move your mouse pointer around in the blank area of the **Key** section as prompted while the key is being generated.
e. Once complete, add your email address to the key comment field to help you identify the key
f. Optionally enter a passphrase in the **Key passphrase** field. If you leave the passphrase blank, you will be able to use the private key for logging in without entering a passphrase. If you enter a passphrase, you will need both the private key and the passphrase to log in.
g. Click **Save public key** and save the key somewhere safe on your computer.
h. Click **Save private key** and save the key somewhere safe on your computer – it can be the same place as the public key. Remember to keep the private key secure. If you lose it, you won't be able to log in to your server.
i. Now you need to copy the public key to your Chaincoin Masternode server.
    i. Right-click in the text field labeled **Public key for pasting into OpenSSH authorized_keys file** and choose Select All
    ii. Copy the selected text, which should start with **ssh-rsa** …
    iii. Log in to the Masternode server as the root user
    iv. Switch to the user you created earlier like this:

    ```
    su - newusername
    ```

    v. Make a directory to hold the key (if it doesn't already exist) like this:

    ```
    mkdir ~/.ssh
    chmod 700 ~/.ssh
    ```

    vi. Make a file inside the directory called **authorized_keys** and paste in the key from your local machine:

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

    vii. Hit CTRL-x to exit the file, then y to save the changes that you made, then ENTER to confirm the file name.
    viii. Now you need to restrict the permissions of the authorized_keys file with this command:

    ```
    chmod 600 ~/.ssh/authorized_keys
    ```

    ix. Then return to the root user:

    ```
    exit
    ```

    x. Logout

    ```
    logout
    ```

xi. Now let’s setup the key in PuTTY:
   1. Start PuTTY
   2. In the Host Name field, enter the IP address of your server
   3. Ensure the port number in the Port field is 22
   4. Select SSH under Protocol
   5. In the left panel of PuTTY, select the Data sub-category, under Connection
   6. Specify the username that you plan on using in the Auto-login username field (i.e. what you used for *newusername*)
   7. Expand the SSH sub-category, under Connection
   8. Highlight the Auth sub-category and click the Browse button, on the right-hand side of the PuTTY window;
   9. Browse your file system and select your previously-created private key (the filename ends in .ppk)
   10. Return to the Session Category and enter a name for this profile in the Saved Sessions field, e.g. newusername@123.456.78.9 newusername CHC Masternode
   11. Click the Save button
   12. Click the Open button to log in to your server and you will not be prompted for a password. However, if you have set a passphrase on your public key, you will be asked to enter the passphrase.

2. Now that you can log in to your server using SSH, let’s remove the ability to log in via username and password, which is less secure.
   a. Log in to your server as *newusername* and open the SSH daemon configuration:

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

   b. Find the line that specifies **PasswordAuthentication**, uncomment it by deleting the # at the start of the line, then change its value to "no". It should end up looking like this:

   ```
   PasswordAuthentication no
   ```

   c. Find, uncomment and change these two values as well:

   ```
   PubkeyAuthentication yes

   ChallengeResponseAuthentication no
   ```

   d. When you are finished making your changes, save and close the file using CTRL-X, then Y, then ENTER
   e. Type this to reload the SSH daemon:

   ```
   sudo /etc/init.d/ssh restart
   ```

   f. Password authentication is now disabled. Your server is now only accessible with SSH key authentication.
   g. Test you can still login.
3. As the final step before we install the Chaincoin Masternode, let’s Setup the firewall. It needs to allow access to OpenSSH so you can login, as well as access to Masternode for clients to connect.
   a. Login to your server as *newusername*

b. Enter these commands:

   ```
   sudo ufw allow OpenSSH
   sudo ufw allow 8333
   sudo ufw allow 11994
   sudo ufw default deny incoming
   sudo ufw default allow outgoing
   sudo ufw enable
   ```

c. This configures and turns on the firewall.
d. You should log out and confirm you can still log in.

3. Compiling the chaincoin masternode

1. Login to your server as *newusername*
2. We need to create a swap drive, as compilations fails with a memory error otherwise – this Vultr server doesn't quite have enough memory to compile the Masternode. The free command shows you how much free memory you have:

   ```
   free
   ```

   Run these commands to make a swapfile:

   ```
   sudo dd if=/dev/zero of=/var/swap.img bs=1024k count=1000
   sudo mkswap /var/swap.img
   sudo swapon /var/swap.img
   ```

   Run **free** again to confirm you now have a swapfile

3. To make the swap file persist if the server is rebooted:

   ```
   sudo chmod 0600 /var/swap.img
   sudo chown root:root /var/swap.img
   sudo nano /etc/fstab
   ```

   Append the following line to the end of the file:

   ```
   /var/swap.img none swap sw 0 0
   ```

   Save the file and exit

3. Install the dependencies needed before compiling the Masternode

   ```
   sudo apt-get update
   sudo apt-get install automake
   sudo apt-get install libdb++-dev
   sudo apt-get install build-essential libtool autotools-dev
   sudo apt-get install autoconf pkg-config libssl-dev
   sudo apt-get install libboost-all-dev
   ```

```
sudo apt-get install libminiupnpc-dev
sudo apt-get install git
sudo apt-get install software-properties-common
sudo apt-get install python-software-properties
sudo apt-get install g++
```

4. Download and compile the Berkely DB v4.8 database

```
sudo add-apt-repository ppa:bitcoin/bitcoin
sudo apt-get update
sudo apt-get install libdb4.8-dev libdb4.8++-dev -y
```

5. Download the chaincoin source code:

```
cd ~
git clone https://github.com/chaincoin/chaincoin.git
```

6. Compile the masternode using the Berkely DB v4.8 and no GUI

```
cd ~/chaincoin/
./autogen.sh
./configure --without-gui
make
sudo make install
```

7. Edit the configuration file for the Masternode
    a. Go to the configuration folder:

    ```
    cd ~/.chaincoin/
    ```

    b. If the folder doesn't exist, create it with

    ```
    mkdir ~/.chaincoin/
    ```

    c. List the contents and look for **chaincoin.conf**

    ```
    ls
    ```

    d. If the file doesn't exist, create it like this:

    ```
    touch chaincoin.conf
    ```

    e. Edit the file:

    ```
    nano chaincoin.conf
    ```

    f. Add these lines to the file if they don't already exist:

    ```
    rpcuser=username
    rpcpassword=password
    server=1
    ```

The rpcuser and rpcpassword values are for the RPC interface, allowing you to interact with the Masternode from the command line. Use any values you like but keep a copy of them on file somewhere.

8. Run chaincoind and wait for it to sync. This may take a while as it needs to download a large file.

   ```
   chaincoind --daemon
   ```

9. Get an address to deposit CHC into, as this is the cost of starting the Masternode

   ```
   chaincoind getaccountaddress 0
   ```

10. Send exactly 1000 CHC to this address from Cryptopia.

11. Get a masternode key and make a note of it

    ```
    chaincoind masternode genkey
    ```

12. Stop chaincoind

    ```
    chaincoind stop
    ```

13. Update your chaincoin.conf file

    ```
    listen=1
    masternode=1
    masternodeprivkey=masternodekey
    masternodeaddr=<your server ip>:11994
    ```

    Note: You must have an IP that's connectable from the internet with port 11994 open.

14. Start chaincoind

    ```
    chaincoind --daemon
    ```

15. Wait for chaincoind to sync, and your 1000 CHC deposit to have 15 confirmations. You can see how many confirmations you have with the command

    ```
    chaincoind listtransactions
    ```

16. Once you have 15 confirmations, start your masternode

    ```
    chaincoind masternode start
    ```

4. Managing your masternode

1. Stopping your masternode

   ```
   chaincoind masternode stop
   ```

   This will stop the Masternode. The Chaincoin daemon will still run.

2. Monitoring your masternode

   ```
   chaincoind getinfo
   ```

   or

   ```
   chaincoind masternode current.
   ```

   You can also do this remotely using the rpc interface.

3. Closing the masternode and recovering your 1000CHC:

   Stop the masternode, and then check your balance

   ```
   chaincoind getinfo
   ```

   You might have to wait if your coins are locked up. Then:

   ```
   chaincoind sendtoaddress <newaddr> <amt>
   ```

   Once you receive your coins

   ```
   chaincoind masternode stop
   ```

5. What can go wrong

- Your SSH key from PuTTY might be in the wrong format (it must start with “ssh-rsa” NOT "---- BEGIN SSH2 PUBLIC KEY ..."
- Using a newer version of Ubuntu can break compilation with errors related to Boost
- Not installing the old Berkely DB v4.8 will break the compilation
- Not enough RAM can cause the compilation to fail, hence the creation of a swap drive
- If you have problems, please contact us on Twitter/Slack…