



Decentralized Evolutive Ecosystem

Cryptocurrency developed for the crypto community.

Featuring: DAO, fast payments, privacy, masternodes and much more!

[Visit us @ https://www.pushiplay.pw/](https://www.pushiplay.pw/)

Pushi Masternode Setup Guide

(HOT+COLD Wallet setup with Vultr.com Linux VPS)
ETN_HERO#1519 & Tekcruzer#9282 - Pushi Discord Server
Click [HERE](#) to Join Server



What is a masternode?

Masternodes are computers on the Pushi network that provide network services and facilitate PrivateSend functionality. They also partake in the governance functions of the network and are allowed to submit and vote on proposals for the network. A masternode should have a fixed ip address and a stable uptime.

EXPLAINING THE IDEA OF A MASTERNODE

It is important to understand how the Pushi ecosystem works before one can see the value of a masternode. Pushi, a popular privacy-centric altcoin, uses a proof-of-work system similar to the one found in bitcoin. However, that is only part of the ecosystem that allows users to earn money. Not everyone is able to contribute to the network as a miner, which is why masternodes were introduced.

Look at a masternode as a special server that is maintained at all times. Masternodes are trustless and decentralized, similar to how bitcoin nodes operate. There is a major difference, though, as Pushi masternodes take care of the anonymization part of the Private Send **protocol**. Users can opt to send transactions anonymously by using this feature directly from their Pushi wallet.

Every masternode on the network provides this anonymization service, ensuring there is no centralized party to attack or take down. Moreover, masternodes ensure all transactions are validated in near real-time, making them quite efficient. Unlike bitcoin nodes, however, owners of a Pushi masternode will receive a financial compensation for providing these invaluable services.

Individual masternodes on the network have a chance to be selected as a recipient of part of each mined block's value. To achieve this ROI, masternode owners must place 1,000 Pushi into the wallet associated with this node. Moving the funds out of the wallet will remove the masternode from the network. Additionally, the wallet address will no longer be eligible for rewards either. It is possible for users to move their funds out of the wallet at any given time, although it is not in anyone's best interest to do so overnight. With the rewards flowing in virtually every week, there is a lot of passive income to be generated by running a masternode. All things considered, using a masternode is an intriguing system.

Pushi masternodes need to run with what's known as a hot/cold wallet setup. The basic premise is that the "cold" wallet is your local computer and the "hot" wallet is a remote VPS (virtual private server). The cold wallet effectively holds your 1,000 Pushi collateral and the rewards that are generated are sent to that wallet. This wallet is (or should be) password protected and can be closed once the masternode is started. The "hot" wallet is an empty wallet on the VPS. You don't send funds to it and don't even need to know its wallet address except during setup. It remains unlocked, but also empty....so if your VPS is ever compromised, there's no risk of losing funds.

Setting up a Masternode requires basic skills in Linux OS commands and file operations. If you have never worked with Linux take one of the many tutorials on the internet.

Example: <http://www.vogella.com/tutorials/Ubuntu/article.html>

With that said, let's get onto the process of setting up a Pushi masternode...

NOTE - If you see any error messages while working through this guide or certain commands aren't working, **please retrace your steps** or stop in at our [Discord](#) channel for support.



Also, there are other setup guides out there and they take different approaches and place files in different directories....**jumping between guides is not recommended** unless you're an experienced Linux user and can easily and account for the differences in paths and commands.

SCREENSHOTS SHOW v1.1.6 WALLET.

BE SURE TO UPDATE YOUR LOCAL WALLET TO v1.1.6 PRIOR TO UPDATING THE MASTERNODE.

As of April 2018 Not using the V1.1.6 wallet is not optional it is required due to a soft-fork. You will not get MN rewards otherwise. Pushi Masternodes Require that Sentinel is enabled and running correctly.

What you will need:

- **More than 1000 PUSHI (You will need a few extra for fees)**
- **One computer with pushi-qt wallet installed. Make sure the wallet contains the Masternode Collateral of at least 1000 PUSHI**
- **One VPS.**
- **A decent amount of technical knowledge. i.e. Knowing what a VPS is and using basic Linux shell commands.**
- **Patience – enough to follow these instructions properly before asking questions!**

Should you need or want it there is a Paper Wallet - [HERE](#) This is **NOT required** for installation purposes. Paper Wallets are used to store Coin where only you have control of keys.

Setup Guide - For V1.1.6 and up.

The current version of Wallet and VPS is @1.1.6 A soft-fork occurred and any prior installations should be upgraded to current version. This install guide can be used for updating as well as clean installs. Masternode rewards are now only operational on this version running Sentinel. All prior versions will be orphaned on the prior blockchain.

Pre - Install - Wallet Download & Installation

Wallet V1.1.6 Download [HERE](#)

Please make sure you choose the right package for the platform you are installing your wallet on (Win 32, 64, Mac etc). If you are making a clean install the latest wallet version is included in the distribution package from Github. Installing the desktop wallet is simply a case of putting the downloaded Pushi-qt.exe in a folder of your choice and running with administrator permissions. It will ask where you want the .pushicore files setup. Choose the preferred location unless you have reason not to and you know what you are doing. Pushi Discord Support will assume the preferred location.

Step 1 - Setting up the collateral transaction

The first step involves sending exactly 1,000 Pushi to a new wallet address. You'll want to have a small amount above 1,000 Pushi to cover the transaction fee, so you'll need to have a starting balance of at least, say 1,001.00

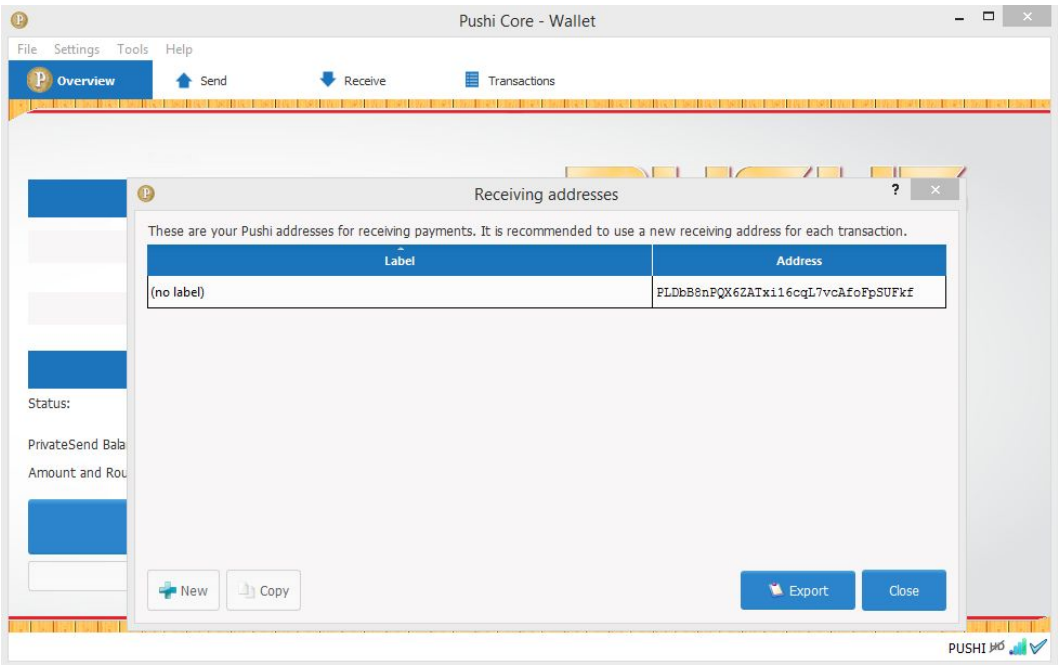


Pushi. First, we'll create a new wallet address to hold the 1,000 collateral. This will also be the address that the masternode rewards are sent to.

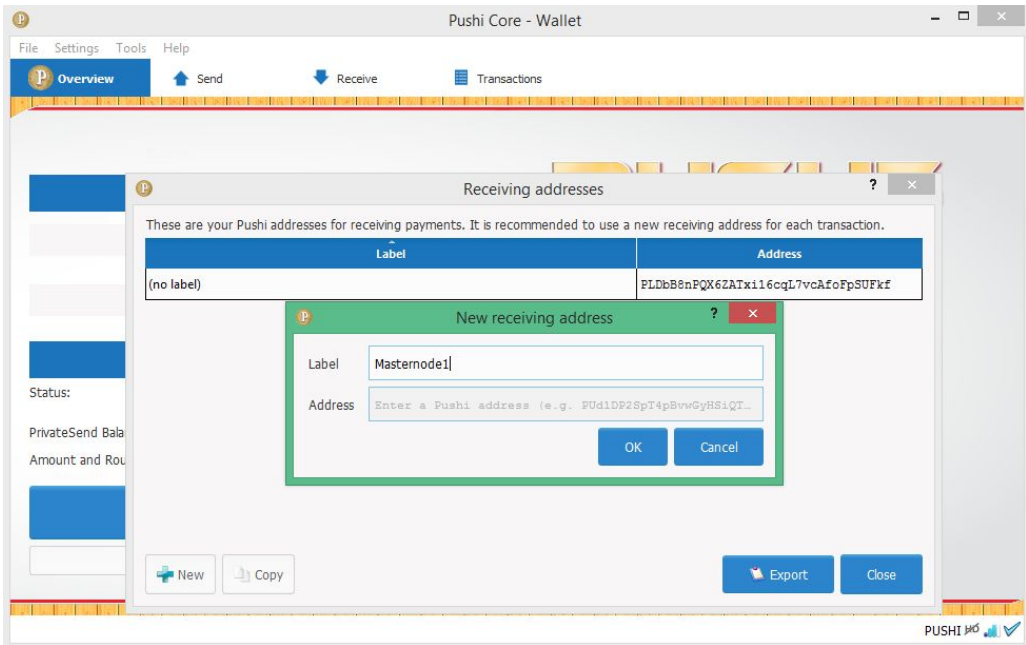
In the QT wallet, choose:

File->Receiving addresses...

This brings up a list of the receiving addresses managed by your wallet.



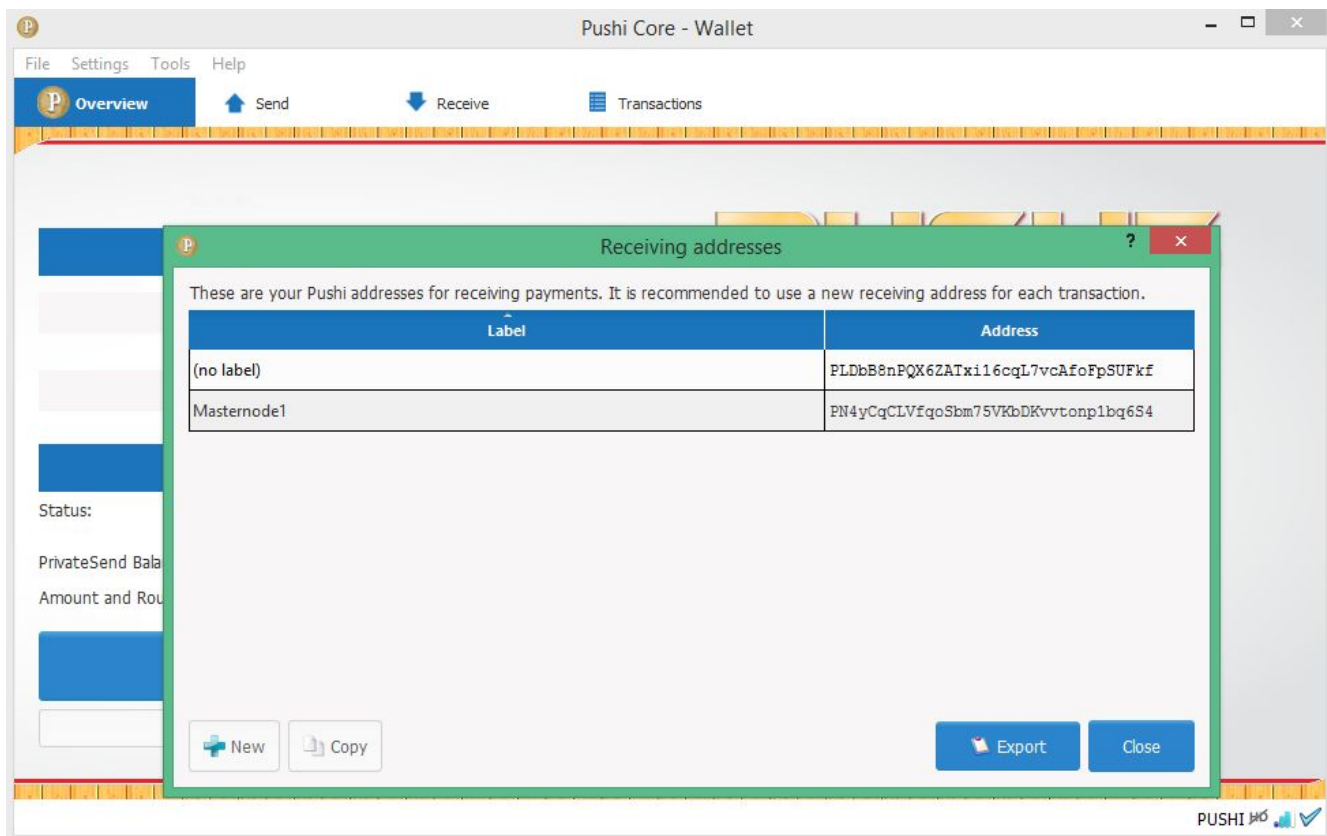
Click **+New**



...and give it a label, such as **Masternode1** or **MN1** - We will use Masternode1 for clarity.



Click **Ok** to create the address. You'll see the new address in the list, select it and choose **Copy** to store that address in the clipboard.



Click **Close** to exit the receiving address dialog. Now we'll send the 1,000 collateral amount to the address you just created.

Under the **Send** tab, paste the address that you copied into the **Pay To:** field. The Label field should pre-populate with the label you gave it earlier, in this case, Masternode1.



Pushi Core - Wallet

File Settings Tools Help

Overview Send Receive Transactions

Pay To: PN4yCqCLVfqoSbm75VKbDKvvtontp1bq6S4

Label: Masternode1

Amount: 1000.00000000 PUSHI ☐ Subtract fee from amount

Transaction Fee: 0.00001000 PUSHI/kB Choose...

Send Clear All Add Recipient

☐ PrivateSend ☐ InstantSend Balance: 1004.99994170 PUSHI

PUSHI

In the Amount: field, enter **1,000**. Do **NOT** check "Subtract fee from amount" as this will subtract the transaction fee from the Amount and your transaction will be below the required 1,000 Pushi.

Pushi Core - Wallet

File Settings Tools Help

Overview Send Receive Transactions

Pay To: PN4yCqCLVfqoSbm75VKbDKvvtontp1bq6S4

Label: Masternode1

Amount: 1000.0000

Transaction Fee: 0.00001000 PUSHI/kB Choose...

Send Clear All Add Recipient

☐ PrivateSend ☐ InstantSend Balance: 1004.99994170 PUSHI

PUSHI

Confirm send coins

Are you sure you want to send?

1000.00000000 PUSHI using any available funds (not anonymous) to Masternode1 (PN4yCqCLVfqoSbm75VKbDKvvtontp1bq6S4)

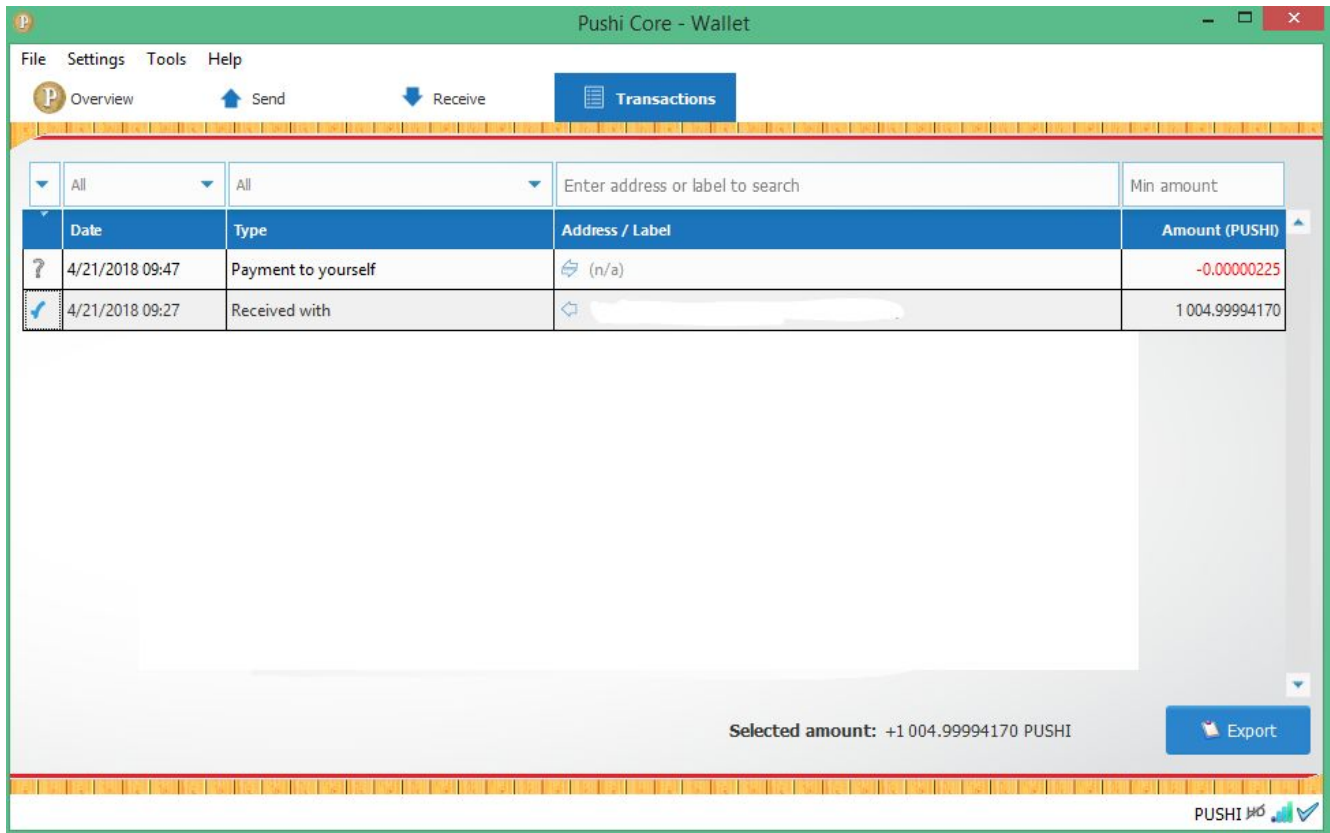
0.00000225 PUSHI are added as transaction fee (0.225 kB)

Total Amount = 1000.00000225 PUSHI
 = 1 000 000.00225 mPUSHI
 = 1 000 000 002.25 μPUSHI
 = 100 000 000 225 duffs

(1 of 1 entries displayed)

Yes Cancel





Click **Send** and your transaction will be broadcast to the network. You'll need to wait for 15 confirmations (the current number of confirmations is viewable in the Transactions tab) before the masternode will fully activate about 40 mins, but we can start working through the rest of the setup in the meantime.

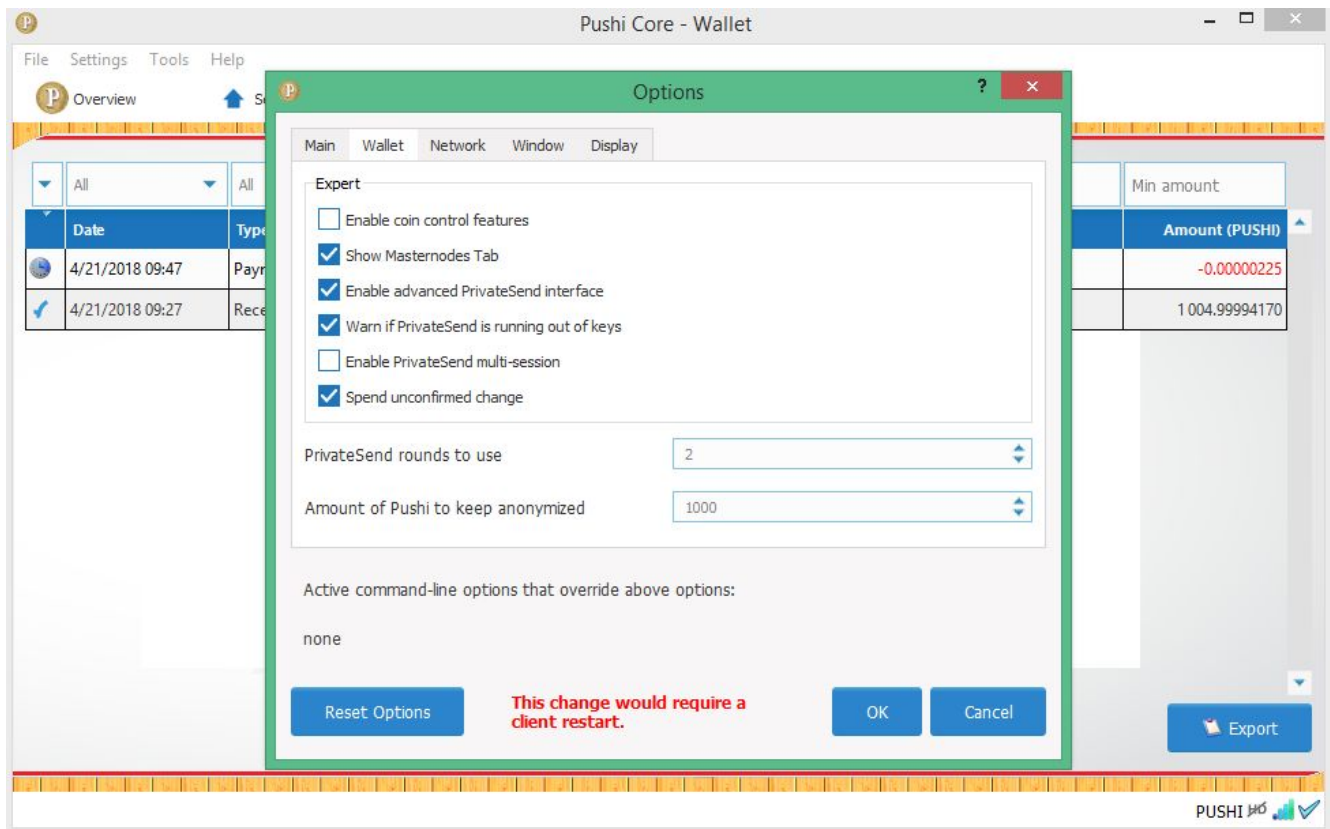
Step 2 - Generate the masternode private key and txid

The next step is to enable the Masternodes tab in your local wallet and use the debug console to output a few important details that we'll need.

To enable the Masternodes tab in your local wallet, go to:

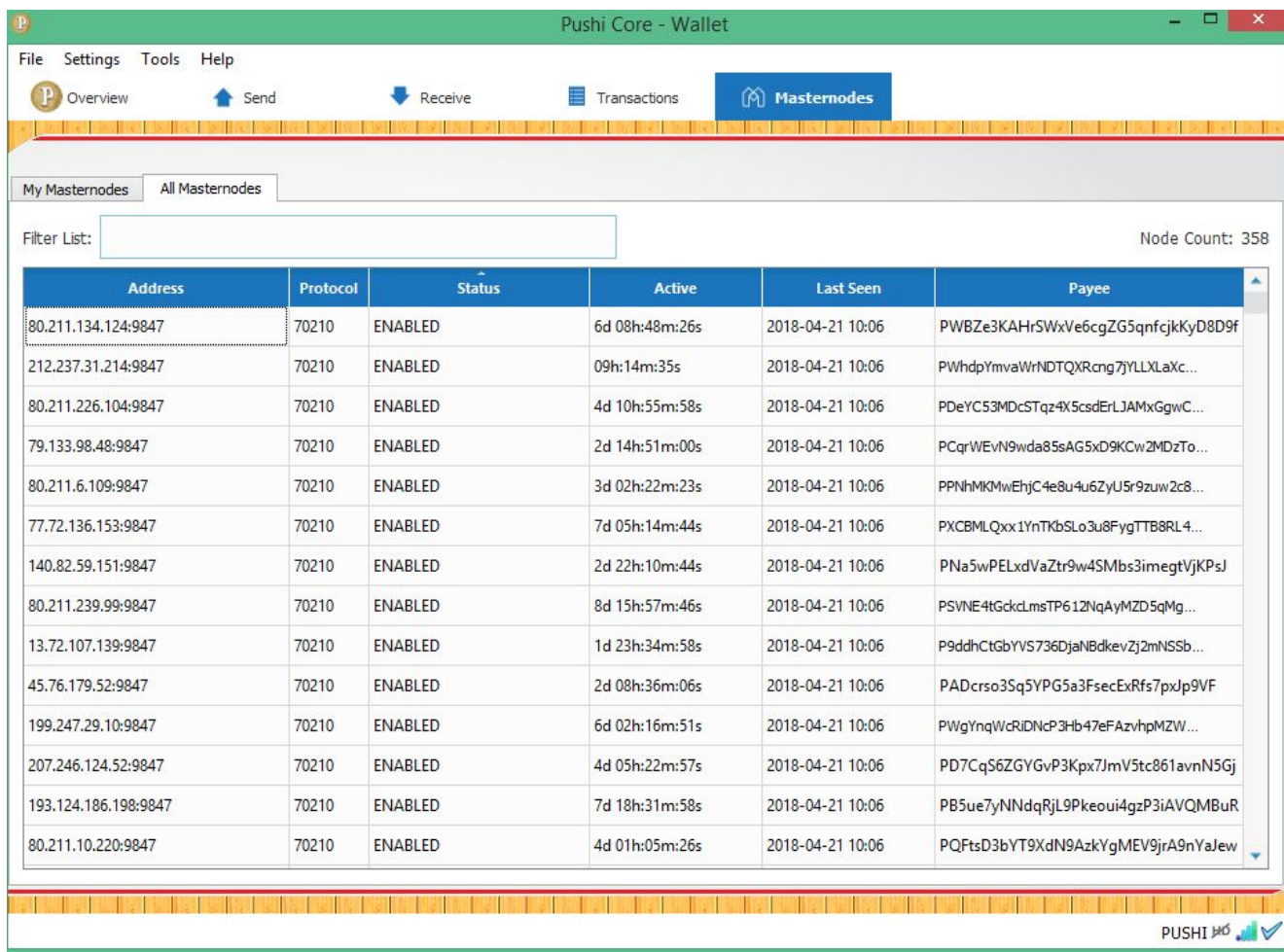
Settings->Options->Wallet->Show Masternodes Tab





Click **Ok** to apply the setting and then close and re-open your local wallet. After re-opening, you should see the Masternodes tab which will display the full list of masternodes on the network, as well as masternodes associated with your local wallet (My Masternodes is probably empty because we haven't added one yet).





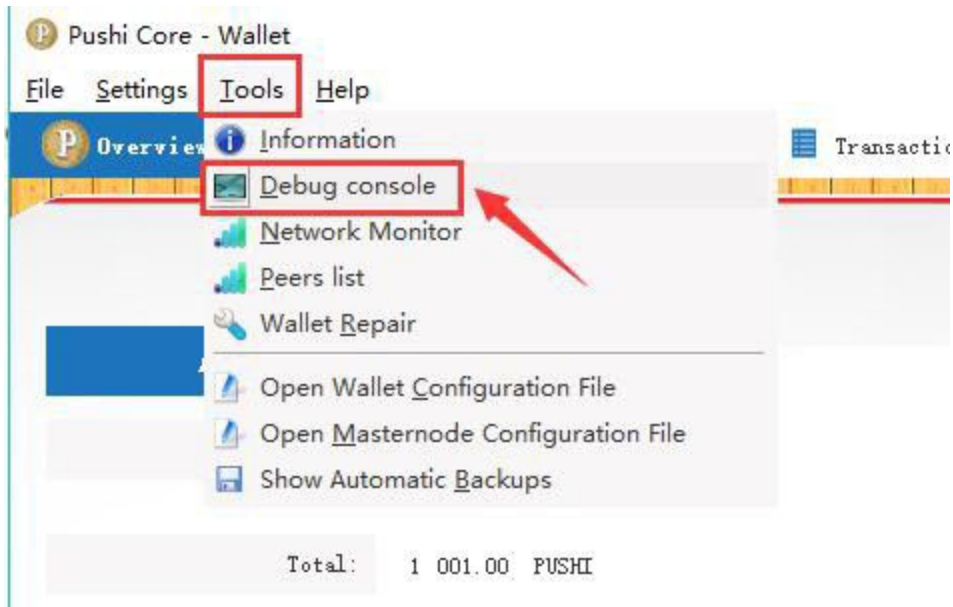
| Address | Protocol | Status | Active | Last Seen | Payee |
|----------------------|----------|---------|----------------|------------------|-------------------------------------|
| 80.211.134.124:9847 | 70210 | ENABLED | 6d 08h:48m:26s | 2018-04-21 10:06 | PWBZe3KAHrSWxVe6cgZG5qnfckjKyD8D9f |
| 212.237.31.214:9847 | 70210 | ENABLED | 09h:14m:35s | 2018-04-21 10:06 | PWhdpYmvaWrNDTQXRong7jYLLXLaXc... |
| 80.211.226.104:9847 | 70210 | ENABLED | 4d 10h:55m:58s | 2018-04-21 10:06 | PDeYC53MDcSTqz4X5csdErLJAMxGgwC... |
| 79.133.98.48:9847 | 70210 | ENABLED | 2d 14h:51m:00s | 2018-04-21 10:06 | PCqrWEvN9wda85sAG5xD9KCw2MDzTo... |
| 80.211.6.109:9847 | 70210 | ENABLED | 3d 02h:22m:23s | 2018-04-21 10:06 | PPNhMKMwEhjC4e8u4u6ZyU5r9zuw2c8... |
| 77.72.136.153:9847 | 70210 | ENABLED | 7d 05h:14m:44s | 2018-04-21 10:06 | PXCBMLQxx1YnTKbSL03u8FygTTB8RL4... |
| 140.82.59.151:9847 | 70210 | ENABLED | 2d 22h:10m:44s | 2018-04-21 10:06 | PNa5wPELxdVaZtr9w4SMbs3imegtVjKPsJ |
| 80.211.239.99:9847 | 70210 | ENABLED | 8d 15h:57m:46s | 2018-04-21 10:06 | PSVNE4tGckLmsTP612NqAyMZD5qMg... |
| 13.72.107.139:9847 | 70210 | ENABLED | 1d 23h:34m:58s | 2018-04-21 10:06 | P9ddhCtGbYVS736DjaNBdkevZj2mNSSb... |
| 45.76.179.52:9847 | 70210 | ENABLED | 2d 08h:36m:06s | 2018-04-21 10:06 | PADcrso3Sq5YPG5a3FsecExRfs7pxJp9VF |
| 199.247.29.10:9847 | 70210 | ENABLED | 6d 02h:16m:51s | 2018-04-21 10:06 | PWgYnqWcRIDNcP3Hb47eFAzvhpMZW... |
| 207.246.124.52:9847 | 70210 | ENABLED | 4d 05h:22m:57s | 2018-04-21 10:06 | PD7CqS6ZGYGvP3Kpx7JmV5tc861avnN5Gj |
| 193.124.186.198:9847 | 70210 | ENABLED | 7d 18h:31m:58s | 2018-04-21 10:06 | PB5ue7yNNdqRjL9Pkeoui4gzP3iAVQMBuR |
| 80.211.10.220:9847 | 70210 | ENABLED | 4d 01h:05m:26s | 2018-04-21 10:06 | PQFtsD3bYT9XdN9AzkYgMEV9jrA9nYaJew |

Next, we need to access the **debug console** to output what's known as the **collateral txid** and the **masternode private key**. The collateral txid is an identifier for the transaction of 1,000 Pushi that you made previously. The masternode private key is a key that is specific to your wallet and is used to validate your masternode on the network. Like any private key, you want to keep this secret...**there's no reason to share this number with anyone or post it publicly.**

To access the debug console, go to:

Tools->Debug console





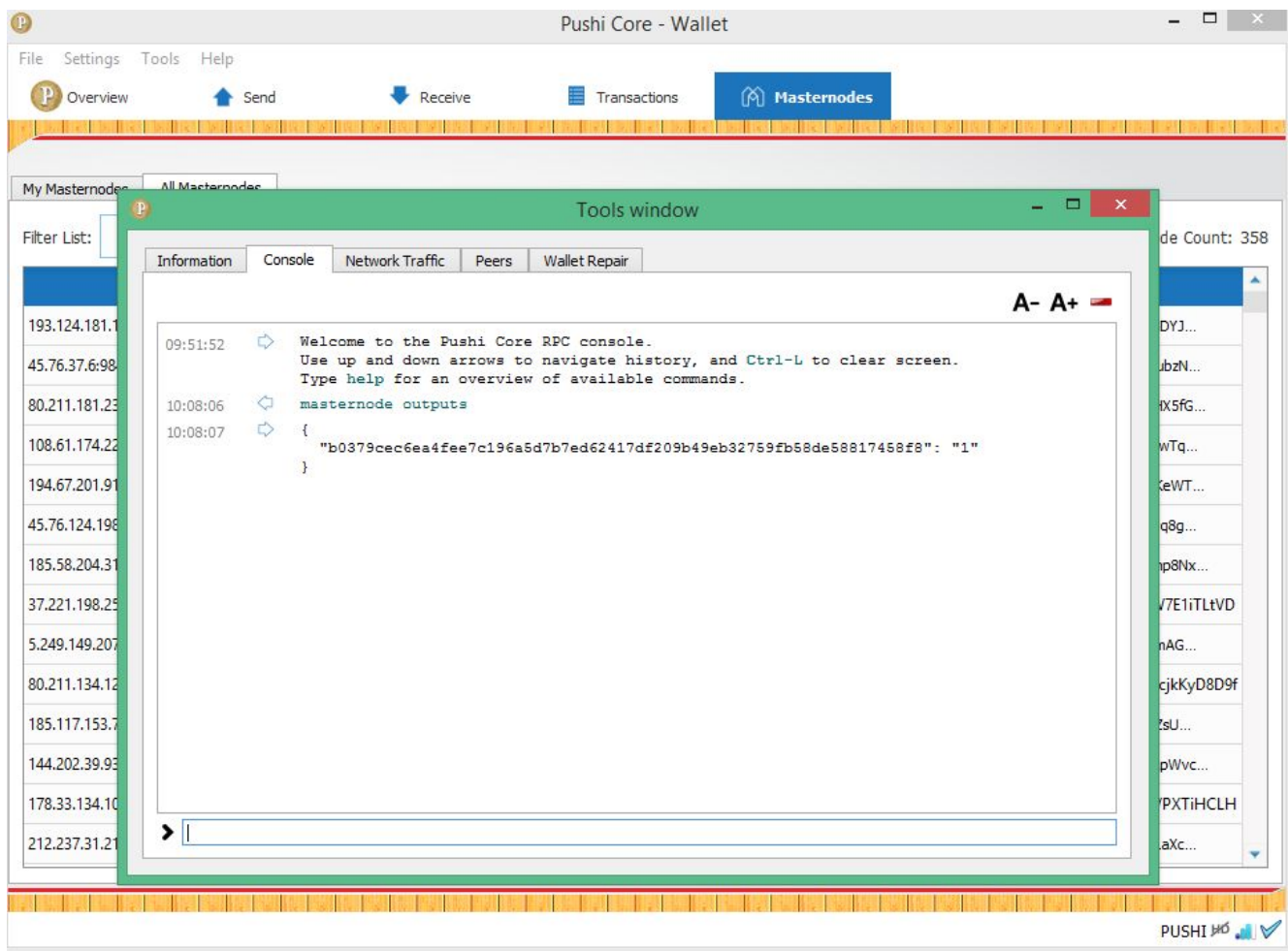
The debug console has a wide range of commands that provide all sorts of helpful information and provide convenient functions, you can see these by typing **help** in the console. In this case, we'll use it to obtain your masternode txid and private key.

In the debug console, type in:

masternode outputs

This will list all transactions that meet the parameters for starting a masternode. Since we sent a transaction to a new local wallet address for 1,000 Pushi, we should see the transaction id listed here (if you're not seeing any output, verify that you have in fact sent a transaction to a local wallet address for exactly 1,000 Pushi):



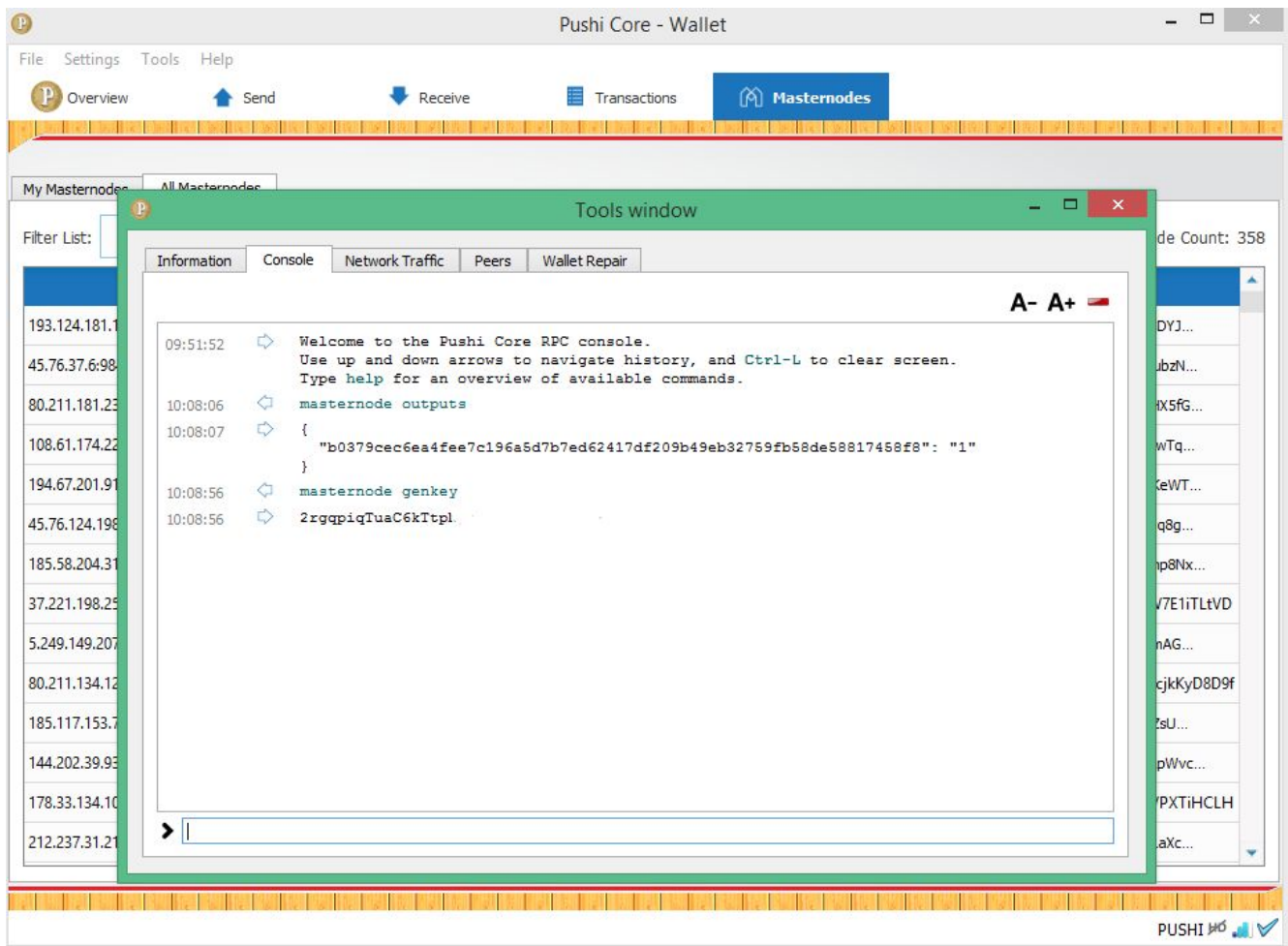


Copy that output and paste it somewhere convenient (in an empty Notepad session is fine).

Next, we'll generate a private key for you masternode. Type in:

masternode genkey





Genkey intentionally cut for privacy reasons. Actual genkey will be longer. REMINDER: Never share this genkey to anyone no matter the circumstances.

Copy and paste this number somewhere convenient as well, we'll use it shortly. With these two numbers output and pasted in a text document, we can proceed with the VPS setup.

Important safety procedure time!

Begin securing your wallet by clicking on SETTINGS, ENCRYPT WALLET.

Enter a password, repeat it, and click OK.

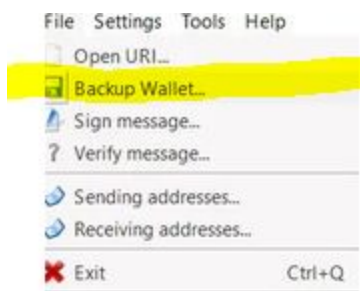
Do not forget your password! Keep it somewhere safe, as you do not want to lose your Pushi!

IMPORTANT: I recommend that you type your password into your notepad file, and then cut/paste into the password field so you are sure of what your password will be. This is a big investment, and it pays to be sure!

You will see your Pushi wallet may become unresponsive for several seconds, this is normal. Please do not hit any keys until your wallet has finished encrypting. Once your wallet has finished and closed, you are ready for the next security step:



- Always do a backup of your WALLET.DAT if you have Pushi on your computer.



Select the location where you want to store a copy of your WALLET.DAT, and backup your wallet.

Please note: I recommend backing up your wallet to several USBs to keep in different locations.

- Think twice before deleting or overwriting anything, as there is no way back.

- Never copy WALLET.DAT when your wallet is running! Please use the BACKUP WALLET command from within your wallet instead.

Trouble Shooting - Wallet

1. Collateral is 1000 Pushi no more no less. Remember to have a few extra for fees.
2. Must use the V1.1.6 Wallet - Otherwise you will be on wrong Blockchain
3. You can check you are on right blockchain by typing
getblockhash 38000 at the debug console and on the VPS ./pushi-cli getblockhash 38000. You should see:

```
./pushi-cli getblockhash 38000
```

```
00000000007fbdab3752bef758e02ff044807e2e5746559639da001438a4a521
```

This the current blockchain. If you see a different number then you need to resync

4. Make sure you have created the masternode.conf, pushi.conf in your pushicore folder usually found in %APPDATA% in you windows <USER> folder.

Step 3 - Setting up a Linux VPS with Vultr.com

This part of the guide will take you through setting up a VPS with vultr.com. There are a few other good VPS services out there, but Vultr has basic servers that can be run for \$2.50/month and provide the resources necessary to run a masternode.

<https://vultr.com>

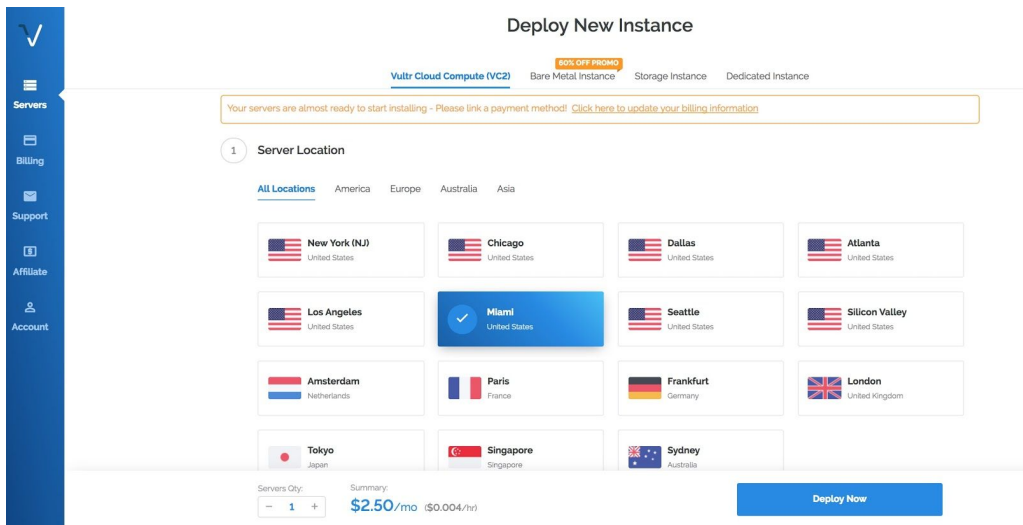
Create an account with Vultr.com and setup a payment method. Once you've created an account with Vultr and linked a credit card or payment method, navigate to:

<https://my.vultr.com/>

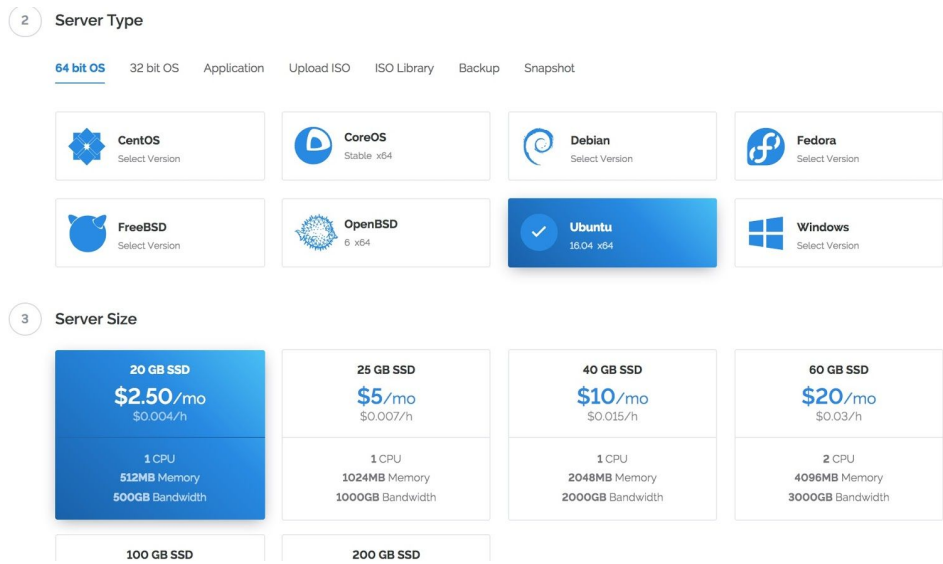


This is your main console for Vultr and where you can deploy and monitor servers. Click on the **Servers** tab on the left. If you have servers running already, you'll see them listed here and can click + to launch a new server. Otherwise, you should already be seeing the server deployment page.

Choose a location:



...and then choose a type. For Pushi, we want a **64-bit Ubuntu 16.04 x64** installation. Next, choose a Server Size. The standard \$2.50/month server should be adequate.



If it's listed as Temporarily Sold Out:



3

Server Size

Temporarily Sold Out

20 GB SSD

\$2.50/mo

\$0.004/h

1 CPU

512MB Memory

500GB Bandwidth

...then scroll back up and choose a different location. It seems that the United States based locations have the best availability in the \$2.50/month server size.

4

Additional Features

- ☐ Enable IPv6
- ☐ Enable Auto Backups **\$0.50/mo**
- ☐ Enable DDOS Protection ? **\$10/mo**
- ☐ Enable Private Networking ?

5

Startup Script ([Manage](#))

+

Add New

6

SSH Keys ([Manage](#))

+

Add New

7

Server Hostname & Label

Enter server hostname

masternode1

Enter server label

masternode1

Servers Qty:

-

1

+

Summary:

\$2.50/mo (\$0.004/hr)

Deploy Now



Servers

Sort by: Location

Instances Snapshots ISO Startup Scripts SSH Keys DNS Backups Block Storage Reserved IPs Firewall



Server added successfully!

Read How-To Articles and FAQs on [Vultr Docs](#)

Good News - Your account can earn some additional free credit! [Click here to view available promos](#)

| <input type="checkbox"/> | Server | OS | Location | Charges | Status |
|--------------------------|----------------------------------------------------|----|------------|---------|------------|
| <input type="checkbox"/> | masternode1 512 MB Server - 207.148.28.6 | | New Jersey | --- | Installing |

Restart

Stop

You'll be taken back to the list of your servers and you'll see that the server you just deployed is installing. Once it's listed as Running, we can login for the first time.

| <input type="checkbox"/> | Server | OS | Location | Charges | Status |
|--------------------------|----------------------------------------------------|----|------------|---------|--------------------------------|
| <input type="checkbox"/> | masternode1 512 MB Server - 207.148.28.6 | | New Jersey | --- | Running Manage |

Restart

Stop

We need to retrieve the login credentials for the initial login. Click on the name of the server, in this case 'masternode1'.

This will bring you to the Server Information page for your VPS:

Location: New Jersey

IP Address: 144.202.0.100

Username: root

Password: 2Ru-

Down in the bottom left you'll see the Username: root and Password. Click on the icon of the eye to reveal the password for the root account. This is the set of credentials that we'll use to login.

Step 4 - Logging into your VPS with PuTTY



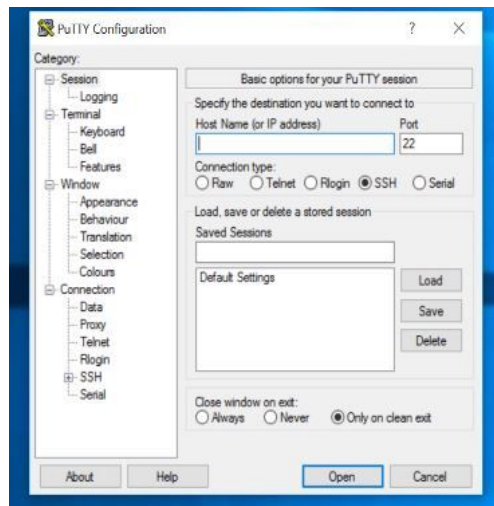
To access the VPS, we need to use a protocol called SSH. An SSH session will give you command-line access to your VPS and will be the mechanism we use for running commands on the Linux machine. We'll use a program called PuTTY. This is a popular and widely used ssh client in Windows. (For OSX users, you can use your Terminal and just run the command `ssh root@<YOUR VPS IP ADDRESS>`)

The latest version of PuTTY can be downloaded from here:
<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

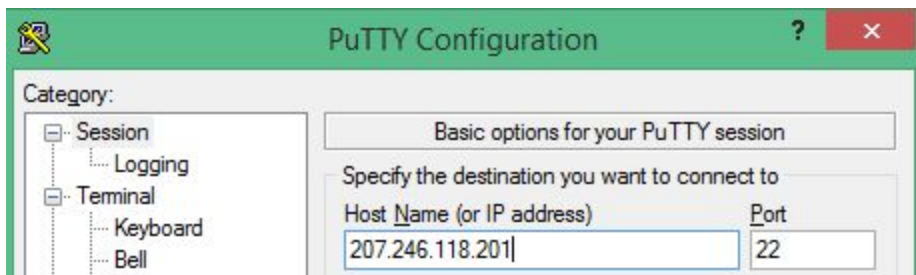
...or the most recent version at the time of this writing, 0.70, can be downloaded directly from here:

<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.70-installer.msi>

Install PuTTY and run it from the Start Menu. Once open, you'll see the standard PuTTY interface:

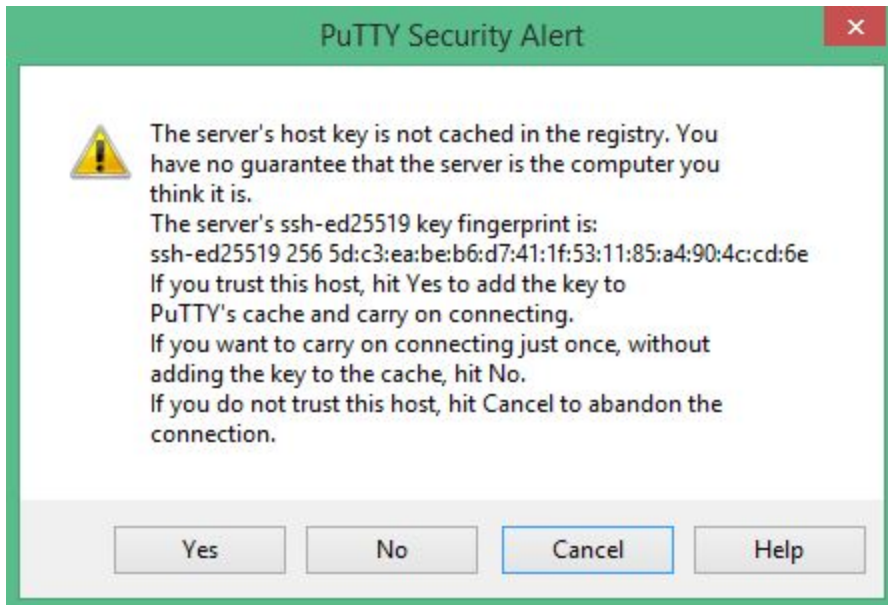


Under Host Name, enter the IP address for your VPS. In this case we use 207.246.118.201:



For convenience, you can also type a name (like "vultr masternode1") under Saved Sessions and click save to store your VPN details. Click Open and PuTTY will initiate the first connection to your VPS. You'll likely see a security alert listing the ssh key fingerprint, choose Yes.





Next up, you'll see the login prompt for your VPS. The username is **root** and the password is the password that was listed on your Server Information page in Vultr:

```
login as: root
root@207.246.118.201's password:
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-109-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

7 packages can be updated.
7 updates are security updates.

*** System restart required ***
Last login: Sat Apr 21 08:01:55 2018 from 101.127.96.119
root@masternode1:~#
```

Note that in PuTTY, a right-click will paste the contents of your clipboard....so if you're copying and pasting the root password, a right-click will paste. You won't see any *'s or feedback for the password when you paste it, so just hit Enter after pasting. If you get an Access Denied message, double-check your root password and try typing it in manually rather than pasting.

Step 5 - Setting up a new user account (Optional)

Ref: <https://www.howtogeek.com/124950/htg-explains-why-you-shouldnt-log-into-your-linux-system-as-root/>

Now that we're logged in as root, you may want to create a separate user to run the masternode under. The root account has the highest level of system access and it's generally not good to run as root if you can avoid it, so



let's make a user named **'pushinode'**.

- **IMPORTANT NOTE:** For the rest of this guide the user *pushinode* and *root* are interchangeable dependent on your choice.

Continue with

adduser pushinode

Assign a password that you'll remember, and then accept the remainder of the defaults:

```
root@masternode1:~# adduser pushinode
Adding user 'pushinode' ...
Adding new group 'pushinode' (1000) ...
Adding new user 'pushinode' (1000) with group 'pushinode' ...
Creating home directory '/home/pushinode' ...
Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for pushinode
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@masternode1:~#
```

Next, we'll give this user what's known as sudo access. This allows that user to run commands with elevated privileges and to do things like install software and system updates.

Type the following:

usermod -aG sudo pushinode

```
root@masternode1:~# usermod -aG sudo pushinode
root@masternode1:~#
```

You won't see any output, but you've just given the user 'pushinode' sudo access. Now we can logoff our root account from the VPS and re-connect as our new 'pushinode' user. Type:

exit

...to close the current ssh connection. This will drop you back to the main PuTTY interface. Re-connect to the VPS the same way we did before, by typing the IP address into the Host Name field and click Open.

You'll be presented with the login prompt again, this time we'll log in as our 'pushinode' user.

login as: **pushinode**

password: **<the password that you set earlier>**



```
login as: pushinode
pushinode@144.202.0.100's password:
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-109-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

116 packages can be updated.
53 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

pushinode@masternode1:~$
```

Step 6 - Installing dependencies, creating swap space, and installing Pushi binaries

At this point, you should be logged into the vps and at a command prompt. The first thing we'll do is install updates to Ubuntu. The installation and management of software in Ubuntu is done using the 'apt-get' command. First run:

Step 6.1 Installing Dependencies

1. sudo apt-get update

```
pushinode@masternode1:~$ sudo apt-get update
[sudo] password for pushinode:
Hit:1 http://security.ubuntu.com/ubuntu xenial-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu xenial InRelease
Hit:3 http://archive.ubuntu.com/ubuntu xenial-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu xenial-backports InRelease
Reading package lists... Done
pushinode@masternode1:~$
```

2. sudo apt-get upgrade




```
After this operation, 12.9 MB of additional disk space will be used.  
Do you want to continue? [Y/n]
```

Whenever you see this question, just type Y and Enter

```
Setting up open-vm-tools (2:10.0.7-3227872-5ubuntu1~16.04.2) ...  
Installing new version of config file /etc/vmware-tools/tools.conf ...  
Setting up overlayroot (0.27ubuntu1.5) ...  
Installing new version of config file /etc/overlayroot.conf ...  
Setting up python3-distupgrade (1:16.04.25) ...  
Setting up python3-update-manager (1:16.04.12) ...  
Setting up ubuntu-release-upgrader-core (1:16.04.25) ...  
Setting up update-manager-core (1:16.04.12) ...  
Setting up update-notifier-common (3.168.8) ...  
Processing triggers for libc-bin (2.23-0ubuntu10) ...  
Processing triggers for initramfs-tools (0.122ubuntu8.11) ...  
update-initramfs: Generating /boot/initrd.img-4.4.0-109-generic  
W: mdadm: /etc/mdadm/mdadm.conf defines no arrays.  
Processing triggers for resolvconf (1.78ubuntu6) ...  
pushinode@masternode1:~$
```

3. sudo apt-get install git

```
pushinode@masternode1:~$ sudo apt-get install git  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
git is already the newest version (1:2.7.4-0ubuntu1.3).  
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.  
pushinode@masternode1:~$
```

4. sudo apt-get install automake

```
Setting up m4 (1.4.17-5) ...  
Setting up autoconf (2.69-9) ...  
Setting up autotools-dev (20150820.1) ...  
Setting up automake (1:1.15-4ubuntu1) ...  
update-alternatives: using /usr/bin/automake-1.15 to provide /usr/bin/automake  
automake) in auto mode  
pushinode@masternode1:~$
```

5. sudo apt-get install build-essential

```
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mo  
de  
Setting up make (4.1-6) ...  
Setting up libdpkg-perl (1.18.4ubuntu1.4) ...  
Setting up dpkg-dev (1.18.4ubuntu1.4) ...  
Setting up build-essential (12.1ubuntu2) ...  
Processing triggers for libc-bin (2.23-0ubuntu10) ...  
pushinode@masternode1:~$
```



6. sudo apt-get install libtool

```
pushinode@masternode1:~$ sudo apt-get install libtool
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  libtool-doc gfortran | fortran95-compiler gcj-jdk
Recommended packages:
  libltdl-dev
The following NEW packages will be installed:
  libtool
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 193 kB of archives.
After this operation, 915 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu xenial/main amd64 libtool all 2.4.6-0.1 [
193 kB]
Fetched 193 kB in 0s (319 kB/s)
Selecting previously unselected package libtool.
(Reading database ... 95662 files and directories currently installed.)
Preparing to unpack .../libtool_2.4.6-0.1_all.deb ...
Unpacking libtool (2.4.6-0.1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up libtool (2.4.6-0.1) ...
pushinode@masternode1:~$
```

7. sudo apt-get install autotools-dev

```
pushinode@masternode1:~$ sudo apt-get install autotools-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
autotools-dev is already the newest version (20150820.1).
autotools-dev set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
pushinode@masternode1:~$
```

8. sudo apt-get install autoconf

```
pushinode@masternode1:~$ sudo apt-get install autoconf
Reading package lists... Done
Building dependency tree
Reading state information... Done
autoconf is already the newest version (2.69-9).
autoconf set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
pushinode@masternode1:~$
```



9. sudo apt-get install pkg-config

```
pushinode@masternode1:~$ sudo apt-get install pkg-config
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  pkg-config
0 upgraded, 1 newly installed, 0 to remove and 3 not upgraded.
Need to get 45.0 kB of archives.
After this operation, 177 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu xenial/main amd64 pkg-config amd64 0.29.1-0ubuntu1 [45.0 kB]
Fetched 45.0 kB in 0s (111 kB/s)
Selecting previously unselected package pkg-config.
(Reading database ... 95688 files and directories currently installed.)
Preparing to unpack .../pkg-config_0.29.1-0ubuntu1_amd64.deb ...
Unpacking pkg-config (0.29.1-0ubuntu1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up pkg-config (0.29.1-0ubuntu1) ...
pushinode@masternode1:~$
```



10. sudo apt-get install libssl-dev

```
pushinode@masternode1:~$ sudo apt-get install libssl-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  zlib1g-dev
Recommended packages:
  libssl-doc
The following NEW packages will be installed:
  libssl-dev zlib1g-dev
0 upgraded, 2 newly installed, 0 to remove and 3 not upgraded.
Need to get 1,511 kB of archives.
After this operation, 7,622 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 zlib1g-dev amd64 1:1.2.8.dfsg-2ubuntu4.1 [168 kB]
Get:2 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libssl-dev amd64 1.0.2g-1ubuntu4.12 [1,343 kB]
Fetched 1,511 kB in 1s (1,437 kB/s)
Selecting previously unselected package zlib1g-dev:amd64.
(Reading database ... 95704 files and directories currently installed.)
Preparing to unpack .../zlib1g-dev_1%3a1.2.8.dfsg-2ubuntu4.1_amd64.deb ...
Unpacking zlib1g-dev:amd64 (1:1.2.8.dfsg-2ubuntu4.1) ...
Selecting previously unselected package libssl-dev:amd64.
Preparing to unpack .../libssl-dev_1.0.2g-1ubuntu4.12_amd64.deb ...
Unpacking libssl-dev:amd64 (1.0.2g-1ubuntu4.12) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up zlib1g-dev:amd64 (1:1.2.8.dfsg-2ubuntu4.1) ...
Setting up libssl-dev:amd64 (1.0.2g-1ubuntu4.12) ...
pushinode@masternode1:~$
```



11. sudo apt-get install libboost-all-dev

```
Setting up mpi-default-dev (1.4) ...
Setting up libboost-mpi1.58-dev (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-mpi-dev (1.58.0.1ubuntu1) ...
Setting up libboost-python1.58.0 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up openmpi-bin (1.10.2-8ubuntu1) ...
update-alternatives: using /usr/bin/mpirun.openmpi to provide /usr/bin/mpirun (m
pirun) in auto mode
Setting up mpi-default-bin (1.4) ...
Setting up libboost-mpi-python1.58.0 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-mpi-python1.58-dev (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-mpi-python-dev (1.58.0.1ubuntu1) ...
Setting up libboost-program-options1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-program-options1.58-dev:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-program-options-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libpython2.7:amd64 (2.7.12-1ubuntu0~16.04.3) ...
Setting up libexpat1-dev:amd64 (2.1.0-7ubuntu0.16.04.3) ...
Setting up libpython2.7-dev:amd64 (2.7.12-1ubuntu0~16.04.3) ...
Setting up libpython-dev:amd64 (2.7.12-1~16.04) ...
Setting up python2.7-dev (2.7.12-1ubuntu0~16.04.3) ...
Setting up python-dev (2.7.12-1~16.04) ...
Setting up libboost-python1.58-dev (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-python-dev (1.58.0.1ubuntu1) ...
Setting up libboost-random1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-random1.58-dev:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-random-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-regex-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-serialization-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-signals1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-signals1.58-dev:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-signals-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-system-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-test-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-thread-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-timer1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-timer1.58-dev:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-timer-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-wave1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-wave1.58-dev:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
Setting up libboost-wave-dev:amd64 (1.58.0.1ubuntu1) ...
Setting up libboost-all-dev (1.58.0.1ubuntu1) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
pushinode@masternode1:~$
```

12. sudo apt-get install libevent-dev

```
Unpacking libevent-dev (2.0.21-stable-2ubuntu0.16.04.1) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Setting up libevent-core-2.0-5:amd64 (2.0.21-stable-2ubuntu0.16.04.1) ...
Setting up libevent-extra-2.0-5:amd64 (2.0.21-stable-2ubuntu0.16.04.1) ...
Setting up libevent-pthreads-2.0-5:amd64 (2.0.21-stable-2ubuntu0.16.04.1) ...
Setting up libevent-openssl-2.0-5:amd64 (2.0.21-stable-2ubuntu0.16.04.1) ...
Setting up libevent-dev (2.0.21-stable-2ubuntu0.16.04.1) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
pushinode@masternode1:~$
```



13. sudo apt-get install nano

```
pushinode@masternode1:~$ sudo apt-get install nano
Reading package lists... Done
Building dependency tree
Reading state information... Done
nano is already the newest version (2.5.3-2ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
pushinode@masternode1:~$
```

14. sudo apt-get install software-properties-common

```
pushinode@masternode1:~$ sudo apt-get install software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
software-properties-common is already the newest version (0.96.20.7).
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
pushinode@masternode1:~$
```

15. sudo apt-add-repository ppa:bitcoin/bitcoin

```
pushinode@masternode1:~$ sudo apt-add-repository ppa:bitcoin/bitcoin

Stable Channel of bitcoin-qt and bitcoind for Ubuntu, and their dependencies

Note that you should prefer to use the official binaries, where possible, to limit
it trust in Launchpad/the PPA owner.

No longer supports precise, due to its ancient gcc and Boost versions.
More info: https://launchpad.net/~bitcoin/+archive/ubuntu/bitcoin
Press [ENTER] to continue or ctrl-c to cancel adding it
gpg: keyring `/tmp/tmpa50nr5s6/secring.gpg' created
gpg: keyring `/tmp/tmpa50nr5s6/pubring.gpg' created
gpg: requesting key 8842CE5E from hkps server keyserver.ubuntu.com
gpg: /tmp/tmpa50nr5s6/trustdb.gpg: trustdb created
gpg: key 8842CE5E: public key "Launchpad PPA for Bitcoin" imported
gpg: no ultimately trusted keys found
gpg: Total number processed: 1
gpg:          imported: 1   (RSA: 1)
OK
pushinode@masternode1:~$
```

16. sudo apt-get update

```
pushinode@masternode1:~$ sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu xenial-security InRelease [102 kB]
Get:2 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial InRelease [17.5 kB]
Hit:3 http://archive.ubuntu.com/ubuntu xenial InRelease
Get:4 http://archive.ubuntu.com/ubuntu xenial-updates InRelease [102 kB]
0% [1 InRelease gpgv 102 kB] [4 InRelease 47.5 kB/102 kB 46%] [Connecting to pp
Get:5 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main amd64 Packages
[2,788 B]
Get:6 http://archive.ubuntu.com/ubuntu xenial-backports InRelease [102 kB]
Get:7 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main i386 Packages
[2,788 B]
Get:8 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main Translation-en
[1,712 B]
Fetched 331 kB in 1s (282 kB/s)
Reading package lists... Done
pushinode@masternode1:~$
```



17. sudo apt-get install libdb4.8-dev

```
pushinode@masternode1:~$ sudo apt-get install libdb4.8-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libdb4.8
Suggested packages:
  db4.8-doc
The following NEW packages will be installed:
  libdb4.8 libdb4.8-dev
0 upgraded, 2 newly installed, 0 to remove and 3 not upgraded.
Need to get 1,204 kB of archives.
After this operation, 4,269 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main amd64 libdb4.8
  amd64 4.8.30-xenial4 [572 kB]
Get:2 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main amd64 libdb4.8
  -dev amd64 4.8.30-xenial4 [632 kB]
Fetched 1,204 kB in 5s (236 kB/s)
Selecting previously unselected package libdb4.8.
(Reading database ... 110971 files and directories currently installed.)
Preparing to unpack .../libdb4.8_4.8.30-xenial4_amd64.deb ...
Unpacking libdb4.8 (4.8.30-xenial4) ...
Selecting previously unselected package libdb4.8-dev.
Preparing to unpack .../libdb4.8-dev_4.8.30-xenial4_amd64.deb ...
Unpacking libdb4.8-dev (4.8.30-xenial4) ...
Setting up libdb4.8 (4.8.30-xenial4) ...
Setting up libdb4.8-dev (4.8.30-xenial4) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
pushinode@masternode1:~$
```



18. sudo apt-get install libdb4.8+-dev

```
pushinode@masternode1:~$ sudo apt-get install libdb4.8+-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libdb4.8++
The following NEW packages will be installed:
  libdb4.8++ libdb4.8+-dev
0 upgraded, 2 newly installed, 0 to remove and 3 not upgraded.
Need to get 1,249 kB of archives.
After this operation, 7,344 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main amd64 libdb4.8++ amd64 4.8.30-xenial4 [602 kB]
Get:2 http://ppa.launchpad.net/bitcoin/bitcoin/ubuntu xenial/main amd64 libdb4.8+-dev amd64 4.8.30-xenial4 [647 kB]
Fetched 1,249 kB in 4s (252 kB/s)
Selecting previously unselected package libdb4.8++.
(Reading database ... 110985 files and directories currently installed.)
Preparing to unpack .../libdb4.8++_4.8.30-xenial4_amd64.deb ...
Unpacking libdb4.8++ (4.8.30-xenial4) ...
Selecting previously unselected package libdb4.8+-dev.
Preparing to unpack .../libdb4.8+-dev_4.8.30-xenial4_amd64.deb ...
Unpacking libdb4.8+-dev (4.8.30-xenial4) ...
Setting up libdb4.8++ (4.8.30-xenial4) ...
Setting up libdb4.8+-dev (4.8.30-xenial4) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
pushinode@masternode1:~$
```



19. sudo apt-get install libminiupnpc-dev

```
pushinode@masternode1:~$ sudo apt-get install libminiupnpc-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libminiupnpc10
Suggested packages:
  minissdpc
The following NEW packages will be installed:
  libminiupnpc-dev libminiupnpc10
0 upgraded, 2 newly installed, 0 to remove and 3 not upgraded.
Need to get 52.3 kB of archives.
After this operation, 216 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libminiupnpc10
amd64 1.9.20140610-2ubuntu2.16.04.2 [23.9 kB]
Get:2 http://archive.ubuntu.com/ubuntu xenial-updates/main amd64 libminiupnpc-de
v amd64 1.9.20140610-2ubuntu2.16.04.2 [28.4 kB]
Fetched 52.3 kB in 0s (108 kB/s)
Selecting previously unselected package libminiupnpc10:amd64.
(Reading database ... 110998 files and directories currently installed.)
Preparing to unpack .../libminiupnpc10_1.9.20140610-2ubuntu2.16.04.2_amd64.deb .
..
Unpacking libminiupnpc10:amd64 (1.9.20140610-2ubuntu2.16.04.2) ...
Selecting previously unselected package libminiupnpc-dev.
Preparing to unpack .../libminiupnpc-dev_1.9.20140610-2ubuntu2.16.04.2_amd64.deb
...
Unpacking libminiupnpc-dev (1.9.20140610-2ubuntu2.16.04.2) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Setting up libminiupnpc10:amd64 (1.9.20140610-2ubuntu2.16.04.2) ...
Setting up libminiupnpc-dev (1.9.20140610-2ubuntu2.16.04.2) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
pushinode@masternode1:~$
```

Step 6.2 - Create Swap Space (Important - otherwise, you may fail to compile and install Pushi):

sudo fallocate -l 3G /swapfile

sudo chmod 600 /swapfile

sudo mkswap /swapfile

sudo swapon /swapfile

```
pushinode@masternode1:~$ sudo fallocate -l 3G /swapfile
pushinode@masternode1:~$ sudo chmod 600 /swapfile
pushinode@masternode1:~$ sudo mkswap /swapfile
Setting up swappiness version 1, size = 3 GiB (3221221376 bytes)
no label, UUID=5c9dae5c-b5f6-42fa-865c-ba35d5e6fc87
pushinode@masternode1:~$ sudo swapon /swapfile
```




```
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab (if you are running as pushinode)
```

```
echo -e "/swapfile none swap sw 0 0\n" >> /etc/fstab (if you are running as root)
```

```
pushinode@masternode1:~$ echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
/swapfile none swap sw 0 0
```

Some VPS providers don't allow individual swap files. If you are in this situation you will have to ask them what you need to do, to be able to create a swapfile.

If any of the above-mentioned commands fail, go back step 6.1 and reinstall ALL dependencies one by one.

Step 6.3 - Download the Pushi binaries

Each line is one command, do them line by line.

```
git clone https://github.com/pushiplay/pushi.git
```

```
cd pushi
```

```
pushinode@masternode1:~$ git clone https://github.com/pushiplay/pushi.git
Cloning into 'pushi'...
remote: Counting objects: 90770, done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 90770 (delta 34), reused 33 (delta 33), pack-reused 90723
Receiving objects: 100% (90770/90770), 88.92 MiB | 18.11 MiB/s, done.
Resolving deltas: 100% (65515/65515), done.
Checking connectivity... done.
pushinode@masternode1:~$ cd pushi
pushinode@masternode1:~/pushi$
```



./autogen.sh

```
Makefile.am: installing 'build-aux/depcomp'
parallel-tests: installing 'build-aux/test-driver'
libtoolize: putting auxiliary files in AC_CONFIG_AUX_DIR, 'build-aux'.
libtoolize: copying file 'build-aux/ltmain.sh'
libtoolize: putting macros in AC_CONFIG_MACRO_DIRS, 'build-aux/m4'.
libtoolize: copying file 'build-aux/m4/libtool.m4'
libtoolize: copying file 'build-aux/m4/ltoptions.m4'
libtoolize: copying file 'build-aux/m4/ltugar.m4'
libtoolize: copying file 'build-aux/m4/ltversion.m4'
libtoolize: copying file 'build-aux/m4/lt~obsolete.m4'
configure.ac:10: installing 'build-aux/compile'
configure.ac:5: installing 'build-aux/config.guess'
configure.ac:5: installing 'build-aux/config.sub'
configure.ac:9: installing 'build-aux/install-sh'
configure.ac:9: installing 'build-aux/missing'
Makefile.am: installing 'build-aux/depcomp'
parallel-tests: installing 'build-aux/test-driver'
libtoolize: putting auxiliary files in AC_CONFIG_AUX_DIR, 'build-aux'.
libtoolize: copying file 'build-aux/ltmain.sh'
libtoolize: putting macros in AC_CONFIG_MACRO_DIRS, 'build-aux/m4'.
libtoolize: copying file 'build-aux/m4/libtool.m4'
libtoolize: copying file 'build-aux/m4/ltoptions.m4'
libtoolize: copying file 'build-aux/m4/ltugar.m4'
libtoolize: copying file 'build-aux/m4/ltversion.m4'
libtoolize: copying file 'build-aux/m4/lt~obsolete.m4'
configure.ac:63: installing 'build-aux/compile'
configure.ac:15: installing 'build-aux/config.guess'
configure.ac:15: installing 'build-aux/config.sub'
configure.ac:25: installing 'build-aux/install-sh'
configure.ac:25: installing 'build-aux/missing'
Makefile.am:5: warning: user variable 'GZIP_ENV' defined here ...
/usr/share/automake-1.15/am/distdir.am: ... overrides Automake variable 'GZIP_ENV' defined here
Makefile.am:58: warning: user target 'distcleancheck' defined here ...
/usr/share/automake-1.15/am/distdir.am: ... overrides Automake target 'distcleancheck' defined here
src/Makefile.am: installing 'build-aux/depcomp'
src/Makefile.am:542: warning: user target '.mm.o' defined here ...
/usr/share/automake-1.15/am/depend2.am: ... overrides Automake target '.mm.o' defined here
parallel-tests: installing 'build-aux/test-driver'
pushinode@masternode1:~/pushi$
```



./configure

```
checking whether make supports nested variables... (cached) yes
checking for pkg-config... /usr/bin/pkg-config
checking pkg-config is at least version 0.9.0... yes
checking for ar... /usr/bin/ar
checking for ranlib... /usr/bin/ranlib
checking for strip... /usr/bin/strip
checking for gcc... gcc
checking whether we are using the GNU C compiler... (cached) yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... (cached) none needed
checking whether gcc understands -c and -o together... (cached) yes
checking dependency style of gcc... (cached) gcc3
checking how to run the C preprocessor... gcc -E
checking for gcc option to accept ISO C89... (cached) none needed
checking if gcc supports -std=c89 -pedantic -Wall -Wextra -Wcast-align -Wnested-externs -Wshadow -Wstrict-prototypes -Wno-unused-function -Wno-long-long -Wno-overlength-strings... yes
checking if gcc supports -fvisibility=hidden... yes
checking for __int128... yes
checking for __builtin_expect... yes
checking for x86_64 assembly availability... yes
checking for CRYPTO... yes
checking for main in -lcrypto... yes
checking for EC functions in libcrypto... yes
checking whether byte ordering is bigendian... no
configure: Using assembly optimizations: x86_64
configure: Using field implementation: 64bit
configure: Using bignum implementation: no
configure: Using scalar implementation: 64bit
configure: Using endomorphism optimizations: no
configure: Building ECDH module: no
configure: Building Schnorr signatures module: no
configure: Building ECDSA pubkey recovery module: yes
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating Makefile
config.status: creating libsecp256k1.pc
config.status: creating src/libsecp256k1-config.h
config.status: executing depfiles commands
config.status: executing libtool commands
Fixing libtool for -rpath problems.
pushinode@masternode1:~/pushi$
```



make (this process can take up to 45 minutes, depending on the spec of your VPS)

```
CXX      libbitcoin_server_a-init.o
init.cpp: In function 'bool AppInit2(boost::thread_group&, CScheduler&)':
init.cpp:1667:56: warning: 'nStart' may be used uninitialized in this function [-Wmaybe-uninitialized]
    LogPrintf(" block index %15dms\n", GetTimeMillis() - nStart);
                                     ^
```

^^^^ Ignore the warning above ^^^^

```
CXX      test/test_test_pushi-net_tests.o
CXX      test/test_test_pushi-netbase_tests.o
CXX      test/test_test_pushi-pmt_tests.o
CXX      test/test_test_pushi-policyestimator_tests.o
CXX      test/test_test_pushi-pow_tests.o
CXX      test/test_test_pushi-prevector_tests.o
CXX      test/test_test_pushi-ratecheck_tests.o
CXX      test/test_test_pushi-reverselock_tests.o
CXX      test/test_test_pushi-rpc_tests.o
CXX      test/test_test_pushi-sanity_tests.o
CXX      test/test_test_pushi-scheduler_tests.o
CXX      test/test_test_pushi-script_P2SH_tests.o
CXX      test/test_test_pushi-script_P2PKH_tests.o
CXX      test/test_test_pushi-script_tests.o
CXX      test/test_test_pushi-scriptnum_tests.o
CXX      test/test_test_pushi-serialize_tests.o
CXX      test/test_test_pushi-sighash_tests.o
CXX      test/test_test_pushi-sigopcount_tests.o
CXX      test/test_test_pushi-skiplist_tests.o
CXX      test/test_test_pushi-streams_tests.o
CXX      test/test_test_pushi-test_pushi.o
CXX      test/test_test_pushi-timedata_tests.o
CXX      test/test_test_pushi-transaction_tests.o
CXX      test/test_test_pushi-txvalidationcache_tests.o
CXX      test/test_test_pushi-versionbits_tests.o
CXX      test/test_test_pushi-uint256_tests.o
CXX      test/test_test_pushi-univalue_tests.o
CXX      test/test_test_pushi-util_tests.o
CXX      test/test_test_pushi-accounting_tests.o
CXX      wallet/test/test_test_pushi-wallet_tests.o
CXX      test/test_test_pushi-rpc_wallet_tests.o
CXXLD    test/test_pushi
CXX      bench/bench_bench_pushi-bench_pushi.o
CXX      bench/bench_bench_pushi-bench.o
CXX      bench/bench_bench_pushi-Examples.o
CXXLD    bench/bench_pushi
make[2]: Leaving directory '/home/pushinode/pushi/src'
make[1]: Leaving directory '/home/pushinode/pushi/src'
make[1]: Entering directory '/home/pushinode/pushi'
make[1]: Nothing to be done for 'all-am'.
make[1]: Leaving directory '/home/pushinode/pushi'
pushinode@masternode1:~/pushi$
```



sudo make install

```
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin:/sbin" ldconfig -n /usr/local/lib
```

Libraries have been installed in:
 /usr/local/lib

If you ever happen to want to link against installed libraries in a given directory, LIBDIR, you must either use libtool, and specify the full pathname of the library, or use the '-LLIBDIR' flag during linking and do at least one of the following:

- add LIBDIR to the 'LD_LIBRARY_PATH' environment variable during execution
- add LIBDIR to the 'LD_RUN_PATH' environment variable during linking
- use the ' -D __LIBTOOL_IS_A_FOOL__ ' linker flag
- have your system administrator add LIBDIR to '/etc/ld.so.conf'

See any operating system documentation about shared libraries for more information, such as the ld(1) and ld.so(8) manual pages.

/bin/mkdir -p '/usr/local/bin'
/bin/bash ../libtool --mode=install /usr/bin/install -c pushid pushi-cli pushi-tx test/test_pushi bench/bench_pushi '/usr/local/bin'
libtool: install: /usr/bin/install -c pushid /usr/local/bin/pushid
libtool: install: /usr/bin/install -c pushi-cli /usr/local/bin/pushi-cli
libtool: install: /usr/bin/install -c pushi-tx /usr/local/bin/pushi-tx
libtool: install: /usr/bin/install -c test/test_pushi /usr/local/bin/test_pushi
libtool: install: /usr/bin/install -c bench/bench_pushi /usr/local/bin/bench_pushi
/bin/mkdir -p '/usr/local/include'
/usr/bin/install -c -m 644 script/pushiconsensus.h '/usr/local/include'
make[3]: Leaving directory '/home/pushinode/pushi/src'
make[2]: Leaving directory '/home/pushinode/pushi/src'
make[1]: Leaving directory '/home/pushinode/pushi/src'
make[1]: Entering directory '/home/pushinode/pushi'
make[2]: Entering directory '/home/pushinode/pushi'
make[2]: Nothing to be done for 'install-exec-am'.
/bin/mkdir -p '/usr/local/lib/pkgconfig'
/usr/bin/install -c -m 644 libpushiconsensus.pc '/usr/local/lib/pkgconfig'
make[2]: Leaving directory '/home/pushinode/pushi'
make[1]: Leaving directory '/home/pushinode/pushi'
pushinode@masternode1:~/pushi\$



```
cd ~/pushi/src/  
./pushid  
pushinode@masternode1:~/pushi$ cd ~/pushi/src  
pushinode@masternode1:~/pushi/src$ ./pushid  
█
```

You will see that the cursor is stuck. The reason is pushi daemon is starting for the first time. It is getting ready all the necessary folder and immediately trying to sync to the blockchain.

You can **CTRL+C** to stop this process and continue with the rest of the guide.

```
pushinode@masternode1:~/pushi$ cd ~/pushi/src  
pushinode@masternode1:~/pushi/src$ ./pushid  
^Cpushinode@masternode1:~/pushi/src$ █
```

Step 7 - Editing pushi.conf and running Pushi

Now we'll create a configuration file for Pushi that has a few important details. We'll use a tool in Linux called 'nano' to edit the file.

nano ~/.pushicore/pushi.conf

```
pushinode@masternode1:~/pushi/src$ nano ~/.pushicore/pushi.conf █
```

This will open an empty file in the text editor nano. Enter in the following lines (**NOTE:** Do not leave any trailing spaces at the end of this file, this has caused issues for some users):

```
rpcuser=YOUR_USER_NAME  
rpcpassword=YOUR_PASSWORD  
rpcallowip=127.0.0.1  
listen=1  
server=1  
daemon=1  
maxconnections=64  
masternode=1  
logtimestamps=1  
masternodeprivkey=MN_GENKEY  
externalip=VPS_IP:9847
```

rpcpassword can be any combination of letters and numbers. You don't need to remember this password and won't need to enter it anywhere. **Just be sure to NOT use any special characters, spaces or symbols in here.....letters and numbers only!**

externalip should match the IP address of your vps

masternodeprivkey should match the private key that we generated earlier in the debug console.



```
pushinode@masternode1: ~/pushi/src
GNU nano 2.5.3 File: /home/pushinode/.pushicore/pushi.conf Modified

rpcuser=someone
rpcpassword=asdfsiaj762
rpcallowip=127.0.0.1
listen=1
server=1
daemon=1
maxconnections=64
masternode=1
logtimestamps=1
masternodeprivkey=2rq
externalip=:9847

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

Press **Ctrl-X** to "Write out" and "Exit" the file in nano and press Y then Enter to accept the default path:

```
File Name to Write: /home/pushinode/.pushicore/pushi.conf
^G Get Help M-D DOS Format M-A Append M-B Backup File
^C Cancel M-M Mac Format M-P Prepend ^T To Files
```



Then press **Ctrl-X** to exit nano. You should now be back at a command line and ready to start Pushi on your vps. Type in:

Cd ~/pushi/src
./pushid

```
pushinode@masternode1:~/pushi/src$ ./pushid
Pushi Core server starting
pushinode@masternode1:~/pushi/src$
```

This starts Pushi and it should now be running in the background. If you ever need to restart your VPS or restart Pushi, this same command can be used to start the Pushi daemon.

To send commands to Pushi, we use the binary **pushi-cli**. The same commands that are available in the debug console of your local wallet are also available with pushi-cli. To check that Pushi is properly connected to the network, type in:

~/pushi/src/pushi-cli getinfo

```
pushinode@masternode1:~/pushi/src$ ./pushi-cli getinfo
{
  "version": 1010600,
  "protocolversion": 70210,
  "walletversion": 61000,
  "balance": 0.00000000,
  "privatesend_balance": 0.00000000,
  "blocks": 41041,
  "timeoffset": 0,
  "connections": 8,
  "proxy": "",
  "difficulty": 5317.097875216805,
  "testnet": false,
  "keypoololdest": 1524311034,
  "keypoolsize": 999,
  "paytxfee": 0.00000000,
  "relayfee": 0.00001000,
  "errors": ""
}
pushinode@masternode1:~/pushi/src$
```

This issues Pushi-cli the 'getinfo' command, which lists some basic information. Mainly, we're looking to see that "connections" has at least 1 connection (hopefully more).

Another helpful command is:

~/pushi/src/pushi-cli mnsync status



```
rapturenode@masternode1:~$ ~/rapturecore-1.0.0/bin/rapture-cli mnsync status
{
  "AssetID": 999,
  "AssetName": "MASTERNODE_SYNC_FINISHED",
  "AssetStartTime": 1517502939,
  "Attempt": 0,
  "IsBlockchainSynced": true,
  "IsMasternodeListSynced": true,
  "IsWinnersListSynced": true,
  "IsSynced": true,
  "IsFailed": false
}
```

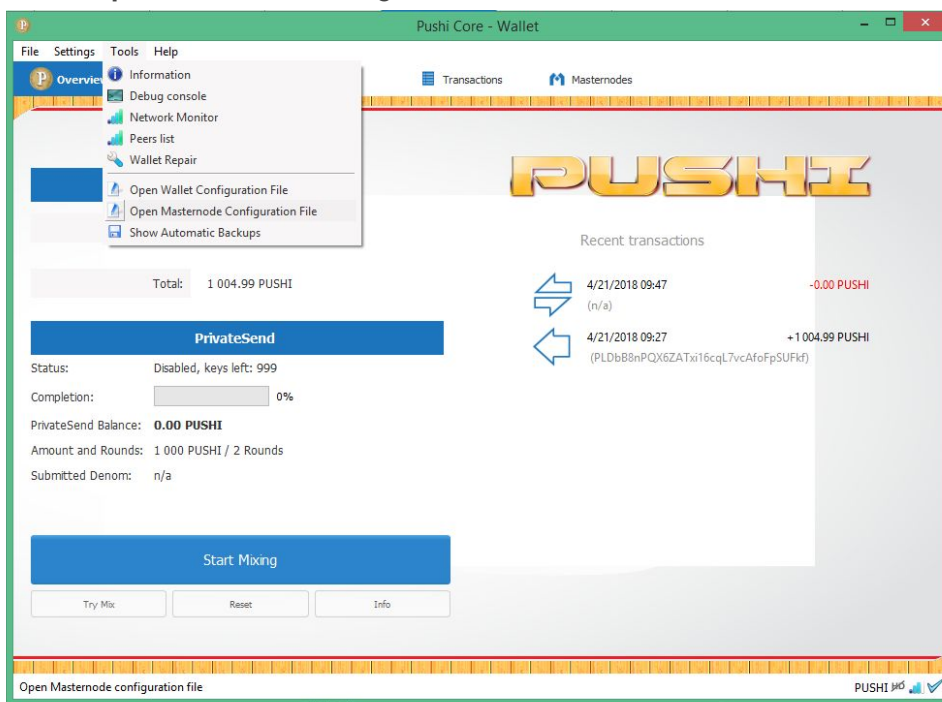
This checks the current status of the wallet sync. When you launch your local wallet, you'll notice that it takes a minute or so to catch up to the latest block and also sync important information about the network, such as the current list of masternodes on the network. Your vps wallet does the exact same thing, but because it's run through the command line, there's no status meter to watch. This mnsync status command can be run multiple times to check how far along the sync is. You're looking for the line "isSynced" to be true, which means that your wallet is fully synced with the blockchain and network.

Step 8 - Creating masternode entry in your local wallet

With Pushi up and running on your VPS, we'll jump back over to your local wallet to get it communicating with the VPS. Keep your VPS PuTTY session up and running, we'll be back to take care of one more important task there shortly.

On your local wallet, we need to give it the configuration information for your masternode. This is done by editing your masternode.conf file. This file is generally stored in %APPDATA%/PushiCore/masternode.conf, but an easier way to edit it is to go to:

Tools->Open Masternode Configuration file



This should open up your masternode.conf file in Notepad.

This file has an example line in that can be used as a guide. The # in front of the line indicates that it's "commented out", which means it's effectively ignored. Following this example, we see that the structure is:

alias - a name for your masternode

ip address - the IP address for your VPS that is hosting the node

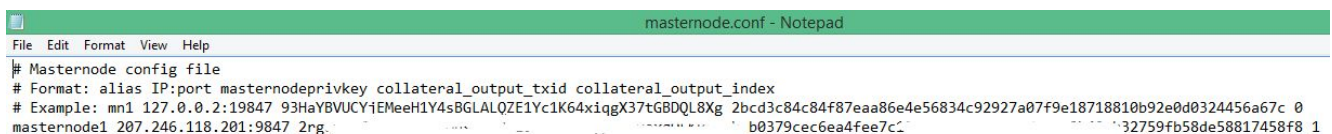
masternode private key - the key that we generated previously using 'masternode genkey'

txid - the transaction id for your 1,000 Pushi collateral transaction

index - the block position of the transaction...this is the number that appears at the end of of the txid listed by 'masternode outputs'

Following this structure, for our example we're going to add a line that looks like this: (Remember all this is in one line. Make sure you don't have any return characters that made the info into two lines)

masternode1 207.246.118.201:9847 2rg<cut for privacy reasons> b0379cec6ea4fee7c<cut for privacy reasons>2759fb58de58817458f8 1



```
# Masternode config file
# Format: alias IP:port masternodeprivkey collateral_output_txid collateral_output_index
# Example: mn1 127.0.0.2:19847 93HaYBVUCYiEMeeH1Y4sBGLALQZE1Yc1K64xiqgX37tG8DQL8Xg 2bcd3c84c84f87eaa86e4e56834c92927a07f9e18718810b92e0d0324456a67c 0
masternode1 207.246.118.201:9847 2rg. b0379cec6ea4fee7c2 32759fb58de58817458f8 1
```

Double check your entries to make sure they all match up, and then Save and close notepad. Every time you edit or make changes to the masternode.conf file (or any conf file), you need to close and re-open the wallet to initialize the changes. So close your local wallet, and then re-open.

Once you've re-opened, over on the Masternodes tab you should see your node listed.



Pushi Core - Wallet

File Settings Tools Help

Overview Send Receive Transactions **Masternodes**

My Masternodes All Masternodes

Note: Status of your masternodes in local wallet can potentially be slightly incorrect. Always wait for wallet to sync additional data and then double check from another node if your masternode should be running but you still do not see "ENABLED" in "Status" field.

| Alias | Address | Protocol | Status | Active | Last Seen | Payee |
|-------------|----------------------|----------|---------|---------|------------------|-------|
| masternode1 | 207.246.118.201:9847 | -1 | MISSING | 00m:00s | 1970-01-01 00:00 | |

Start alias Start all Start MISSING Update status Status will be updated automatically in (sec): 42

Click the **Start alias** button and then **click Yes** to confirm. (**NOTE**, before clicking start, make sure your transaction of 1,000 Pushi has **AT LEAST 15 confirmations**, you can check the number of confirmations by hovering the mouse cursor over the transaction in the **Transactions** tab)

| Alias | Address | Protocol | Status | Active | Last Seen | Payee |
|-------------|----------------------|----------|---------|---------|------------------|-------|
| masternode1 | 207.246.118.201:9847 | -1 | MISSING | 00m:00s | 1970-01-01 00:00 | |

Confirm masternode start

Are you sure you want to start masternode masternode1?

Yes Cancel

This remotely connects to your VPS and uses your private key and txid to activate the masternode. If everything has been configured properly, you should see the following message:



My Masternodes

All Masternodes

Note: Status of your masternodes in local wallet can potentially be slightly incorrect. Always wait for wallet to sync additional data and then double check from another node if your masternode should be running but you still do not see "ENABLED" in "Status" field.

| Alias | Address | Protocol | Status | Active | Last Seen | Payee |
|-------------|----------------------|----------|---------|---------|------------------|-------|
| masternode1 | 207.246.118.201:9847 | -1 | MISSING | 00m:00s | 1970-01-01 00:00 | |

Pushi-Qt

Alias: masternode1

Successfully started masternode.

OK

Step 9 - Setting up sentinel

At this point, your masternode has been enabled and will start broadcasting itself to the network, but will likely be showing WATCHDOG_EXPIRED status.

your masternode should be running but you still do not see "ENABLED" in "Status" field.

| Alias | Address | Protocol | Status | Active | Last Seen | Payee |
|-------------|----------------------|----------|------------------|---------|------------------|-----------------------------|
| masternode1 | 207.246.118.201:9847 | 70210 | WATCHDOG_EXPIRED | 00m:00s | 2018-04-21 21:23 | PN4yCqCLVfqoSbm75VKbDKvv... |

There's one more important step before your node is fully enabled and eligible for masternode rewards and voting. We need to configure sentinel, which is a service that interfaces with the masternode. If this step is skipped, your masternode will continue to display WATCHDOG_EXPIRED status and will not receive rewards....so onwards with this important step!

Back over to your VPS now (if your PuTTY session disconnected, just reconnect and login with your 'pushinode' user as before).

First, we want to install a package called virtualenv. From the command line, type:

```
cd ~/pushi/src
python --version
```

```
pushinode@masternode1:~/pushi/src$ python --version
Python 2.7.12
pushinode@masternode1:~/pushi/src$
```




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

```
Selecting previously unselected package python-pip-whl.
(Reading database ... 111017 files and directories currently installed.)
Preparing to unpack .../python-pip-whl_8.1.1-2ubuntu0.4_all.deb ...
Unpacking python-pip-whl (8.1.1-2ubuntu0.4) ...
Selecting previously unselected package python-pkg-resources.
Preparing to unpack .../python-pkg-resources_20.7.0-1_all.deb ...
Unpacking python-pkg-resources (20.7.0-1) ...
Selecting previously unselected package python-virtualenv.
Preparing to unpack .../python-virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking python-virtualenv (15.0.1+ds-3ubuntu1) ...
Selecting previously unselected package python3-virtualenv.
Preparing to unpack .../python3-virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking python3-virtualenv (15.0.1+ds-3ubuntu1) ...
Selecting previously unselected package virtualenv.
Preparing to unpack .../virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking virtualenv (15.0.1+ds-3ubuntu1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up python-pip-whl (8.1.1-2ubuntu0.4) ...
Setting up python-pkg-resources (20.7.0-1) ...
Setting up python-virtualenv (15.0.1+ds-3ubuntu1) ...
Setting up python3-virtualenv (15.0.1+ds-3ubuntu1) ...
Setting up virtualenv (15.0.1+ds-3ubuntu1) ...
pushinode@masternode1:~/pushi/src$
```

This should output some information and ultimately finish back at the command line.

Next we'll clone sentinel from the PushiCore github repository and go inside the sentinel folder using the following command:

```
git clone https://github.com/pushisplay/sentinel.git && cd sentinel
```

```
pushinode@masternode1:~/pushi/src$ git clone https://github.com/pushisplay/sentinel.git && cd sentinel
Cloning into 'sentinel'...
remote: Counting objects: 3461, done.
remote: Total 3461 (delta 0), reused 0 (delta 0), pack-reused 3461
Receiving objects: 100% (3461/3461), 1.20 MiB | 0 bytes/s, done.
Resolving deltas: 100% (2302/2302), done.
Checking connectivity... done.
pushinode@masternode1:~/pushi/src/sentinel$
```



Next, we'll run a command to setup a python virtual environment in the sentinel directory. Type in:

virtualenv ./venv

```
pushinode@masternode1:~/pushi/src/sentinel$ virtualenv ./venv
Running virtualenv with interpreter /usr/bin/python2
New python executable in /home/pushinode/pushi/src/sentinel/venv/bin/python2
Also creating executable in /home/pushinode/pushi/src/sentinel/venv/bin/python
Installing setuptools, pkg_resources, pip, wheel...done.
pushinode@masternode1:~/pushi/src/sentinel$
```

This command creates a directory in your current sentinel directory that contains the python executables and libraries. Next, we'll install the requirements and dependencies for sentinel.



Run:

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

```
51187bd5dfdeafed7a6bd22ea87e601cbd961dd/peewee-2.8.3.tar.gz (501kB)
 100% |████████████████████████████████████████| 512kB 31.0MB/s
Collecting py==1.4.31 (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/19/f2/4b71181a49a4673a12c8f5075b8744c5feb0ed9eba352dd22512d2c04d47/py-1.4.31-py2.py3-none-any.whl (81kB)
 100% |████████████████████████████████████████| 92kB 35.1MB/s
Collecting pycodestyle==2.3.1 (from -r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/e4/81/78fe51eb4038d1388b7217dd63770b0f428370207125047312886c923b26/pycodestyle-2.3.1-py2.py3-none-any.whl (45kB)
 100% |████████████████████████████████████████| 51kB 25.8MB/s
Collecting pytest==3.0.1 (from -r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/00/b1/62d1da704e18603026245380b2407387b3ec7b0d3e3399c98933f2487d2c/pytest-3.0.1-py2.py3-none-any.whl (169kB)
 100% |████████████████████████████████████████| 174kB 38.8MB/s
Collecting python-bitcoinrpc==1.0 (from -r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/71/eb/80313654302a7f3ba2fa3a8ea9e4c65f910b353112df8da77e4974978665/python-bitcoinrpc-1.0.tar.gz
Collecting simplejson==3.8.2 (from -r requirements.txt (line 7))
  Downloading https://files.pythonhosted.org/packages/f0/07/26b519e6ebb03c2a74989f7571e6ae6b82e9d7d81b8de6fcdbfc643c7b58/simplejson-3.8.2.tar.gz (76kB)
 100% |████████████████████████████████████████| 81kB 32.5MB/s
Building wheels for collected packages: inflection, peewee, python-bitcoinrpc, simplejson
  Running setup.py bdist_wheel for inflection ... done
  Stored in directory: /home/pushinode/.cache/pip/wheels/9f/5a/d3/6fc3bf6516d2a3eb7e18f9f28b472110b59325f3f258fe9211
  Running setup.py bdist_wheel for peewee ... done
  Stored in directory: /home/pushinode/.cache/pip/wheels/d0/81/75/29fe10f49530573d2a23975d57d2d511d2e9924397228f0728
  Running setup.py bdist_wheel for python-bitcoinrpc ... done
  Stored in directory: /home/pushinode/.cache/pip/wheels/b8/4f/ed/3ee51a2291b290d9f485af88fd718559b5aa00b761f531abcc
  Running setup.py bdist_wheel for simplejson ... done
  Stored in directory: /home/pushinode/.cache/pip/wheels/b0/2a/43/57bc8cc4565cda29e70b3a7a427d73f43d98fedb9celb1d00c
Successfully built inflection peewee python-bitcoinrpc simplejson
Installing collected packages: inflection, peewee, py, pycodestyle, pytest, python-bitcoinrpc, simplejson
Successfully installed inflection-0.3.1 peewee-2.8.3 py-1.4.31 pycodestyle-2.3.1 pytest-3.0.1 python-bitcoinrpc-1.0 simplejson-3.8.2
pushinode@masternode1:~/pushi/src/sentinel$
```



That will display information on the packages being installed and then drop you back to the command line. At this point, we can run sentinel for the first time and make sure that it runs properly. Type:

./venv/bin/py.test ./test

```
pushinode@masternode1:~/pushi/src/sentinel$ ./venv/bin/py.test ./test
===== test session starts =====
platform linux2 -- Python 2.7.12, pytest-3.0.1, py-1.4.31, pluggy-0.3.1
rootdir: /home/pushinode/pushi/src/sentinel, inifile:
collected 23 items

test/integration/test_jsonrpc.py .
test/unit/test_misc.py .
test/unit/test_models.py ..
test/unit/test_pushi_config.py .
test/unit/test_pushid_data_shims.py ..
test/unit/test_pushiy_things.py .....
test/unit/test_submit_command.py .
test/unit/models/test_proposals.py ....
test/unit/models/test_superblocks.py .....

===== 23 passed in 0.31 seconds =====
pushinode@masternode1:~/pushi/src/sentinel$
```

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

```
pushinode@masternode1:~/pushi/src/sentinel$ ./venv/bin/python bin/sentinel.py
pushinode@masternode1:~/pushi/src/sentinel$
```

When sentinel is running properly, it will process for a few seconds and then return to the command line without any output. If you receive any error messages, take a look at the [Troubleshooting](#) section of the guide.

If sentinel runs without any errors, then we can proceed with the final step, which is setting up what's known as a cron job that schedules the sentinel command to run every minute. To create the cron job, we'll run the following command:

crontab -e

You'll be prompted to choose an editor, we'll use **nano**:




```

pushinode@masternode1:~/pushi/src/sentinel$ crontab -e
no crontab for pushinode - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      <---- easiest
 3. /usr/bin/vim.basic
 4. /usr/bin/vim.tiny

Choose 1-4 [2]: █

```

Down bellow all of the # (commented) lines, we'll create a new line and insert (**NOTE:** If you selected a username OTHER than 'pushinode', edit this path and replace 'pushinode' with your username. If you've deviated from the guide and are logged in and installing as **root**, then the path should be /root/sentinel):

If you are using **pushinode** to login and install Pushi:

```
***** cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1
```

If you are using **root** to login and install Pushi:

```
***** cd /root/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1
```

```

█ Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
* * * * * cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1

```

This basically tells the sentinel command to run every minute. Type:

Ctrl-O and then **Enter**

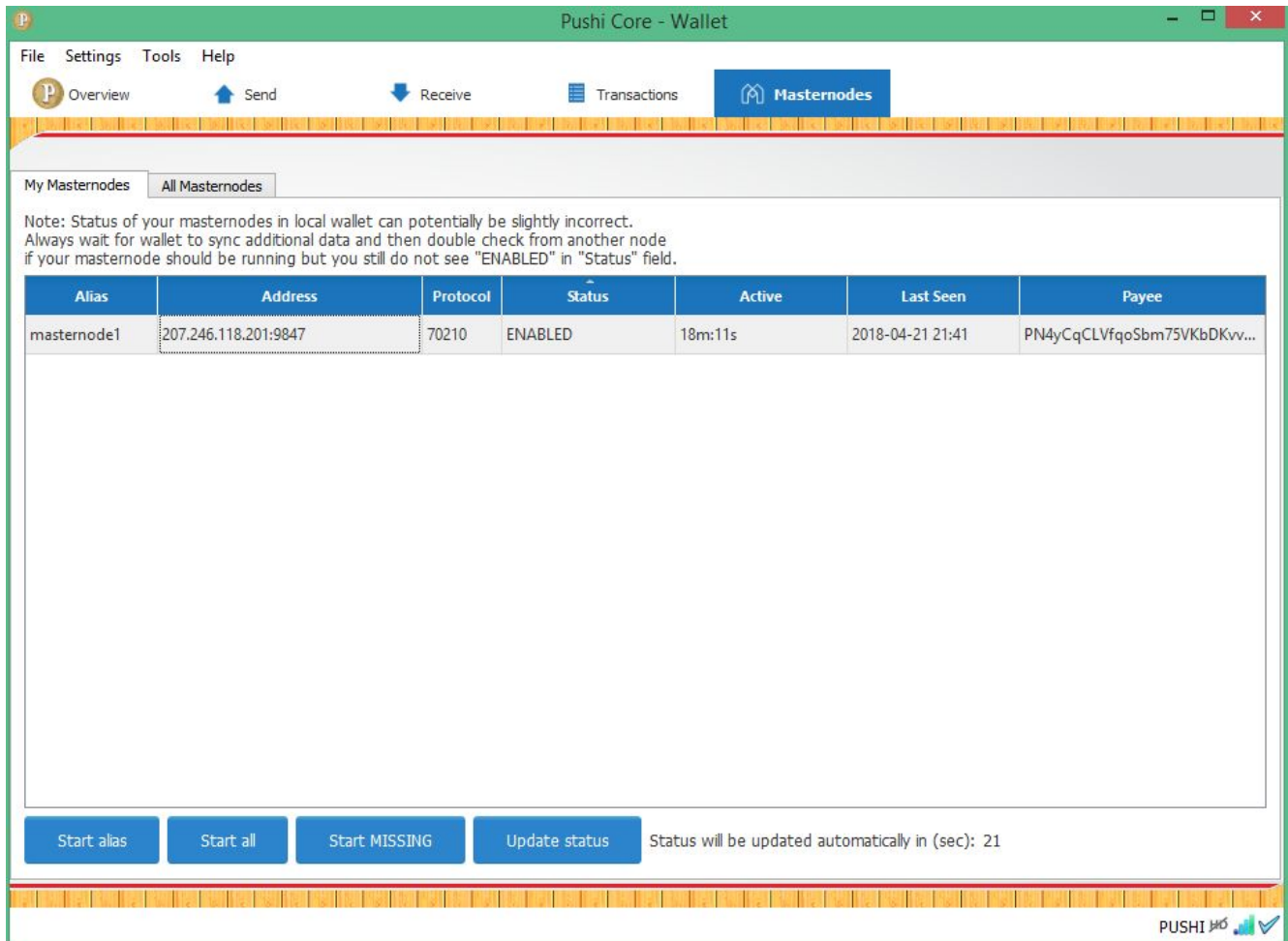
to write out the file. Then:

Ctrl-X to exit



Step 10 - Wait for ENABLED status

At this point we've performed all of the steps in setting up a properly functioning masternode. It will take about 15-20 minutes for the node to switch over to **ENABLED** status:



The screenshot shows the 'Pushi Core - Wallet' application window. The 'Masternodes' tab is selected, displaying a table of masternodes. The table has columns: Alias, Address, Protocol, Status, Active, Last Seen, and Payee. One masternode is listed with the alias 'masternode1', address '207.246.118.201:9847', protocol '70210', and status 'ENABLED'. Below the table, there are buttons for 'Start alias', 'Start all', 'Start MISSING', and 'Update status'. A status message indicates 'Status will be updated automatically in (sec): 21'. A note above the table states: 'Note: Status of your masternodes in local wallet can potentially be slightly incorrect. Always wait for wallet to sync additional data and then double check from another node if your masternode should be running but you still do not see "ENABLED" in "Status" field.'

| Alias | Address | Protocol | Status | Active | Last Seen | Payee |
|-------------|----------------------|----------|---------|---------|------------------|-----------------------------|
| masternode1 | 207.246.118.201:9847 | 70210 | ENABLED | 18m:11s | 2018-04-21 21:41 | PN4yCqCLVfqoSbm75VKbDKvv... |

Buttons: Start alias, Start all, Start MISSING, Update status

Status will be updated automatically in (sec): 21

Once it has switched to ENABLED, you can close your local wallet if you prefer. It does not need to be open in order to receive rewards. If after about 20 minutes, you're seeing anything other than ENABLED, proceed to the Troubleshooting section below.

When will you get rewards? 18-24hr for your first payment and the network to agree that you're a stable node. After that you'll get payments regularly everyday.



TROUBLESHOOTING

Sentinel debugging

The number one issue that users run into tends to be a misconfiguration with sentinel. It could be that the cron task that keeps sentinel running in the background wasn't setup properly, or it could be that sentinel wasn't installed properly to begin with.

Checking the cron log

Whenever a cron task runs, it writes to the system log. You can easily check this by typing:

sudo grep CRON /var/log/syslog

This command checks the system log for entries with CRON in the name. You should see an output like this:

```
pushinode@masternode1:~/pushi/src$ sudo grep CRON /var/log/syslog
Jan 11 00:13:16 guest cron[769]: (CRON) INFO (pidfile fd = 3)
Jan 11 00:13:16 guest cron[769]: (CRON) INFO (Running @reboot jobs)
Apr 21 08:01:20 guest cron[683]: (CRON) INFO (pidfile fd = 3)
Apr 21 08:01:20 guest cron[683]: (CRON) INFO (Running @reboot jobs)
Apr 21 08:17:01 guest CRON[25174]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 09:17:01 guest CRON[5318]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 10:17:02 guest CRON[7393]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 11:02:01 guest CRON[8011]: (root) CMD ( test -x /etc/cron.daily/popularity-contest && /etc/cron.daily/popularity-contest --crond)
Apr 21 11:17:01 guest CRON[8110]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 12:17:01 guest CRON[8577]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 13:17:01 guest CRON[9032]: (root) CMD ( cd / && run-parts --report /etc/cron.hourly)
Apr 21 13:34:01 guest CRON[10018]: (pushinode) CMD (cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1)
Apr 21 13:35:01 guest CRON[10036]: (pushinode) CMD (cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1)
Apr 21 13:36:01 guest CRON[10044]: (pushinode) CMD (cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1)
Apr 21 13:37:01 guest CRON[10068]: (pushinode) CMD (cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1)
Apr 21 13:38:01 guest CRON[10074]: (pushinode) CMD (cd /home/pushinode/pushi/src/sentinel && ./venv/bin/python bin/sentinel.py >/dev/null 2>&1)
pushinode@masternode1:~/pushi/src$
```



Here we can see that the cron task we setup in Step 9 is running every minute. If you're not seeing something similar, or you're seeing error messages, go back to [here](#) and repeat the crontab procedure.

Checking that sentinel runs properly

We can double-check that sentinel is running without issue by running the following commands:

```
cd ~/pushi/src/sentinel
```

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

This should process for a second or two and then return to the command line without any messages or errors. If you receive the message: **Invalid Masternode Status, cannot continue**....then your masternode was not started successfully.

We can check the status of the masternode by running:

```
~/pushi/src/pushi-cli masternode status
```

If the status says anything other than "**Masternode successfully started**", try restarting the masternode by clicking the **Start all** button on the Masternodes tab in your local wallet. Re-check the status by running the previous command:

```
pushinode@masternode1:~/pushi/src$ ./pushi-cli masternode status
{
  "outpoint": "b0379cec6ea4fee7c196a5d7b7ed62417df209b49eb32759fb58de58817458f8-1",
  "service": "207.246.118.201:9847",
  "payee": "PN4yCqCLVfgoSbm75VKbDKvvtonp1bq6S4",
  "status": "Masternode successfully started"
}
pushinode@masternode1:~/pushi/src$
```

if all is configured correctly, then try running sentinel to verify that the problem is fixed:

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

Checking the masternode status

The status of the masternode can be checked on the VPS by running:

```
~/pushi/src/pushi-cli masternode status
```



```
pushinode@masternode1:~/pushi/src$ ./pushi-cli masternode status
{
  "outpoint": "b0379cec6ea4fee7c196a5d7b7ed62417df209b49eb32759fb58de58817458f8-1",
  "service": "207.246.118.201:9847",
  "payee": "PN4yCqCLVfgoSbm75VKbDKvvttonp1bq6S4",
  "status": "Masternode successfully started"
}
pushinode@masternode1:~/pushi/src$ █
```

If everything is configured correctly, this should return the status: **Masternode successfully started**

If you receive another status, such as **"Not capable masternode: Masternode not in masternode list"**, check the following:

- Click **Start all** from the Masternodes tab in your local wallet
- Verify that your collateral transaction of 1,000 Pushi has at least 15 confirmations. The masternode will only start successfully once that transaction has 15 confirmations (you can check this by hovering the mouse cursor over the transaction in the Transactions tab of the local wallet)
- [Check your conf files for any errors](#) (This is important - and occurs mostly with .conf files and RPC Uname and Password errors.

Masternode failed to start - CONF file errors

If you're having issues starting the masternode via the Masternodes tab in your local wallet, best to check your local masternode.conf file and your VPS pushi.conf file to make sure that they match up.

You can open your local masternode.conf file by choosing:

Tools->Open Masternode Configuration File...

...in your local wallet. On your VPS, you can open the pushi.conf file by typing:

nano ~/.pushicore/pushi.conf

Arrange both of these side-by-side and double check that the IP address and masternode private key match.



UPGRADING

Upgrading masternode from v1.1.5 to v1.1.6

On February 22, 2017, a mandatory update was committed to the github repository bringing Pushi up to v1.1.0. This update includes a re-balance between proof-of-work and masternode rewards after block height 29,000. As with any adjustment to block subsidies, the update is mandatory and users MUST upgrade both their LOCAL and MASTERNODE wallets. The local update is very straight forward, just download your preferred binary from:

<https://github.com/pushiplay/pushi/releases/tag/v1.1.6>

If you're using the Windows installer, just install the new version and you should be all set. If you're using the zipped binaries, just delete your old Pushicore-1.0.0 directory and replace it with Pushicore-1.1.0.

To update your Linux VPS masternode, login to your VPS via putty (see the final commands in [Step 5](#) if you forget how to do this) and run the following commands:

First, download the v1.1.6 binaries from github:

Wget <https://github.com/pushiplay/pushi/releases/download/v1.1.6/linux64-cli-v1.1.6.tar.gz>

<screenshot>

Then extract those binaries:

tar -zxvf linux64-cli-v1.1.6.tar.gz

<screenshot>

Next, we'll shutdown the previous version of the Pushi daemon:

~/pushi/src/pushi-cli stop

<screenshot>

Wait a few seconds for the previous daemon to fully shutdown, and then start up v1.1.0:

~/pushi/src/pushid

<screenshot>

You should now be running v1.1.6, you can verify this by running:

~/pushi/src/pushi-cli getinfo



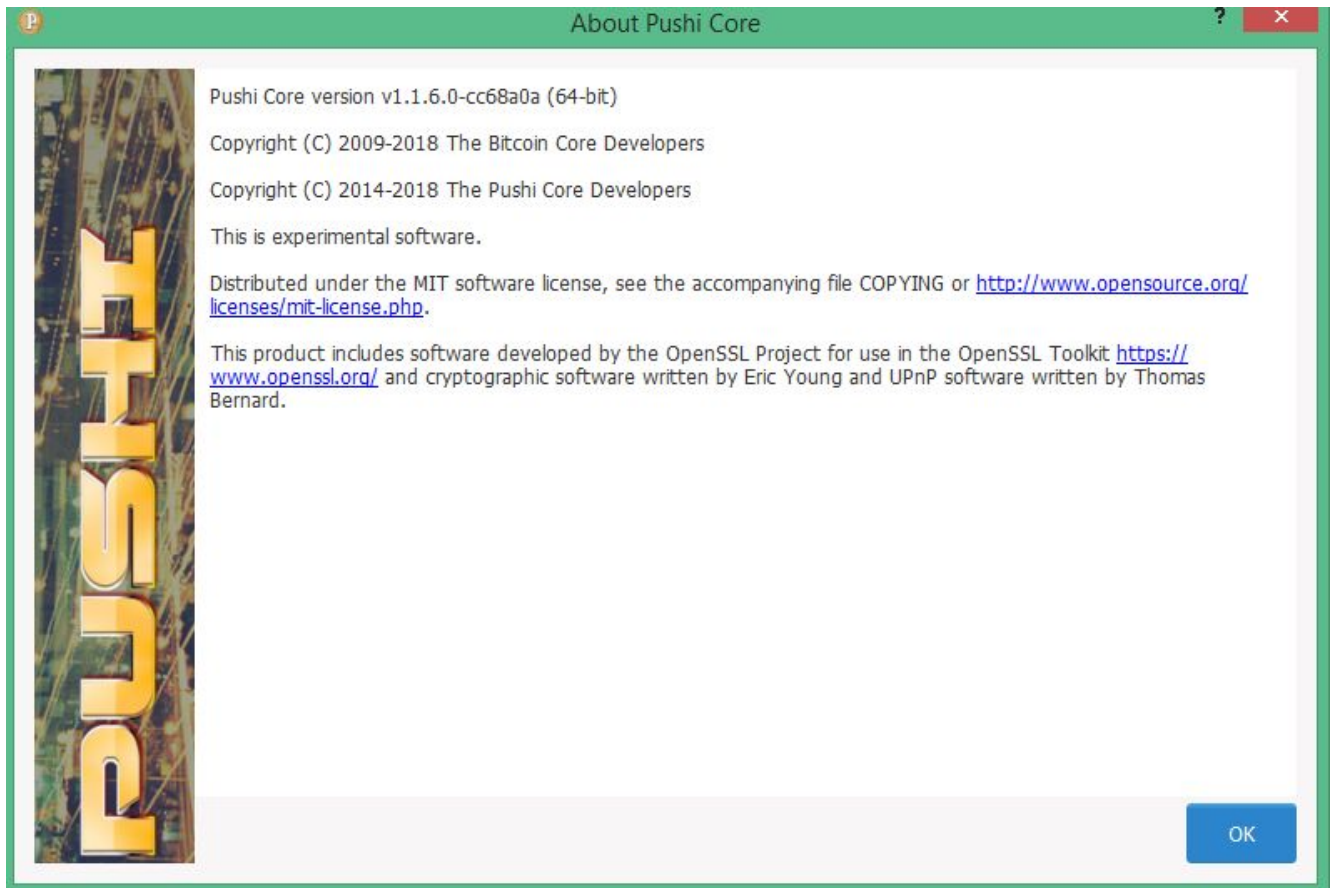

```

pushinode@masternode1:~/pushi/src$ ./pushi-cli getinfo
{
  "version": 1010600,
  "protocolversion": 70210,
  "walletversion": 61000,
  "balance": 0.00000000,
  "privatesend_balance": 0.00000000,
  "blocks": 42677,
  "timeoffset": 0,
  "connections": 12,
  "proxy": "",
  "difficulty": 13478.93772870781,
  "testnet": false,
  "keypoololdest": 1524311034,
  "keypoolsize": 999,
  "paytxfee": 0.00000000,
  "relayfee": 0.00001000,
  "errors": ""
}
pushinode@masternode1:~/pushi/src$ █

```

You'll see the version listed as 1010600 and the protocol version as 70210.

At this point, head back over to your local wallet (which you should have already update to v1.1.6) and start your masternode using **Start alias**:



Your node should show up as successfully started and will switch to **PRE_ENABLED**. PRE_ENABLED should last for ~10 minutes and then switch to **ENABLED**.

You can double-check the status of the masternode on the VPS using:

~/pushi/src/pushi-cli masternode status

```
pushinode@masternode1:~/pushi/src$ ./pushi-cli masternode status
{
  "outpoint": "b0379cec6ea4fee7c196a5d7b7ed62417df209b49eb32759fb58de58817458f8-1",
  "service": "207.246.118.201:9847",
  "payee": "PN4yCqCLVfqoSbm75VKbDKvvtonp1bq6S4",
  "status": "Masternode successfully started"
}
pushinode@masternode1:~/pushi/src$
```

The status should say **"Masternode successfully started"**. If the status reads as **"Not capable masternode: Invalid protocol version"**, make sure that the local wallet that you're activating from has been upgraded to v1.1.6. If so, then this message will clear momentarily and no additional actions are required.

When everything looks good, you can remove v1.1.6 tar.gz file from your VPS by running (paste this exactly as-is):

rm ./linux64*.tar.gz

At this point, you're fully updated to v1.1.6!



APPENDIX

Understanding your Masternode

Masternode Status

PRE_ENABLED: New starts require time (approximately 10 minutes maybe a bit longer) to connect to peers and verify connection, **be patient.**

ENABLED: Working properly. (Occasionally Sentinel will not run in full synchronization and it will go to WATCHDOG_EXPIRED give it a few hours and see if it switches back to ENABLED).

WATCHDOG_EXPIRED: Sentinel is not reporting properly. Check crontab -e and ensure all install file paths are correct. Specifically the path to sentinel needs to match your sentinel location. Find the full path to sentinel by typing: **cd ~/sentinel;pwd**

WATCHDOG_EXPIRED Troubleshooting can be found [here](#)

NEW_START_REQUIRED: Several possible causes. Try restarting using start-alias <your masternode name> and wait up to 30 minutes for ENABLED. You can highlight your MN in the Masternodes Tab and Right Click to do this.

If not begin [troubleshooting](#).

Real-time Masternode status can be checked online here:

<https://masternodes.online/monitoring/>

Commonly Used VPS Console Commands

./pushi-cli - Is the command line interface and used to call functions for specific

Local Wallet Terminology

Collateral Transaction: This is the amount of coins required to operate the Masternode. 1000 Pushi are required for a Pushi Masternode. This amount has to EXACT - NO MORE - NO LESS

[Sending your Collateral Transaction]: You must send exactly 1000 coins to generate a valid Transaction ID(txid) to be used for a Masternode. (When sending a transaction to yourself you will see it in your history "payment to self" as a small fee. 1000 coins will not show here.)

Transaction ID(txid): This value is generated when you successfully send exactly 1000 Pushi to your new receiving address. This process ensures you only have 1000 Pushi in the specified address and allows the wallet to lock those funds in place. These will now be used for your Masternode Collateral.

vout: This can be confusing to understand. This number is often 0, 1 or 2.



(The... blockchain uses a per-output transactional model, in which every transaction has a set of inputs and a set of outputs. Each input "spends" one output of a previous transaction, with the blockchain ensuring that this output cannot be spent elsewhere. The full history of transactions forms a multiway connected chain... All of the [assets] in a transaction's inputs flow into that transaction, which are then distributed across its outputs in accordance with the quantities written within. As a result most regular payments require two outputs – one with the intended amount for the recipient, and the other containing "change" which goes back to the sender for use in a subsequent transaction.)

Private Key: This is essentially a password that your VPS will use to access your local wallet and verify your Collateral Transaction. KEEP THIS SAFE. DO NOT SHARE IT. PEOPLE CAN ACCESS YOUR WALLET WITH THIS KEY. Make sure you have multiple backups of this information and of your wallet.dat file in your Pushi folder on desktop.

Virtual Private Server (VPS)

A VPS is essentially a computer you rent. There are several providers to choose from. I personally use DigitalOcean but Vultr is widely used as well.

(\$10 free credit at Digitalocean with this referral link: <https://m.do.co/c/22b7cd6629d3>)

Renting a VPS is vital to running a Masternode to ensure security/stability/quality of life. When running a "hot/cold wallet" you are able to close your wallet and turn off your local machine while the Masternode stays active. Masternodes also require a static IP and broadcast their IP address to the network. For this reason it's best not host from your local machine.

Hot/Cold Wallet: This term refers to your VPS and Local Machine respectively. Your VPS(Hot) will run your Masternode and stay online while your Local wallet(Cold) is free to go offline.

[Vultr](#) - Currently \$2.50pm 1 CPU, 512MB Mem 500gb bandwidth

[DigitalOcean](#) Currently \$5 pm , 1 CPU, 1GB Mem, 1 Tb bandwidth

[Time4VPS](#) - Make sure they grant you Superuser Access Euro 2.99pm 1 CPU, 512MB Mem, 1 Tb bandwidth

[Amazon EC2](#) - Free for a year and uses VMware management tools.

[Google Cloud Compute](#) Various options and prices - check link.

Putty

Putty is a tool used to access your VPS via ssh. You can input your IP address and connect remotely through the Putty console. This program is relaying the information you provide to your VPS. It allows you to act as if you were sitting in front of a keyboard at your hosts site. The latest version of Putty can be downloaded here:

<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

Ubuntu Commands

adduser : creates a new user profile

usermod -aG sudo pushinode : gives user profile (in this case, 'pushinode') admin power

exit : closes current ssh session

sudo apt-get update : updates the list of packages and versions listed by the apt repositories

sudo apt-get upgrade : upgrades any packages that have a new version reported by apt-get update

sudo apt install -y nano git : installs nano, a text editor, and git, a tool for downloading github repositories

wget : downloads from the specified target



crontab -e : edits the crontab file. Cronjobs are automated request that you setup. In this instance we ask the crontab to tell the sentinel command every 60 seconds.

LINKS

PushiCoinTeam-02/24/2018

Announce: <https://bitcointalk.org/index.php?topic=2888147.0>

Web: <http://pushiplay.pw/>

Masternode Guide: <http://www.pushiplay.pw/PUSHI-MN-GUIDE-EN.pdf>

Sentinel config: <https://github.com/pushiplay/sentinel/blob/master/README.md>

Masternode Stats: <https://pushi.overemo.com/masternodes>

Mining Pool Stats: <https://pushi.overemo.com/pools>

Github: <https://github.com/pushiplay>

Explorer: <http://pushiplay.pw:8080/chain/Pushi>

Twitter: <https://twitter.com/PushiCoinTeam>

Discord: [https://discord.gg/Qd7YEK4\(edited\)](https://discord.gg/Qd7YEK4(edited))

Mining pools:

<https://hash4.life/>

<https://pool.ionik.fr/>

<https://www.cryptorushmining.com/>

<https://eu2.multipool.es/>

