Quick Start Guide

Edit New Page

Jump to bottom

Michael Polzer edited this page on 29 May 2018 · 107 revisions

First things First

Before you start working with this guide: Do not create issue reports if you fail to get it working! This is most often caused by a setup mistake, over-reading a configuration or system step or simply not being into Linux, Apache and MySQL enough. But don't fret! MPOS has a very active community, so if you are struggling to get things to work, try our IRC Channel FIRST! New issue reports regarding the setup or running of MPOS will always be closed by commenting to join IRC for further assistance: https://webchat.freenode.net/?channels=#mpos

Now, onwards! Make sure to always try fixing things yourself before asking others to hold your hands!

Disclaimer!

Before following this guide this warning is given as advice and a word of warning. Running a live pool is a job which requires in depth knowledge of pool code as well as the ability to debug and fix the pool. By NO means is this an in depth guide to running a pool and as such should only be used for private/testnet pools! As running a pool is a long and tiresome process where many different things can affect the stability and usability of the pool, it is guaranteed that problems will occur. If help is required it can be obtained in the MPOS channel however a person must be able to show that they have tried to resolve a problem themselves. Otherwise help will not be given as the running of the pool in itself is a difficult process

Description

This guide will help you get your MPOS interface setup including all dependencies and software required. Please use this as a guide on how get things to work but keep in mind that things change and maybe this Wiki page is outdated and things do not work anymore as expected.

If you intend to follow this quick start guide please ensure you know what you are doing. Basic concepts in Linux like compiling and installing software, configuring Apache and other services should be a no-brainer to you.

This guide is **NOT** taking care of security or any special setups to make this installation secure. You should consider:

- adding a special DB user just for `MPOS`
- ensures you are not using root which could break things
- create a new virtual host and point the DocumentRoot towards the public folder
- This way the cronjobs folder can be locked down and not be available to the website itself
- run all 3rd party services as non-root
- if one of them has a flaw your root account is not compromised

Website Footer

When you decide to use `MPOS` please be so kind and leave the footer intact. You are not the author of the software and should honor those that have worked on it. I don't mind changing the LTC donation address at the bottom, but keep in mind who really wrote this software and would deserve those ;-).

OS Setup

My base system used in this guide is **Ubuntu Server 16.10 LTS**. During setup I kept most defaults and installed the OpenSSH (for easier access) and LAMP Server groups during software selection. Other software and tools required will be mentioned in this guide.

Update your server:

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

Run these two commands till there are no more updates. Remember to run these commands again any time you install dependencies to make sure you have the latest versions.

In order to get started install some basic tools. Other tools will be installed when needed.

```
sudo apt-get install git
sudo apt-get install build-essential libcurl4-openssl-dev libdb5.3-dev libdb5.3++-
dev mysql-server
```

I have organized all the dependencies you will need here at the beginning of the guide, so that they may be put out of the way right away.

For NOMP stratum

Install Redis Server following these instructions.

```
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
sudo apt-get install -y nodejs
```

For python-stratum

```
sudo apt-get install python-twisted python-mysqldb python-dev python-setuptools
python-memcache python-simplejson python-pylibmc
sudo easy_install -U distribute
```

For MPOS:

```
sudo apt-get install memcached php-memcached php7.0-mysqlnd php7.0-curl php7.0-json
php7.0-curl libapache2-mod-php7.0
sudo apt-get install php-mbstring php-dom
sudo apache2ctl -k stop; sleep 2; sudo apache2ctl -k start
```

Litecoind

```
sudo apt-get install libtool autotools-dev automake pkg-config libssl-dev libevent-
dev bsdmainutils python3
sudo apt-get install libboost-system-dev libboost-filesystem-dev libboost-chrono-dev
libboost-program-options-dev libboost-test-dev libboost-thread-dev
# Alternative for boost:
sudo apt-get install libboost-all-dev
```

Because of Bitcoin, the crypto world just loves to use BerkeleyDB 4.8 - it is available from the archive

```
wget http://download.oracle.com/berkeley-db/db-4.8.30.zip
unzip db-4.8.30.zip
cd db-4.8.30
cd build_unix/
../dist/configure --prefix=/usr/local --enable-cxx
make
make install
cd ..
```

Note: Litecoin is only used as an example, as it should guarantee a working MPOS, please, replace litecoind with the coin of your choice.

Download

You can grab the latest version from Litecoin @ Github:

```
cd ~
git clone git://github.com/litecoin-project/litecoin.git
```

Compile

With that all set up we can compile and install:

```
# Change to litecoin folder
cd ~
cd litecoin
./autogen.sh
./configure --with-incompatible-bdb
make
make check
make install
```

Test Network

Again, this is not necessary for a real MPOS pool. Testing MPOS on the testnet of your coin is suggested

Now that we have a working executable we can fetch the testnet for Litecoin from Github:

```
cd ~
git clone git://github.com/xrobau/litecoin-testnet-box.git
cd litecoin-testnet-box
```

The included .conf in the /1 folder will not function properly if the username and password remain the same, so you will have to edit the file so that one of them does not match the other.

```
vi 1/litecoin.conf
```

Now you can start your server.

```
make start
```

If everything went well you should now get some information from the RPCs, this data can vary depending on version numbers used and changes to the net since the first access:

```
root@ubuntu-server:~/litecoin-testnet-box# make getinfo
litecoind -datadir=1  getinfo
{
    "version" : 60300,
    "protocolversion" : 60001,
    "walletversion" : 60000,
    "balance" : 0.00000000,
    "blocks" : 0,
    "connections" : 1,
    "proxy" : "",
    "difficulty" : 0.00024414,
    "testnet" : true,
    "keypoololdest" : 1369211361,
    "keypoolsize" : 101,
    "paytxfee" : 0.00000000,
    "mininput" : 0.00010000,
    "errors" : ""
}
litecoind -datadir=2  getinfo
{
    "version" : 60300,
    "protocolversion" : 60001,
    "walletversion" : 60000,
    "balance" : 0.00000000,
    "blocks" : 0,
    "connections" : 1,
    "proxy" : "",
```

```
    "difficulty" : 0.00024414,
    "testnet" : true,
    "keypoololdest" : 1369211361,
    "keypoolsize" : 101,
    "paytxfee" : 0.00000000,
    "mininput" : 0.00010000,
    "errors" : ""
}
```

Success! Litecoin is now setup and working properly running on a test network. No actual blocks are calculated here but this is enough for testing purposes later.

You can fetch your wallet address via litecoind inside your testnet checkout from the testnet folder:

```
litecoind -datadir=1 getaccountaddress ""
```

Remember this address, you'll need it later.

To mine on this server with a remote graphic card on your network you need to determine your machines IP.

```
ifconfig
```

Now point your miner, I will be using cgminer.

```
cgminer.exe --scrypt -o http://yourip:19334 -u -p
```

You should now be mining!

Stratum Mining Server Software

There exists many stratum mining server software alternatives which you have to choose one from and configure MPOS to work with;

Stratum Mining

Stratum Mining is a software which supports all PoW and PoS coins including Litecoin, Bitcoin, PPcoin and Novacoin.This is rather easy to implement so here a quick start guide if you wish to try it out. `MPOS` does support stratum, For VARDIFF support MPOS must be set to match the initial pool target in the `stratum-mining` config file.

NOMP

A new stratum has been created, named NOMP, if you would like to use this instead of stratum-mining, please follow this link. Either one will work. NOMP supports x11, quark, scrypt, sha, and more. NOMP is written in node.js

CoiniumServ

Yet another stratum server alternative called CoiniumServ exists which can run on any platform (Windows, Linux and MacOS) and supports many algorithms. Check project's documentation wiki for setup & configuration guides.

Requirements

First we need to install some packages required to run `stratum-mining` :

```
cd ~
sudo apt-get install python-twisted python-mysqldb python-dev python-setuptools
python-memcache python-simplejson python-pylibmc
sudo easy_install -U distribute
```

Downloading

We need to fetch `stratum-mining` and some additional code for a `stratum` implementation:

```
git clone https://github.com/Tydus/litecoin_scrypt.git
git clone https://github.com/ahmedbodi/stratum-mining.git
git clone https://github.com/ahmedbodi/stratum.git
```

That covers the download. Lets go ahead and prepare the software!

Installation

We need to install `litecoin_scrypt` and `stratum` :

```
cd stratum-mining
git submodule init
git submodule update

cd externals/litecoin_scrypt
sudo python setup.py install
```

```
sudo python setup.py install
```

```
cd ~
cd stratum-mining/externals/stratum
sudo python setup.py install
```

Configuration

Now that we have everything installed we can configure `stratum-mining` to run with our testnet:

```
cd ~
cp stratum-mining/conf/config_sample.py stratum-mining/conf/config.py
vi stratum-mining/conf/config.py
```

You will need to adjust some settings for this to work:

```
CENTRAL_WALLET = 'Your_Valid_Bitcoin_or_Litecoin_Address'
[...]
COINDAEMON_TRUSTED_HOST = 'localhost'
COINDAEMON_TRUSTED_PORT = 19334
COINDAEMON_TRUSTED_USER = 'testnet'
COINDAEMON_TRUSTED_PASSWORD = 'testnet'
COINDAEMON_ALGO = 'scrypt'
COINDAEMON_Reward = 'POW'
COINDAEMON_SHA256_TX = 'no
[...]
HOSTNAME = 'yourservername'
[...]
DATABASE_DRIVER = 'mysql'
DB_MYSQL_HOST = 'localhost'
DB_MYSQL_DBNAME = 'mpos'
DB_MYSQL_USER = 'root'
DB_MYSQL_PASS = 'root'
[...]
POOL_TARGET = 16
[...]
SOLUTION_BLOCK_HASH = True
```

Starting stratum-mining

This is the easy part, but don't do it till you set up the database in the next steps. Change to the `stratum-mining` folder and fire up the service:

```
cd stratum-mining
twistd -ny launcher.tac
```

If you want to run it in the background you can remove the `-ny` and replace it with `-y` :

```
twistd -y launcher.tac
```

Special Notes

When running stratum-mining I noticed that stratum and pushpoold use different settings. @pooler was nice enough to explain it to me in detail:

`pushpoold` uses a target bits terminology and stratum a difficulty setting. These are different. When running `pushpoold` at a target bit of 20 you will match the default setting of 16 in `stratum-mining` . This will ensure that hashrates on `MPOS` match up! If you'd think you could set `pushpoold` to 16 and match it with stratum you will be off.

He devised a formula that can be used to change `stratum` difficulty and match `pushpoold` and `MPOS` to it:

```
(stratum diff) ~= 2^((target bits in pushpool) - 16)
```

Add The Result To The Stratum Pool_Target

MPOS

Requirements

We need to install some additional software to ensure `MPOS` will work as expected. This is a sample for Debian based systems and PHP 5, please adjust your packages if you are using a more recent PHP version:

```
sudo apt-get install memcached php5-memcached php5-mysqlnd php5-curl php5-json
libapache2-mod-php5 php5-xml
sudo apache2ctl -k stop; sleep 2; sudo apache2ctl -k start
```

Note: PHP7 will require an additional package called `php-dom` for Composer to work. You may also receive a warning that `zip` is missing, the run `apt-get install php-zip zip` as well.

Also, please download and check out the security warnings and errors triggered by phpsecinfo and fix those before attempting to run MPOS:
http://phpsec.org/projects/phpsecinfo/

Just download the ZIP archive, unzip into your pools webroot and run the `index.php`. Do not continue until everything shows green! The only exception is Maximum post size which needs to be a bit bigger so MPOS can store admin panel settings to the DB when clicking save.

Download

Fetch the latest version of `MPOS`:

```
# We move into the default webroot of Ubuntu
cd /var/www
sudo git clone git://github.com/MPOS/php-mpos.git MPOS
cd MPOS
sudo git checkout master
php composer.phar install
```

Mailserver Setup

To allow your site to send messages you need to set up the mail server.

```
sudo apt-get install postfix
```

Accept 'internet server' and keep the defaults. This will set up Postfix.

Database Setup

During server installation MySQL was installed by using the `LAMP Server` group. This also setup a password for your `root` user. Use this password now to create the database and import the structure:

```
# Create database
```

```
sudo mysql -p -e "create database mpos"

# Import structure
sudo mysql -p mpos < sql/000_base_structure.sql
```

Configuration

Folder Permissions

First, give proper permissions to our compiled templates and caching folder. This example shows the procedure in *Ubuntu*, other distributions may vary (*apache* in *CentOS*, *httpd* in others). Ensure to give the service user access that will be executing the scripts!

```
sudo chown -R www-data templates/compile templates/cache logs
```

Main Configuration

We need to configure the project to work on our newly installed server:

```
sudo cp include/config/global.inc.dist.php include/config/global.inc.php
```

Now edit the configuration file and adjust the settings to match your database and litecoin RPC Client:

```
$config['db']['host'] = 'localhost';
$config['db']['user'] = 'root';
$config['db']['pass'] = 'root';
$config['db']['port'] = 3306;
$config['db']['name'] = 'mpos';
[...]
$config['wallet']['type'] = 'http';
$config['wallet']['host'] = 'localhost:19334';
$config['wallet']['username'] = 'testnet';
$config['wallet']['password'] = 'testnet';
[...]
$config['gettingstarted']['stratumurl'] = 'localhost';
```

You will also need to setup a proper, random (!) `$config['SALT']` and another `$config['SALTY']` secret! If you forgot that, you will get an error message after installing MPOS: `You absolutely SHOULD NOT leave your SALT or SALTY default changing them will require registering again` . SALT and SALTY must be a minimum of 24 characters or you will get an error message: 'SALT or SALTY is too short, they should be more than 24

characters and changing them will require registering again'
One final edit should be to edit: /usr/local/lib/python2.7/dist-packages/stratum-0.2.13-py2.7.egg/stratum/websocket_transport.py

Change:

```
from autobahn.websocket import WebSocketServerProtocol, WebSocketServerFactory
```

To:

```
from autobahn.twisted.websocket import WebSocketServerProtocol,
WebSocketServerFactory
```

Now everything is setup and we should be able to test things.

Testing

MPOS

Open your browser and head over to `MPOS` on your server `http:///MPOS/public

You should be greeted with the homepage. Create a new account on the registration page (the first account will have admin rights and does not require a mail confirmation) and add a new worker!

Test the Pool

Now with everything in place and running you can fire up a miner of your choice and test if shares are committed:

```
minerd -o stratum+tcp://<yourserverIP>:3333 -O YourAccount.worker:WorkerPassword -t
1
```

If all went well you should see this:

```
minerd -o stratum+tcp://localhost:3333 -O DummyUser.test:test -t 1

[2013-05-22 11:01:20] 1 miner threads started, using 'scrypt' algorithm
```

```
[2013-05-22 11:01:20] 1 miner threads started, using 'scrypt' algorithm.

[...]

[2013-05-22 11:01:21] thread 0: 4104 hashes, 6.38 khash/s

[2013-05-22 11:01:34] thread 0: 137208 hashes, 10.85 khash/s

[2013-05-22 11:01:34] accepted: 1/1 (100.00%), 10.85 khash/s (yay!!!)

[2013-05-22 11:01:34] thread 0: 288 hashes, 10.66 khash/s
```

Please test the same with `cgminer` or a similar tool supporting stratum on port `3333`, the default port opened by the configuration in `stratum-mining`.

Now with shares being committed and new blocks being generated in our test network head to the webinterface and your stats should be updated!

Cronjobs

`MPOS` comes with a few cronjobs that are part of the backend. Please set the up according to regular cron guides. `MPOS` comes with a wrapper script that will run all crons for you. It is called `run-crons.sh` and can be found in the cronjobs folder. Adjust this script to suit your needs, then add it to your cron service.

For testing purposes you can also run them from command line:

```
cd /var/www/MPOS/cronjobs

./run-crons.sh
```

You will not receive any output. Please check in `cronjobs/logs` for the appropriate logfiles. For more detailed logs you can turn on debugging in the `cronjobs/shared.inc.php` file.

For a small pool, this will work fine. If you are planning to run a medium sized pool in the future or expect a lot of shares, please read: cronjob advanced setup.

E-Mail

You are required to have a proper mail setup running on your box. Keep in mind that large pools can have quite excessive e-mails being sent over all. This is by design and increases security while also notifying uses about state changes in the pool. For smaller pools, running via an external provider will suffice. For larger once, you should look into a contract for Mail Relays.

A simple guide to get started with Gmail and Postfix can be found here: https://rtcamp.com/tutorials/linux/ubuntu-postfix-gmail-smtp/

Basic DoS Protection

You are required to secure your server, Using internal and external means. Pasting the below rules into your SSH console one line at a time from top to bottom would be a good start, But there are other considerations as well.

This assumes you will be using port 53 for DNS and you don't want UDP for anything else, And your webserver is serving on port 80, This doesn't restrict services such as SSH or FTP. You want to limit the rate your clients can connect at the door, and not rely solely on a cache to do it.

```
sudo iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m limit --limit
50/minute --limit-burst 200 -j ACCEPT

sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -m limit --limit
50/second --limit-burst 50 -j ACCEPT

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG
NONE -j DROP

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN -j DROP

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j DROP

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags FIN,RST FIN,RST -j DROP

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP

sudo iptables -A INPUT -i eth0 -p tcp -m tcp --tcp-flags ACK,URG URG -j DROP

sudo iptables -A PORT_SCANNING -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --
limit 1/s -j RETURN

sudo iptables -A PORT-SCANNING j DROP
```

```
sudo iptables -A INPUT -p icmp -m limit --limit 2/second --limit-burst 2 -j ACCEPT

sudo iptables -A INPUT -p tcp --tcp-flags ALL FIN,PSH,URG -j DROP

sudo iptables -A INPUT -p udp --sport 53 -j ACCEPT

sudo iptables -A INPUT -p udp --dport 53 -j ACCEPT

sudo iptables -A OUTPUT -p udp --sport 53 -j ACCEPT

sudo iptables -A OUTPUT -p udp --dport 53 -j ACCEPT

sudo iptables -A INPUT -p udp -j DROP

sudo iptables -A OUTPUT -p udp -j DROP
```

You can find a more detailed explanation of what these do: HERE

Conclusion

Congratulations! You have now a running instance of `MPOS` to test with! Please consider donating if you like my work and find this guide helpful!

Quick Start

Follow these instructions to perform this entire setup, minus the file configurations, in a quick and efficient bundle.

```
cd ~

sudo apt-get update

sudo apt-get dist-upgrade

sudo apt-get install git

sudo apt-get install build-essential libboost-all-dev libcurl4-openssl-dev libdb5.1-
dev libdb5.1++-dev mysql-server

git clone git://github.com/litecoin-project/litecoin.git

cd litecoin/src

make -f makefile.unix USE_UPNP=-
```

```
sudo cp litecoind /usr/bin

cd ~

git clone git://github.com/xrobau/litecoin-testnet-box.git

cd litecoin-testnet-box

cd ~

sudo apt-get install python-twisted python-mysqldb python-dev python-setuptools
python-memcache python-simplejson

easy_install -U distribute

git clone https://github.com/Tydus/litecoin_scrypt.git

git clone https://github.com/ahmedbodi/stratum-mining.git

git clone https://github.com/ahmedbodi/stratum.git

cd stratum-mining

git submodule init

git submodule update

cd externals/litecoin_scrypt

sudo python setup.py install

cd ~

cd stratum-mining/externals/stratum

sudo python setup.py install

cd ~

cp stratum-mining/conf/config_sample.py stratum-mining/conf/config.py

sudo apt-get install memcached php5-memcached php5-mysqlnd php5-curl

sudo apache2ctl -k stop; sleep 2; sudo apache2ctl -k start

cd ..

cd ..

cd var/www
```

```
sudo git clone git://github.com/MPOS/php-mpos.git MPOS

cd MPOS

php composer.phar install

sudo git checkout master

mysql -u -p -e "create database mpos"

mysql -u -p mpos < sql/000_base_structure.sql

sudo chown -R www-data templates/compile templates/cache

sudo cp include/config/global.inc.dist.php include/config/global.inc.php
```

Helpful Hints

Make sure you allow for 2 Gb of RAM in memory settings, if you leave it at the default 512 Mb some compiling commands won't be able to execute properly.

Or you can create a pagefile if more RAM is not available by using these commands.

```
sudo dd if=/dev/zero of=/swapfile bs=1024 count=2048k

sudo mkswap /swapfile

sudo chown root:root /swapfile

sudo chmod 0600 /swapfile

sudo swapon /swapfile
```

This is temporary unless you add the file to your fstab.

```
sudo vi /etc/fstab

Add the following line to the end of the file

/swapfile swap swap defaults 0 0
```

Remember to set VMware to bridge your adapter so that your server will show up on the local network.

The server can be really touchy when trying to get a miner to connect to it. Keep trying, restart, fiddle with the .conf settings, etc, eventually it will begin working properly, though I had to create the server 8 times before it worked.

If easy_install doesn't work then run these additional lines first.

```
cd ~

sudo rm /usr/bin/easy_install*

sudo rm /usr/local/bin/easy_install*

curl -O http://python-distribute.org/distribute_setup.py

sudo python distribute_setup.py

sudo rm distribute_setup.py
```

This will compile and install two libraries that are required to run `stratum-mining`. If running the git commands does not pull the git subfiles the first time, try deleting the stratum-mining folder and downloading it again.

To remove the slightly annoying "Could not reliably determine server name" warning message when restarting apache, just insert this line of code into the /etc/apache2/httpd.conf file:

```
ServerName yourservername
```  ikikeee

- ◦ installation of Redis requires TCL 8.5 or newer TCL Download Page

+ Add a custom sidebar

### Clone this wiki locally

```
https://github.com/MPOS/php-mpos.wiki.git
```