Blockstack Mining Zero to Testnet Style

This demo will cover mining on the Stacks 2.0 Testnet (Argon Phase) using Virtualbox, Ubuntu, and a shell script.

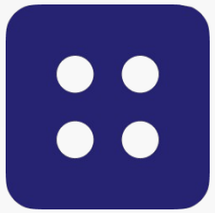
Inspiration

- To create a simplified and unified method of installing and updating the miner
- To emulate similar install processes found with nvm, rust, and other tools used
- To allow for an easily accessible and isolated method for running the miner

Resources

- Virtualbox Website and Downloads
  (supports Windows / Mac / Linux)
- Ubuntu Website and Downloads
  (free and open source operating system)
- Blockstack Testnet Website and Status
- Jason’s Zero to Testnet Blog Series and GitHub repository for the mining script
- The magic one-liner command for setup:

```
curl -o- https://raw.githubusercontent.com/AbsorbingChaos/bks-setup-miner/master/config-miner-argon.sh | bash
```



1.
Setting up Virtualbox

The first part of this demo will cover the configuration of Virtualbox, outlined in Step 0 of the Zero to Testnet series.

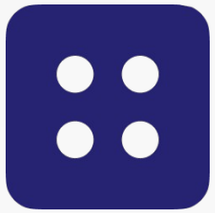
Why this software?

- **VirtualBox** is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. This program allows you to run an entire "virtualized computer" on your machine. This means that whether you are running Windows, Mac, or Linux (the "Host OS"), you are able to run a second computer with a different operating system (the "Guest OS"). At the time of writing this post, the version used was VirtualBox 6.1.6.

- **Ubuntu Server** is an open source server operating system with low memory requirements, a command-line interface by default (no graphical user interface or "GUI"), and a wide range of support for software packages. Using Ubuntu Server in combination with Virtualbox allows us to set up a Blockstack Node with as little as 1gb RAM! At the time of writing this post, the long-term supported version used was Ubuntu Server 20.04 LTS.

The setup process

- Download the necessary files
- Install Virtualbox on the system
- Create a new virtual machine in Virtualbox
  - Add the Ubuntu ISO file for installation at boot
  - Change Networking from NAT to Bridged Adapter
- Boot the virtual machine and install Ubuntu
- Reboot to remove the Ubuntu ISO
- Test logging into the new virtual machine!



2.
Setting up the Miner

The second part of this demo will cover running the one-liner script that sets up, updates, and runs the miner.

Running the script

- The script and the configuration file are both hosted in the same GitHub repository.

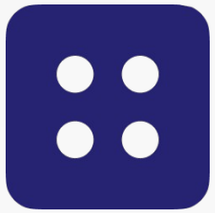
- Once Ubuntu is setup, the next step is to run the script using the command below:

```
curl -o- https://raw.githubusercontent.com/AbsorbingChaos/bks-setup-miner/master/config-miner-argon.sh | bash
```

- The script will provide output at each step.

What the script does

1) Install or update operating system prerequisites
2) Detect or install node version manager (nvm)
3) Detect or install Node.js (via nvm)
4) Detect or install Rust (via rustup.rs)
5) Download or update the stacks-blockchain repository (via git)
6) Detect or create the keychain file (via blockstack-cli make_keychain)
7) Detect or request test BTC balance (via keychain file and faucet)
8) Detect or download miner configuration file (via GitHub)
9) If the miner configuration file is downloaded, automatically inserts the private key (via keychain file)
10) Check test BTC balance before starting the miner, loop until found
11) Start the miner



3.
Tips and Tricks

The third part of this demo will cover common screens, helpful commands, and additional technical resources.

Common screens

- Initial startup / UTXOs found – *the miner started!*

```
INFO [1594924698.860] [testnet/stacks-node/src/run_loop/neon.rs:86] [ThreadId(1)] Miner node: checki
ng UTXOs at address: mhZy2d7TumrN9wXmpciTDwSxtA7RDoT4j2
INFO [1594924699.499] [testnet/stacks-node/src/run_loop/neon.rs:94] [ThreadId(1)] Miner node: starti
ng up, UTXOs found.
```

- Fetching burnchain blocks – *syncing started!*

```
INFO [1594924768.476] [src/burnchains/burnchain.rs:855] [ThreadId(1)] Node will fetch burnchain bloc
ks 0-11547...
```

- Leader key invalid hash – *normal during sync!*

```
WARN [1594925114.978] [src/chainstate/burn/operations/leader_key_register.rs:243] [ThreadId(5)] Inva
lid leader key registration: invalid consensus hash 1463c95a4a8bcd0d8a3d4c29077f4ec9243db551
WARN [1594925114.980] [src/burnchains/burnchain.rs:475] [ThreadId(5)] REJECTED(1866) leader key regi
ster 3487e0eca612a1ac1f0f1210f3b71676366b6c5972d353a747d85f55209823d2 at 1866,2: LeaderKeyBadConsens
usHash
```

Common screens

- Did not win sortition – *try again next time!*

```
WARN [1594929656.051] [testnet/stacks-node/src/neon_node.rs:496] [ThreadId(6)] Did not win sortition
, my blocks [burn_hash= 0a6a59420e790ec4f8293c15c5be559fd6122f5779dfcc49263e978f626af17a, block_hash
= 3062dfe30c9f0242f4e541fc37787c0431fd8b7ce33a8a250a80f8312ceb7fab], their blocks [par_burn_hash= 0a
6a59420e790ec4f8293c15c5be559fd6122f5779dfcc49263e978f626af17a, burn_hash= 6373d39c2dea21a2f755a1597
db0c7125a0fadfc0d95de4bbe2421524a5104c4, block_hash =22b098232ca24c8b4d65253874c590474309aeabdfc74a5
5fc7be862c654239e]
```

- Won sortition – *happy dance!*

```
INFO [1594930482.862] [testnet/stacks-node/src/neon_node.rs:430] [ThreadId(6)] Won sortition! stacks
_header=204c47ca8728a95dcaf9bee3f271b5ed04befa832d9d122cbc54c195d7c0f01c, burn_header=515efdbd893f1d
261009373f98a3d61431fe520ebfb6ff8b60133fe47f43465c
```

- Processing a smart contract

```
INFO [1594929662.232] [src/chainstate/stacks/db/transactions.rs:555] [ThreadId(6)] Contract-call to
ST2SW87CAR3BYO2VXGX5FV517NGX2Y9FQE26WACVA.hello_world.ClarityName("set-value") args [Buffer(666f6f),
 Buffer(626172)] returned Response(ResponseData { committed: true, data: UInt(1) })
INFO [1594929662.238] [src/chainstate/stacks/db/transactions.rs:556] [ThreadId(6)] Contract-call to
ST2SW87CAR3BYO2VXGX5FV517NGX2Y9FQE26WACVA.hello_world.ClarityName("set-value") args [Buffer(666f6f),
 Buffer(626172)] cost ExecutionCost { write_length: 107, write_count: 1, read_length: 1573, read_cou
nt: 2, runtime: 1780 }
```

Helpful commands

- Create a keychain file:

```
npx blockstack-cli@1.1.0-beta.1 make_keychain -t
```

- Check test BTC balance:

```
curl https://sidecar.staging.blockstack.xyz/sidecar/v1/faucets/btc/BTC_ADDRESS
```

- Request test BTC from the faucet:

```
curl https://sidecar.staging.blockstack.xyz/sidecar/v1/faucets/btc\?address\
=BTC_ADDRESS
```

Technical resources

- Proof of Transfer (PoX) Whitepaper
- Stacks Improvement Proposals (SIPs) on GitHub
  - Proof of Burn (PoB):
    SIP-001 Burn Election
  - Proof of Transfer (PoX):
    SIP-007 Stacking Consensus
- Slides on PoX from a discussion with NEAR
- Joe Bender’s NYC Virtual STX Mining Meetup

Reward calculations

Snippet from Jude in Discord:

- As a miner, you are rewarded as follows:
  - you get all of your coinbase and anchored block transaction fees
  - you get 40% of the transaction fees of the microblocks you append to your anchored block
  - you get 60% of the transaction fees of the microblocks you build upon when you produce your anchored block
- The rewards are locked up for a maturity window (currently 100 blocks, but subject to change)

Questions or Comments?