CAS: CRYPTO APPLICATION SERVER

Monero

Installation & Configuration

May 2019

Please report any errors in this documentation to tech support.
We feel that giving you the proper tools is essential to enabling our customers to use this product quickly and efficiently.

Suggestions? Do tell!

For best results, review the contents of this guide *before* configuration.

Before you begin:

The following example is based on the following presumptions:

- Your CAS is installed and functioning.
- Your server is running Ubuntu 16.04 LTS.
- Your server is online.

The server successfully used for this guide (meet or exceed these specs):

- 2 vCPU
- 80 GB SSD storage
- 4 GB of memory (RAM)

The instructions in this guide requires sudo (root) access.

The current blockchain size can be quickly determined at the following URL:

```
https://bitinfocharts.com/monero/
```

This is a step-by-step guide for adding a Monero node as a Hot Wallet Buy source. This guide was tested using the Linux 64-bit v0.14.0.2 of the Monero Core wallet. This is only an example and not to be construed as an endorsement for any particular exchange or wallet.

Monero (monerod) is the actual node; it talks to the other nodes on the Monero network and shares the blockchain. Once setup, you simply leave the node to run, and it becomes part of the Monero backbone. The node communicates with you via two similar interfaces:

- monero-wallet-cli: this is how users control the Monero node from the command line (CLI). While there are other ways to affect the node, the CLI client is the most graceful and far less prone to cause data disruption. It is predictable. Predictable = good. This client is typically rarely used after the wallet is initially setup. It is also used to shut down the monerod node.
- monero-wallet-rpc: this is the RPC server that CAS uses to communicate with the Monero node. After your wallet has been created (using monero-wallet-cli), you will use monero-wallet-rpc as the “go-between”. CAS sends and receives data to it, then monero-wallet-rpc relays CAS’ info to the monerod node itself.

The following guide is written with instructions to be executed as root. This is discouraged, but simplicity is the goal in mind - not security. You should modify these procedures to safeguard your assets as needed. You will be a target for hackers, and your Monero will be at risk. You are strongly encouraged to implement any and all restrictions available to you to protect your server from unauthorized access. These enhancements however, are beyond the scope of this guide.

Download and extract the Monero installation file:

Start as root at the terminal prompt.

Select the Monero package - either 32 bit or 64 bit:
At the CLI, you can easily determine if your server will support 64-bit. Most VPS' only offer 64-bit versions of Ubuntu these days. If you are unsure, then at your terminal, type:

```
lscpu | grep '64-bit'
```

- It should report something resembling: "CPU op-mode(s): 32-bit, 64-bit"
- Use the modern 64-bit whenever possible.
- If the command above comes back empty, you *must* use the 32-bit version.

Navigate to your terminal's root home directory:

```
cd ~
```

Download the installation file to root's HOME directory:
Choose one method to use:

- From the CLI terminal, enter (for 64-bit):

  ```
  wget https://downloads.getmonero.org/cli/linux64 -O monerod.tar.bz2
  ```

  For the 32-bit version use the CLI:

  ```
  wget https://downloads.getmonero.org/cli/linux32 -O monerod.tar.bz2
  ```

- From your GUI browser:
  - Navigate to: https://getmonero.org/downloads/
  - Scroll down to "Linux".
  - Locate the link to the latest (Command-Line Tools Only) software for either 32 or 64 bit. The 32/64 decision is determined in the previous step.
  - Click on the appropriate link and save the file.
  - Copy the file to your VPS' /root/ directory

Extract Monero:

```
tar jxvf monerod.tar.bz2
```

Navigate into the program directory (the directory will change as the version changes):

```
cd ~/monero-v0.14.0.2
```

Start the monerod daemon (background process):

The Monero blockchain will begin downloading during this step. This may take hours or even days. In testing, the typical blockchain sync time on a dual-CPU SSD system is about 12 hours. The time depends upon your hardware and throughput. It can be interrupted as needed, but you cannot proceed until the blockchain has been ENTIRELY downloaded & synchronized.

A configuration file is optional. Merely start the daemon as follows:

```
./monerod --detach
```

- You should see something resembling:

```
2019-04-22 18:01:22.213      7f1611ffa780   INFO  globalsrc/daemon/main.cpp:287
Monero 'Boron Butterfly' (v0.14.0.2-release)
Forking to background...
```

You can interrupt monerod using the following command. The node *should* pickup where it left off. This is ill-advised, but may become necessary. Restart the node exactly the same way as it started above.

```
cd ~/monero-v0.14.0.2
./monerod exit
```

- The daemon may take a few minutes to completely shut down, be patient.
- Use the “top” command to watch for complete shutdown.

The Monero node blockchain has been completely synchronized when it reports:

```
 You are now synchronized with the network. You may now start monero-wallet-cli
```

Create a wallet:

After the Monero daemon downloads and synchronizes the blockchain, it will prompt you to create a wallet. You cannot use CAS with the Monero node until this is done. You also cannot create a wallet until the node is completely synchronized. Be patient; it *will* happen eventually.

After the daemon has synchronized your node, it will report:
“You are now synchronized with the network. You may now start monero-wallet-cli"
The CLI wallet interface (monero-wallet-cli) is used to create a Monero wallet. Once a wallet is created, you’re unlikely to use this program again. It’s sole function (in the CAS context) is to create a hot wallet for your node.

Create your wallet; type:

```
cd ~/monero-v0.14.0.2
./monero-wallet-cli
```

You will see something similar to the following:

```
This is the command line monero wallet. It needs to connect to a monero
daemon to work correctly.
```

```
Monero 'Boron Butterfly' (v0.14.0.2-release)
Logging to ./monero-wallet-cli.log
Specify wallet file name (e.g., MyWallet). If the wallet doesn't exist, it will
be created.
Wallet file name (or Ctrl-C to quit):
```

Enter a wallet file name (example provided):

```
Wallet file name (or Ctrl-C to quit): CASWallet
No wallet found with that name. Confirm creation of new wallet named: CASWallet
(Y/Yes/N/No): Y
```

Type in the password that will protect your Monero wallet:

```
Generating new wallet...
Enter a new password for the wallet:
Confirm password:
```

- Make it strong and unique! You won't have to type it frequently.
- Save the wallet name & password!
    - You'll use them in the Monero RPC configuration file (created next).

Select your language of choice.

```
List of available languages for your wallet's seed:
If your display freezes, exit blind with ^C, then run again with
--use-english-language-names
0 : Deutsch
1 : English
2 : Español
3 : Français
4 : Italiano
5 : Nederlands
6 : Português
7 : русский язык
8 : 日本語
9 : 简体中文 (中国)
10 : Esperanto
11 : Lojban
Enter the number corresponding to the language of your choice: 1
```

The language selected is used to generate the password recovery seeds. It has nothing to do with the interface; it merely determines what your "seed" will look like. The seed can be used to recover your wallet in the event of data loss (your wallet file is destroyed). Protect your funds; write down the seed values or save them someplace safe and private!

The next lines you'll see are your wallet details. The wallet address is listed first, followed by the key that signs all your outgoing Monero transactions:

```
Generated new wallet:
47HJo2wJFVkBrEp1eRSwkkPQteTPqXj6NT9RB58eYuCsB4pUBoME8LS63TiJWanw4wH7WB1s8e4QCg4
s8QwxbQP9FSaqYWx
View key: 7e156d6940fbb85a95e1cbe994a22230b99d2d09592998f74dac684ed2b2e100
```

- The wallet address is where you **RECEIVE** Monero.

The password recovery seeds are then listed last:

```
NOTE: the following 25 words can be used to recover access to your wallet.
Write them down and store them somewhere safe and secure. Please do not store
them in your email or on file storage services outside of your immediate
control.

nerves onward wives atrium dilute bomb pizza gang
whipped poverty vats humid amply pigment semifinal eleven
exult soothe seismic ocean software return napkin nodes nodes
```

- Save the seeds somewhere safe and private. Protect your funds!

The wallet will normally begin to sync automatically, but if you desire to manually ensure that it's happening, at the (wallet-cli) prompt type:

```
refresh
```

The wallet will now begin to synchronize with the Monero network:

```
Starting refresh...
Refresh done, blocks received: 1192
Untagged accounts:
      Account            Balance     Unlocked balance               Label
 *    0 47HJo2    0.000000000000    0.000000000000    Primary account
-------------------------------------------------------------------------------
      Total       0.000000000000    0.000000000000
Currently selected account: [0] Primary account
Tag: (No tag assigned)
Balance: 0.000000000000, unlocked balance: 0.000000000000
Background refresh thread started
```

- Once the wallet has caught up to the network, you may exit the wallet client.
- This sync may take quite a while (days), and you cannot proceed to the next steps until it has completed. Specifically, the RPC client WILL NOT work until the sync is finished!

The wallet client now waits patiently for further instructions (i.e. exit, status, refresh):

```
[wallet 47HJo2 (out of sync)]:
```

- Type "status" to see the current state of synchronization.

After it's synchronized, you can exit the Monero wallet client:

```
[wallet 47HJo2]: exit
```

You can only use one wallet with CAS, so you're unlikely to use this program again for creating another wallet (for our purposes).

The RPC server/client:

The monero-wallet-rpc program is the part of the Monero node that CAS actually "talks" to. This RPC program in turn communicates with the Monero node daemon:

CAS ← → monero-wallet-rpc ← → monerod

"monero-wallet-rpc" is both a server and a client.

In this step, we define the ports that the monero RPC server will expose to CAS, and also tell the RPC server about the wallet that it will permit to be controlled by CAS. We can do this on the command line, or more securely through a configuration file. When we use a configuration file, all we need to tell monero-wallet-rpc is where to find that file. If permissions are properly restricted, then your wallet password will be secure.

CAS uses an RPC port to connect to the monero-wallet-rpc program. It also will need a username and password. This combination will be different from your wallet username/password. For the purposes of this example, we'll assign port 18088 to the RPC server for CAS to connect with it.

The RPC server will communicate directly with the daemon through yet *another* RPC port. The daemon normally exposes port 18081 for use by it's own wallet clients. Unless the daemon RPC port is changed by you, you may safely allow the RPC server (monero-wallet-rpc) to use the defaults.

So to clarify this example, the whole chain of communication will ultimately look like this:

CAS ←<RPC port 18088> → monero-wallet-rpc ←<RPC port 18081> → monerod

Setting up the CAS settings is done at the same time you configure the RPC server.

Once you have started the RPC client/server, you normally leave it running. If you decide to shut down the program, the "proper" way to exit is from another terminal (and all on one line):

```
curl  -u CAS:Complicated --digest \
      -X POST http://127.0.0.1:18088/json_rpc    \
      -d '{"jsonrpc":"2.0","id":"0","method":"stop_wallet"}' \
      -H 'Content-Type: application/json'
```

- Some of those settings are determined in the next step.

Create the RPC server configuration file:

Create and open the file in nano:

```
nano ~/monero-v0.14.0.2/monero-rpc.conf
```

Add your wallet settings:

```
wallet-file=/root/monero-v0.14.0.2/CASWallet
password=ComplexPW
```

Add your CAS settings:

```
rpc-bind-ip=127.0.0.1
rpc-bind-port=18088
rpc-login=CAS:Complicated
```

- Rpc-bind-ip is always 127.0.0.1 if CAS is running on the same server (typical).
- Rpc-bind-port may be any unused port. 18088 is used in this example.
- Rpc-login is the arbitrary username/password you create for CAS. It doesn't have to do anything special or specific, but don't make it too complicated until you've tested it all out. **Whatever you set here will also be used in the "parameters" of the CAS Crypto Setting.**

Exit nano & save the configuration file - press ctrl-X and accept all defaults.

Start the monero-wallet-rpc program using the configuration file:

```
cd ~/monero-v0.14.0.2
./monero-wallet-rpc --config-file ~/monero-v0.14.0.2/monero-rpc.conf &
```

Both the node daemon and monero-wallet-rpc ***must*** remain running for CAS to use the local Monero wallet.

Assemble the required information for CAS:

User: this is the first half of the "rpc-login" you set earlier. For this example we created "CAS".
Password: second part of the "rpc-login" set earlier. We made the password "Complicated".
Host: from rpc-bind-ip - normally "127.0.0.1"
Port: from rpc-bind-port - we set it to "18088"

Next, add a new crypto-currency type to CAS:

1. Navigate to the "Crypto Settings" menu in the left-hand column.
2. Click on the "+ ADD" button.
3. Create a suitable description, i.e. "Monero Local Node".
4. Select "XMR" for the required "Crypto Currency*" setting.
5. Set "Configuration Cash Currency*" to "USD" (or your preference).
6. Set "Buy Rate Source" to whatever source you prefer.
7. Set "Minimum Cash Amount Per Transaction*" to zero – this enables *any* sale.
8. Select your Hot Wallet Buy source to "Monerod RPC".
9. Enter the "Parameters" for the Hot Wallet Buy source as determined in the previous step.
    - ***Example: "CAS:Complicated:127.0.0.1:18088"***
10. Click on "SUBMIT" to save your settings.

Finally, test it out:

- Select "Test Hot Wallet Buy" from the RUN XMR SETTINGS TEST.
- Presuming everything was entered correctly, you should receive confirmation of success!

- You should also now configure the "Hot Wallet Sell" if you implement it.

UNINSTALLING MONERO CORE:

Uninstallation is essentially the reverse order of installation. The following procedures will also delete any unposted transactions and is completely irreversible. Any Monero in your local wallet will be lost **forever** *unless* you've saved your seeds. Again:

THIS WILL DELETE ALL PENDING TRANSACTIONS IN YOUR LOCAL WALLET!!

THIS PROCEDURE MAY RESULT IN PERMANENT DATA AND/OR Monero LOSS!!

Change your Crypto settings in CAS to reflect your replacement hot wallet.

- In CAS, navigate to "Crypto Settings" and update all settings that involve Monero.

Login to your terminal.

Shut down the monerod daemon and RPC server:

```
cd ~/monero-v0.14.0.2
./monerod exit
```

- The RPC server should quickly lose the connection to the daemon and shut itself down.
- Use "top" to watch for the process to exit gracefully.

Remove any startup references in crontab:

```
crontab -e
```

You may also need to configure your firewall to block inbound connections to port 18081.

That's it - Monero has been stopped on your server, however the data files are left behind. This final step deletes all traces of Monero on your server!!

The Monero files are located in a few different places (by default).

To PERMANENTLY delete the local blockchain:

```
rm -rf ~/.bitmonero/
```

To PERMANENTLY delete the node software, including wallet clients:

```
rm -rf ~/.monero-v0.14.0.2/
```

To PERMANENTLY delete your wallets, you must know your wallet file name, then:

```
rm -rf ~/walletname
```

- This is dangerous. Be careful what you type!

TROUBLESHOOTING Monero CORE:

You may discover that running a full node taxes your system dramatically. The following suggestions are offered to assist you in this regard. These pitfalls have nothing to do with CAS, and are offered merely as a courtesy to our valued customers.

You are strongly encouraged to:

- Run your Monero node with as much RAM as practical.
- Use a SSD drive instead of a legacy hard disk.

Excessive CPU Usage:

The Monero daemon (monerod) will consume as much CPU as possible. This will cause other programs on your system to run slowly, even to the point of triggering a violation of your VPS' TOS (Terms of Service) / AUP (Acceptable Use Policy). If you receive a complaint from your VPS, turn it down a few notches using "cpulimit".

1) Install the cpulimit program:

```
sudo apt update && sudo apt install cpulimit
```

2) Now determine the PID of the monerod process:

```
pgrep -f monerod
```

- The number returned is the desired PID.

3) Finally tell cpulimit to restrict that process' cpu load:

```
cpulimit -b -l 50 -p 1234
```

- In this example, the PID is 1234 and we reduced the process' CPU load to 50%

Play with the load % until your system is running within the proper guidelines.

Excessive RAM Usage:

The Monero daemon (monerod) will consume as much memory as you have available. This will cause other programs on your system to start slowly, sometimes VERY slowly - as monerod releases RAM to the system. It is a RAM glutton, however a very well-behaved RAM glutton.

To restrict the amount of RAM that the system permits monerod, try ulimit:

```
ulimit -d sizeKiB
```

- Set the sizeKiB somewhere between 1000 and your actual free RAM.
- Type "free" at the CLI to view your available memory (and used).

Excessive Disk I/O (page faults)

After monerod has eaten all your system's RAM, your OS will begin relying upon disk paging. This will result in potentially enormous amounts of "thrashing"; copying data back and forth between a hard drive and RAM. If you use a VPS, this will raise red flags.

VPS hosts have suggested SSD drives for those nodes.

Once the blockchain has syncronized, the thrashing should stop (or slow significantly). Your VPS may shut down your account before then though, and you'll have to start from scratch. One possible solution is to download the huge (35gb) blockchain file and import it.

See this link for more information on that:

```
https://www.monero.how/tutorial-how-to-speed-up-initial-blockchain-sync
```

Keep your server secure:

Your exposure as a Monero node will draw some unwanted attention. You may become the victim of targeted attacks. Some suggested ways to mitigate this:

- Change your SSH port – attempts to hack into your server become greatly reduced.
- Implement fail2ban – it will reduce the effects of attacks against your server.