travis Documentation

Release latest

May 27, 2019

Contents

This is the official documentation for the CyberMiles blockchain. It walks you through key features of the smart contract platform, including validator and governance related transactions as well as the Lity programming language and virtual machine.

CHAPTER 1

Contents

1.1 Getting Started

In this document, we will discuss how to create and run a single node CyberMiles blockchain on your computer. It allows you to connect and test basic features such as coin transactions, staking and unstaking for validators, governance, and smart contracts.

1.1.1 Use Docker

The easiest way to get started is to use our pre-build Docker images. Please make sure that you have Docker installed and that your Docker can work without sudo.

Initialize

Let's initialize a docker image for the Travis build first.

```
docker run --rm -v ~/volumes/local:/travis cybermiles/travis node init --home /travis
```

The node's data directory is `~/volumes/local` on the local computer.

Run

Now you can start the CyberMiles Travis node in docker.

```
docker run --privileged --name travis -v ~/volumes/local:/travis -t -p 26657:26657 -p␣
↪8545:8545 cybermiles/travis node start --home /travis
```

At this point, you can Ctrl-C to exit to the terminal and travis will remain running in the background. You can check the CyberMiles Travis node's logs at anytime via the following docker command.

```
docker logs -f travis
```

You should see blocks like the following in the log.

```
INFO [07-14|07:23:05] Imported new chain segment               blocks=1 txs=0 mgas=0.
↪000 elapsed=431.085µs mgasps=0.000 number=163 hash=05e16c...a06228
INFO [07-14|07:23:15] Imported new chain segment               blocks=1 txs=0 mgas=0.
↪000 elapsed=461.465µs mgasps=0.000 number=164 hash=933b97...0c340c
```

Connect

You can connect to the local CyberMiles node by attaching an instance of the Travis client.

```
# Get the IP address of the travis node
docker inspect -f '{{ .NetworkSettings.IPAddress }}' travis
172.17.0.2

# Use the IP address from above to connect
docker run --rm -it cybermiles/travis attach http://172.17.0.2:8545
```

It opens the web3-cmt JavaScript console to interact with the virtual machine. The example below shows how to unlock the coinbase account so that you have coins to spend.

```
Welcome to the Travis JavaScript console!

instance: vm/v1.6.7-stable/linux-amd64/go1.9.3
coinbase: 0x7eff122b94897ea5b0e2a9abf47b86337fafebdc
at block: 231 (Sat, 14 Jul 2018 07:34:25 UTC)
 datadir: /travis
 modules: admin:1.0 cmt:1.0 eth:1.0 net:1.0 personal:1.0 rpc:1.0 web3:1.0

> personal.unlockAccount('0x7eff122b94897ea5b0e2a9abf47b86337fafebdc', '1234')
true
>
```

1.1.2 Build from source

Currently, we only support source builds for CentOS 7 and Ubuntu 16.04 linux distributions.

Prerequisite

You must have GO language version 1.10+ installed in order to build and run a Travis node. The easiest way to get GO 1.10 is through the GVM. Below are the commands on a Linux server.

```
$ bash < <(curl -s -S -L https://raw.githubusercontent.com/moovweb/gvm/master/
↪binscripts/gvm-installer)
$ vim ~/.bash_profile
inset into the bash profile: source "$HOME/.bashrc"
log out and log in
$ sudo apt-get install bison
$ gvm version
output should look like: Go Version Manager v1.0.22 installed at /home/myuser/.gvm
```

(continues on next page)

(continued from previous page)

```
$ gvm install go1.10.3
$ gvm use go1.10.3 --default
```

Build

First we need to checkout the correct branch of Travis from Github:

```
go get github.com/CyberMiles/travis (ignore if an error occur)
cd $GOPATH/src/github.com/CyberMiles/travis
git checkout master
```

Next, we need to build libENI and put it into the default Travis data directory `~/.travis/`.

```
sudo rm -rf ~/.travis
wget -O $HOME/libeni.tgz https://github.com/CyberMiles/libeni/releases/download/v1.3.
↪4/libeni-1.3.4_ubuntu-16.04.tgz
tar zxvf $HOME/libeni.tgz -C $HOME
mkdir -p $HOME/.travis/eni
cp -r $HOME/libeni-1.3.4/lib $HOME/.travis/eni/lib
```

Currently libENI can only run on Ubuntu 16.04 and CentOS 7. If your operating system is CentOS, please change the downloading url. You can find it here: https://github.com/CyberMiles/libeni/releases

Now, we can build and install Travis binary. It will populate additional configuration files into `~/.travis/`

```
cd $GOPATH/src/github.com/CyberMiles/travis
make all
```

If the system cannot find glide at the last step, make sure that you have `$GOPATH/bin` under the `$PATH` variable.

Run

Let's start a Travis node locally using the `~/.travis/` data directory.

```
travis node init
travis node start
```

Connect

You can connect to the local CyberMiles node by attaching an instance of the Travis client.

```
travis attach http://localhost:8545
```

It opens the web3-cmt JavaScript console to interact with the virtual machine. The example below shows how to unlock the coinbase account so that you have coins to spend.

```
Welcome to the Travis JavaScript console!

instance: vm/v1.6.7-stable/linux-amd64/go1.9.3
coinbase: 0x7eff122b94897ea5b0e2a9abf47b86337fafebdc
at block: 231 (Sat, 14 Jul 2018 07:34:25 UTC)
 datadir: /travis
```

(continues on next page)

(continued from previous page)

```
 modules: admin:1.0 cmt:1.0 eth:1.0 net:1.0 personal:1.0 rpc:1.0 web3:1.0

> personal.unlockAccount('0x7eff122b94897ea5b0e2a9abf47b86337fafebdc', '1234')
true
>
```

1.1.3 Test transactions

You can now send a transaction between accounts like the following.

```
personal.unlockAccount("from_address")
cmt.sendTransaction({"from": "from_address", "to": "to_address", "value": web3.
↪toWei(0.001, "cmt")})
```

Next, you can paste the following script into the Travis client console, at the > prompt.

```
function checkAllBalances() {
  var totalBal = 0;
  for (var acctNum in cmt.accounts) {
      var acct = cmt.accounts[acctNum];
      var acctBal = web3.fromWei(cmt.getBalance(acct), "cmt");
      totalBal += parseFloat(acctBal);
      console.log("  cmt.accounts[" + acctNum + "]: \t" + acct + " \tbalance: " +␣
↪acctBal + " CMT");
  }
  console.log("  Total balance: " + totalBal + "CMT");
};
```

You can now run the script in the console, and see the results.

```
> checkAllBalances();
cmt.accounts[0]:      0x6.....................................230      balance: 466.
↪798526 CMT
cmt.accounts[1]:      0x6.....................................244      balance: 1531␣
↪CMT
Total balance: 1997.798526CMT
```

1.2 Deploy a TestNet Node

In this document, we will discuss how to start your own node and connect to the CyberMiles Travis TestNet. While we highly recommend you to run your own Travis node, you could still directly access RPC services from a node provided by the CyberMiles Foundation at `https://testnet-rpc.cybermiles.io:8545`.

1.2.1 Snapshot

The easiest and fastest way to start a node is to use a snapshot. It is also recommended for most people. You can run the node inside a Docker container or on Ubuntu 16.04 / CentOS 7 servers.

Option 1: Docker from a snapshot

Prerequisite

Please setup docker.

Docker Image

Docker image for Travis is stored on Docker Hub. The current version of the TestNet environment is always using the 'vTestnet' release which can be pulled as follows.

```
docker pull cybermiles/travis:vTestnet
```

Note: Configuration and data will be stored at /travis directory in the container. The directory will also be exposed as a volume. The ports 8545, 26656 and 26657 will be exposed for connection.

Getting Travis TestNet Config

```
rm -rf $HOME/.travis && mkdir -p $HOME/.travis/config
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪genesis.json > $HOME/.travis/config/genesis.json
```

Download snapshot

Get a list of recent snapshots of the testnet from AWS S3 travis-ss-testnet

You can splice the file name from the bucket list. The downloading url will be like `https://s3-us-west-2.amazonaws.com/travis-ss-testnet/testnet/travis_ss_testnet_1542623121_254975.tar`. You must have found that the file name contains timestamp and block number at which the snapshot is made.

```
wget $(curl -s http://s3-us-west-2.amazonaws.com/travis-ss-testnet/latest.html)
```

Extract the file and copy the `data` and `vm` subdirectories from the uncompressed directory to `$HOME/.travis`

Start the Node and Join Travis TestNet

Change your name from default name `local`.

```
vim ~/.travis/config/config.toml
# here you can change your name
moniker = "<your_custom_name>"
```

Run the docker Travis application:

```
docker run --privileged --name travis -v $HOME/.travis:/travis -t -p 26657:26657␣
↪cybermiles/travis:vTestnet node start --home /travis
```

Attach to the Node and run web3-cmt.js

In another terminal window, log into the Docker container and then run the `travis` client and attach to the node. It will open a console to run `web3-cmt.js` commands.

```
docker exec -it travis bash
> ./travis attach http://localhost:8545
```

Option 2: Binary from a snapshot

Make sure your os is Ubuntu 16.04 or CentOS 7

Download snapshot

Get a list of recent snapshots of the testnet from AWS S3 travis-ss-testnet

You can splice the file name from the bucket list. The downloading url will be like `https://s3-us-west-2.amazonaws.com/travis-ss-testnet/testnet/travis_ss_testnet_1542623121_254975.tar`. You must have found that the file name contains timestamp and block number at which the snapshot is made.

```
rm -rf $HOME/.travis

mkdir -p $HOME/release
cd $HOME/release
SNAPSHOT_URL=$(curl -s http://s3-us-west-2.amazonaws.com/travis-ss-testnet/latest.
↪html)
wget $SNAPSHOT_URL
TAR_FILE="${SNAPSHOT_URL##*/}"
tar xf $TAR_FILE

# if your os is Ubuntu 16.04
mv .travis $HOME
wget https://github.com/CyberMiles/travis/releases/download/vTestnet/travis_vTestnet_
↪ubuntu-16.04.zip
unzip travis_vTestnet_ubuntu-16.04.zip
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib

# or if your os is CentOS 7
mv .travis $HOME
wget https://github.com/CyberMiles/travis/releases/download/vTestnet/travis_vTestnet_
↪centos-7.zip
unzip travis_vTestnet_centos-7.zip
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib
```

Set env variables for eni lib

```
# for convenience, you should also put these two lines in your .bashrc or .zshrc
export ENI_LIBRARY_PATH=$HOME/.travis/eni/lib
export LD_LIBRARY_PATH=$HOME/.travis/eni/lib
```

Start the Node and Join Travis TestNet

Download the testnet config and change your name from default name `local`.

```
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪config.toml > $HOME/.travis/config/config.toml
vim ~/.travis/config/config.toml
# here you can change your name
moniker = "<your_custom_name>"
```

Start the application

```
cd $HOME/release
./travis node start --home $HOME/.travis
```

Attach to the Node and Run web3-cmt.js

In another terminal window, run the `travis` client and attach to the node. It will open a console to run `web3-cmt.js` commands.

```
cd $HOME/release
./travis attach http://localhost:8545
```

1.2.2 Test transactions

In this section, we will use the `travis` client's web3-cmt JavaScript console to send some transactions and verify that the system is set up properly. You can't test transactions untill you are completely in sync with the TestNet. It might take hours to sync.

Create and fund a test account

Once you attach the `travis` to the node as above, create two accounts on the TestNet.

```
Welcome to the Travis JavaScript console!
> personal.newAccount()
...
```

Now you have created TWO accounts `0x1234FROM` and `0x1234DEST` on the Travis TestNet. It is time to get some test CMTs. Please go visit the website below, and ask for 1000 TestNet CMTs for account `0x1234FROM`. We will also send 1000 TEST tokens, issued by the TEST smart contract, to the account.

http://travis-faucet.cybermiles.io

Test transactions

You can test transactions between your two accounts. Remember to unlock both of your accounts.

```
> personal.unlockAccount("0x1234FROM","password")
true
...
> cmt.sendTransaction({from:"0x1234FROM", to:"0x1234DEST",value:1000})
```

(continues on next page)

(continued from previous page)

```
...
> cmt.getBalance("0x1234DEST")
...
```

You can also test smart contract transactions for the TEST token as below.

```
> abi = [{"constant":true,"inputs":[],"name":"name","outputs":[{"name":"","type":
↪"string"}],"payable":false,"stateMutability":"view","type":"function"},{"constant
↪":false,"inputs":[{"name":"_spender","type":"address"},{"name":"_value","type":
↪"uint256"}],"name":"approve","outputs":[{"name":"","type":"bool"}],"payable":false,
↪"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs":[],"name
↪":"totalSupply","outputs":[{"name":"","type":"uint256"}],"payable":false,
↪"stateMutability":"view","type":"function"},{"constant":false,"inputs":[{"name":"_
↪from","type":"address"},{"name":"_to","type":"address"},{"name":"_value","type":
↪"uint256"}],"name":"transferFrom","outputs":[{"name":"","type":"bool"}],"payable
↪":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,"inputs
↪":[],"name":"INITIAL_SUPPLY","outputs":[{"name":"","type":"uint256"}],"payable
↪":false,"stateMutability":"view","type":"function"},{"constant":true,"inputs":[],
↪"name":"decimals","outputs":[{"name":"","type":"uint256"}],"payable":false,
↪"stateMutability":"view","type":"function"},{"constant":false,"inputs":[],"name":
↪"unpause","outputs":[],"payable":false,"stateMutability":"nonpayable","type":
↪"function"},{"constant":true,"inputs":[],"name":"paused","outputs":[{"name":"","type
↪":"bool"}],"payable":false,"stateMutability":"view","type":"function"},{"constant
↪":false,"inputs":[{"name":"_spender","type":"address"},{"name":"_subtractedValue",
↪"type":"uint256"}],"name":"decreaseApproval","outputs":[{"name":"success","type":
↪"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"},{
↪"constant":true,"inputs":[{"name":"_owner","type":"address"}],"name":"balanceOf",
↪"outputs":[{"name":"balance","type":"uint256"}],"payable":false,"stateMutability":
↪"view","type":"function"},{"constant":false,"inputs":[],"name":"pause","outputs":[],
↪"payable":false,"stateMutability":"nonpayable","type":"function"},{"constant":true,
↪"inputs":[],"name":"owner","outputs":[{"name":"","type":"address"}],"payable":false,
↪"stateMutability":"view","type":"function"},{"constant":true,"inputs":[],"name":
↪"symbol","outputs":[{"name":"","type":"string"}],"payable":false,"stateMutability":
↪"view","type":"function"},{"constant":false,"inputs":[{"name":"_to","type":"address
↪"},{"name":"_value","type":"uint256"}],"name":"transfer","outputs":[{"name":"","type
↪":"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"},{
↪"constant":false,"inputs":[{"name":"_spender","type":"address"},{"name":"_addedValue
↪","type":"uint256"}],"name":"increaseApproval","outputs":[{"name":"success","type":
↪"bool"}],"payable":false,"stateMutability":"nonpayable","type":"function"},{
↪"constant":true,"inputs":[{"name":"_owner","type":"address"},{"name":"_spender",
↪"type":"address"}],"name":"allowance","outputs":[{"name":"","type":"uint256"}],
↪"payable":false,"stateMutability":"view","type":"function"},{"constant":false,
↪"inputs":[{"name":"newOwner","type":"address"}],"name":"transferOwnership","outputs
↪":[],"payable":false,"stateMutability":"nonpayable","type":"function"},{"inputs":[],
↪"payable":false,"stateMutability":"nonpayable","type":"constructor"},{"anonymous
↪":false,"inputs":[],"name":"Pause","type":"event"},{"anonymous":false,"inputs":[],
↪"name":"Unpause","type":"event"},{"anonymous":false,"inputs":[{"indexed":true,"name
↪":"previousOwner","type":"address"},{"indexed":true,"name":"newOwner","type":
↪"address"}],"name":"OwnershipTransferred","type":"event"},{"anonymous":false,"inputs
↪":[{"indexed":true,"name":"owner","type":"address"},{"indexed":true,"name":"spender
↪","type":"address"},{"indexed":false,"name":"value","type":"uint256"}],"name":
↪"Approval","type":"event"},{"anonymous":false,"inputs":[{"indexed":true,"name":"from
↪","type":"address"},{"indexed":true,"name":"to","type":"address"},{"indexed":false,
↪"name":"value","type":"uint256"}],"name":"Transfer","type":"event"}]
> tokenContract = web3.cmt.contract(abi)
> tokenInstance = tokenContract.at("0xb6b29ef90120bec597939e0eda6b8a9164f75deb")
> tokenInstance.transfer.sendTransaction("0x1234DEST", 1000, {from: "0x1234FROM"})
```

After 10 seconds, you can check the balance of the receiving account as follows.

```
> tokenInstance.balanceOf.call("0x1234DEST")
```

Fee free transactions

On CyberMiles blockchain, we have made most transactions (except for heavy users or spammers) fee-free. You can try it like this in `travis` client console.

```
> cmt.sendTransaction({from:"0x1234FROM", to:"0x1234DEST",value:1000,gasPrice:0})
...
```

To try a fee-free smart contract-based token transaction, use the following in the `travis` client console.

```
> tokenInstance.transfer.sendTransaction("0x1234DEST", 1000, {from: "0x1234FROM",␣
↪gasPrice: 0})
```

1.2.3 Sync from Genesis

Experts Only: This section is not recommend not necessary for most people. But it is important that we can always start the CyberMiles blockchain from genesis to prove its correctness.

You can always start a new CyberMiles blockchain node from genesis, and sync it all the way to the current block height. The process is fairly involved since it requires you to upgrade and restart the node at certain block heights.

One of the key characteristics of the CyberMiles blockchain is the finality of each block. The blockchain will never fork. It will only produce a new block when 2/3 of the validator voting power reach consensus. Software upgrade on the CyberMiles blockchain is done via consensus. That is, at an agreed upon block height, all nodes must upgrade to a new version of the software to continue. Any node that does not upgrade will not reach consensus with the rest of the blockchain and stop.

The table below shows the software version and their corresponding block heights on the testnet.

Blocks	Software version	Notes
0 - 224550	0.1.2-beta	
224551 - 1083930	0.1.3-beta-hotfix1	
1083931 - 1190700	0.1.4-beta	
1190701 - 1248300	0.1.5-beta	
1248301 - 1306100	0.1.6-beta-testnet	
1306101 - 1653200	0.1.7-beta-testnet	
1653201 -	0.1.8-beta	Crashed at 1694600,fixed w/o changing version #

The general process for syncing a node from genesis is as follows:

- The 0.1.2-beta software starts from genesis
- It automatically stops at block 224550
- You will download 0.1.3-beta-hotfix1 software, and restart the node
- The process repeats until the block height is current

In the instructions below, we will explain how to sync a Linux binary node and a Docker node from genesis.

Option 3 (the hard way): Binary from genesis

Make sure your os is Ubuntu 16.04 or CentOS 7

Download pre-built binaries

Get software version 0.1.2-beta from from release page

```
mkdir -p $HOME/release
cd $HOME/release

# if your os is Ubuntu
wget https://github.com/CyberMiles/travis/releases/download/v0.1.2-beta/travis_v0.1.2-
↪beta_ubuntu-16.04.zip
unzip travis_v0.1.2-beta_ubuntu-16.04.zip

# or if your os is CentOS
wget https://github.com/CyberMiles/travis/releases/download/v0.1.2-beta/travis_v0.1.2-
↪beta_centos-7.zip
unzip travis_v0.1.2-beta_centos-7.zip
```

Getting Travis TestNet Config

```
rm -rf $HOME/.travis
cd $HOME/release
./travis node init --env testnet
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪genesis.json > $HOME/.travis/config/genesis.json
```

Change your name from default name `local`

```
cd $HOME/.travis
vim $HOME/.travis/config/config.toml

# here you can change your name
moniker = "<your_custom_name>"
```

Copy libeni into the default Travis data directory

```
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib

# set env variables for eni lib
# for convenience, you should also put these two lines in your .bashrc or .zshrc
export ENI_LIBRARY_PATH=$HOME/.travis/eni/lib
export LD_LIBRARY_PATH=$HOME/.travis/eni/lib
```

Start the Node and Join Travis TestNet

```
cd $HOME/release
./travis node start
```

Upgrade and Continue

At certain block heights, the node will stop. Download the next version of the software (e.g., `0.1.3-beta-hotfix1` at block height 224550), and restart.

```
rm -rf $HOME/release
mkdir -p $HOME/release
cd $HOME/release

# if your os is Ubuntu
wget https://github.com/CyberMiles/travis/releases/download/v0.1.3-beta-hotfix1/
↪travis_v0.1.3-beta-hotfix1_ubuntu-16.04.zip
unzip travis_v0.1.3-beta-hotfix1_ubuntu-16.04.zip

# or if your os is CentOS
wget https://github.com/CyberMiles/travis/releases/download/v0.1.3-beta-hotfix1/
↪travis_v0.1.3-beta-hotfix1_centos-7.zip
unzip travis_v0.1.3-beta-hotfix1_centos-7.zip

./travis node start
```

Option 4 (the hard way): Docker from genesis

Prerequisite

Please setup docker.

Docker Image

Docker image for Travis is stored on Docker Hub. Genesis starts from software version `0.1.2-beta`

```
docker pull cybermiles/travis:v0.1.2-beta
```

Note: Configuration and data will be stored at `/travis` directory in the container. The directory will also be exposed as a volume. The ports 8545, 26656 and 26657 will be exposed for connection.

Getting Travis TestNet Config

```
rm -rf $HOME/.travis
docker run --rm -v $HOME/.travis:/travis -t cybermiles/travis:v0.1.2-beta node init --
↪env testnet --home /travis
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init/config/
↪genesis.json > $HOME/.travis/config/genesis.json
```

(continues on next page)

(continued from previous page)

Start the Node and Join Travis TestNet

First change your name from default name `local`

```
vim ~/.travis/config/config.toml

# here you can change your name
moniker = "<your_custom_name>"
```

Run the docker Travis application:

```
docker run --privileged --name travis -v $HOME/.travis:/travis -p 26657:26657 -t␣
↪cybermiles/travis:v0.1.2-beta node start --home /travis
```

Upgrade and Continue

At certain block heights, the node will stop. Download the next version of the software (e.g., `0.1.3-beta-hotfix1` at block height 224550), and restart.

```
docker stop travis
docker rm travis

docker pull cybermiles/travis:v0.1.3-beta-hotfix1
docker run --privileged --name travis -v $HOME/.travis:/travis -p 26657:26657 -t␣
↪cybermiles/travis:v0.1.3-beta-hotfix1 node start --home /travis
```

1.3 Deploy a MainNet Node

In this document, we will discuss how to start your own node and connect to the CyberMiles MainNet. While we highly recommend you to run your own Travis node, you could still directly access RPC services from a node provided by the CyberMiles Foundation at `https://rpc.cybermiles.io:8545`.

1.3.1 Snapshot

The easiest and fastest way to start a node is to use a snapshot. You can run the node inside a Docker container or on Ubuntu 16.04 / CentOS 7 servers.

Option 1: Docker from snapshot

Prerequisite

Please setup docker.

Docker Image

Docker image for Travis is stored on Docker Hub. MainNet environment is currently at the 'v0.1.8-beta-hotfix' release which can be pulled as follows.

```
docker pull cybermiles/travis:v0.1.8-beta-hotfix
```

Note: Configuration and data will be stored at `/travis` directory in the container. The directory will also be exposed as a volume. The ports 8545, 26656 and 26657 will be exposed for connection.

Getting Travis MainNet Config

```
rm -rf $HOME/.travis
docker run --rm -v $HOME/.travis:/travis -t cybermiles/travis:v0.1.8-beta-hotfix node␣
↪init --env mainnet --home /travis
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪genesis.json > $HOME/.travis/config/genesis.json
```

Download snapshot

Get a list of recent snapshots of the mainnet from AWS S3 travis-ss-bucket

You can splice the file name from the bucket list. The downloading url will be like `https://s3-us-west-2.amazonaws.com/travis-ss-bucket/mainnet/travis_ss_mainnet_1558862782_1724747.tar`. You must have found that the file name contains timestamp and block number at which the snapshot is made.

```
wget $(curl -s http://s3-us-west-2.amazonaws.com/travis-ss-bucket/latest.html)
```

Extract the file and copy the `data` and `vm` subdirectories from the uncompressed directory to `$HOME/.travis`

Start the Node and Join Travis MainNet

Change your name from default name `local`.

```
vim $HOME/.travis/config/config.toml
# here you can change your name
moniker = "<your_custom_name>"
```

For the security concern, the rpc service is disabled by default, you can enable it by changing the `config.toml`:

```
vim $HOME/.travis/config/config.toml
rpc = true
```

Run the docker Travis application:

```
docker run --privileged --name travis -v $HOME/.travis:/travis -t -p 26657:26657␣
↪cybermiles/travis:v0.1.8-beta-hotfix node start --home /travis
```

Attach to the Node and run web3-cmt.js

In another terminal window, log into the Docker container and then run the `travis` client and attach to the node. It will open a console to run `web3-cmt.js` commands.

```
docker exec -it travis bash
> ./travis attach http://localhost:8545
```

Option 2: Binary from snapshot

Make sure your os is Ubuntu 16.04 or CentOS 7

Download snapshot

Get a list of recent snapshots of the mainnet from AWS S3 travis-ss-bucket

You can splice the file name from the bucket list. The downloading url will be like `https://s3-us-west-2.amazonaws.com/travis-ss-bucket/mainnet/travis_ss_mainnet_1558862782_1724747.tar`. You must have found that the file name contains timestamp and block number at which the snapshot is made.

```
rm -rf $HOME/.travis

mkdir -p $HOME/release
cd $HOME/release
SNAPSHOT_URL=$(curl -s http://s3-us-west-2.amazonaws.com/travis-ss-bucket/latest.html)
wget $SNAPSHOT_URL
TAR_FILE="${SNAPSHOT_URL##*/}"
tar xf $TAR_FILE

# if your os is Ubuntu 16.04
mv .travis $HOME
wget https://github.com/CyberMiles/travis/releases/download/v0.1.8-beta-hotfix/travis_
↪v0.1.8-beta-hotfix_ubuntu-16.04.zip
unzip travis_v0.1.8-beta-hotfix_ubuntu-16.04.zip
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib

# or if your os is CentOS 7
mv .travis $HOME
wget https://github.com/CyberMiles/travis/releases/download/v0.1.8-beta-hotfix/travis_
↪v0.1.8-beta-hotfix_centos-7.zip
unzip travis_v0.1.8-beta-hotfix_centos-7.zip
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib
```

Set env variables for eni lib

```
# for convenience, you should also put these two lines in your .bashrc or .zshrc
export ENI_LIBRARY_PATH=$HOME/.travis/eni/lib
export LD_LIBRARY_PATH=$HOME/.travis/eni/lib
```

Start the Node and Join MainNet

Download the mainnet config and change your name from default name `local`.

```
mkdir -p $HOME/.travis/config
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪config.toml > $HOME/.travis/config/config.toml
vim $HOME/.travis/config/config.toml
# here you can change your name
moniker = "<your_custom_name>"
```

For the security concern, the rpc service is disabled by default, you can enable it by changing the `config.toml`:

```
vim $HOME/.travis/config/config.toml
rpc = true
```

Start the application

```
cd $HOME/release
./travis node start --home $HOME/.travis
```

Attach to the Node and Run web3-cmt.js

In another terminal window, run the `travis` client and attach to the node. It will open a console to run `web3-cmt.js` commands.

```
cd $HOME/release
./travis attach http://localhost:8545
```

1.3.2 Sync from Genesis

Expert Only: You can always start a new CyberMiles blockchain node from genesis, and sync it all the way to the current block height. The process is fairly involved since it requires you to upgrade and restart the node at certain block heights.

One of the key characteristics of the CyberMiles blockchain is the finality of each block. The blockchain will never fork. It will only produce a new block when 2/3 of the validator voting power reach consensus. Software upgrade on the CyberMiles blockchain is done via consensus. That is, at an agreed upon block height, all nodes must upgrade to a new version of the software to continue. Any node that does not upgrade will not reach consensus with the rest of the blockchain and stop.

The table below shows the software version and their corresponding block heights on the mainnet.

Blocks	Software version	Note
0 - 230767	v0.1.2-beta	The chain stops itself at 230767
230768 - 386223	v0.1.3-beta-hotfix1	
386224 - 386245	v0.1.3-beta-hotfix2	Mannualy stop the chain within this height range and deploy hotfix2
286246 - 1321175	v0.1.3-beta-hotfix2	The chain stops itself at 1321175
1321176 - 1700580	v0.1.7-beta	The chain stops itself at 1700580
1700581 -	v0.1.8-beta-hotfix	v0.1.8-beta crashes at 1724748

The general process for syncing a node from genesis is as follows:

- The 0.1.2-beta software starts from genesis
- It automatically stops at block 230767
- You will download 0.1.3-beta-hotfix1 software, and restart the node
- The process repeats until the block height is current

In the instructions below, we will explain how to switch from one version of the software to the next at specific block heights.

Option 3 (the hard way): Binary from Genesis

Make sure your os is Ubuntu 16.04 or CentOS 7

Download pre-built binaries

Get software version `0.1.2-beta` from from release page

```
mkdir -p $HOME/release
cd $HOME/release

# if your os is Ubuntu
wget https://github.com/CyberMiles/travis/releases/download/v0.1.2-beta/travis_v0.1.2-
↪beta_ubuntu-16.04.zip
unzip travis_v0.1.2-beta_ubuntu-16.04.zip

# or if your os is CentOS
wget https://github.com/CyberMiles/travis/releases/download/v0.1.2-beta/travis_v0.1.2-
↪beta_centos-7.zip
unzip travis_v0.1.2-beta_centos-7.zip
```

Getting Travis MainNet Config

```
rm -rf $HOME/.travis
cd $HOME/release
./travis node init --env mainnet
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪genesis.json > $HOME/.travis/config/genesis.json
```

Change your name from default name `local`.

```
cd $HOME/.travis
vim $HOME/.travis/config/config.toml

# here you can change your name
moniker = "<your_custom_name>"
```

Copy libeni into the default Travis data directory

```
mkdir -p $HOME/.travis/eni
cp -r $HOME/release/lib/. $HOME/.travis/eni/lib

# set env variables for eni lib
# for convenience, you should also put these two lines in your .bashrc or .zshrc
export ENI_LIBRARY_PATH=$HOME/.travis/eni/lib
export LD_LIBRARY_PATH=$HOME/.travis/eni/lib
```

Start the Node and Join Travis MainNet

```
cd $HOME/release
./travis node start
```

Upgrade and Continue

At certain block heights, the node will stop. Download the next version of the software (e.g., `0.1.3-beta-hotfix1` at block height 230767), and restart. A notable exception is the switch between `0.1.3-beta-hotfix1` and `0.1.3-beta-hotfix2` – that has to happen manually within a specific range of block heights.

```
rm -rf $HOME/release
mkdir -p $HOME/release
cd $HOME/release

# if your os is Ubuntu
wget https://github.com/CyberMiles/travis/releases/download/v0.1.3-beta-hotfix1/
↪travis_v0.1.3-beta-hotfix1_ubuntu-16.04.zip
unzip travis_v0.1.3-beta-hotfix1_ubuntu-16.04.zip

# or if your os is CentOS
wget https://github.com/CyberMiles/travis/releases/download/v0.1.3-beta-hotfix1/
↪travis_v0.1.3-beta-hotfix1_centos-7.zip
unzip travis_v0.1.3-beta-hotfix1_centos-7.zip

./travis node start
```

Option 4 (the hard way): Docker from Genesis

Prerequisite

Please setup docker.

Docker Image

Docker image for Travis is stored on Docker Hub. Genesis starts from software version `0.1.2-beta`

```
docker pull cybermiles/travis:v0.1.2-beta
```

Note: Configuration and data will be stored at `/travis` directory in the container. The directory will also be exposed as a volume. The ports 8545, 26656 and 26657 will be exposed for connection.

Getting Travis MainNet Config

```
rm -rf $HOME/.travis
docker run --rm -v $HOME/.travis:/travis -t cybermiles/travis:v0.1.2-beta node init --
↪env mainnet --home /travis
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪config.toml > $HOME/.travis/config/config.toml
curl https://raw.githubusercontent.com/CyberMiles/testnet/master/travis/init-mainnet/
↪genesis.json > $HOME/.travis/config/genesis.json
```

Start the Node and Join MainNet

First change your name from default name `local`.

```
vim ~/.travis/config/config.toml

# here you can change your name
moniker = "<your_custom_name>"
```

Run the docker Travis application:

```
docker run --privileged --name travis -v $HOME/.travis:/travis -p 26657:26657 -t␣
↪cybermiles/travis:v0.1.2-beta node start --home /travis
```

Upgrade and Continue

At certain block heights, the node will stop. Download the next version of the software (e.g., `0.1.3-beta-hotfix1` at block height 230767), and restart. A notable exception is the switch between `0.1.3-beta-hotfix1` and `0.1.3-beta-hotfix2` – that has to happen manually within a specific range of block heights.

```
docker stop travis
docker rm travis

docker pull cybermiles/travis:v0.1.3-beta-hotfix1
docker run --privileged --name travis -v $HOME/.travis:/travis -p 26657:26657 -t␣
↪cybermiles/travis:v0.1.3-beta-hotfix1 node start --home /travis
```

1.4 Transactions

The CyberMiles blockchain is fully backward compatible with the Ethereum protocol. That means all Ethereum transactions are supported on the CyberMiles blockchain. For developers, we recommend you use the web3-cmt.js library interact with the blockchain. The `web3-cmt.js` library is a customized version of the Ethereum web3.js

library, with the `eth` module renamed to the `cmt` module. The `web3-cmt.js` library is integrated into the `travis` client console by default.

Beyond Ethereum, however, the most important transactions that are specific for the CyberMiles blockchain are for DPoS staking operations and for blockchain governance operations.

1.4.1 Staking transactions

A key characteristic of the CyberMiles blockchain is the DPoS consensus mechanism. You can read more about the CyberMiles DPoS protocol. With the staking transactions, CMT holders can declare candidacy for validators, stake and vote for candidates, and unstake as needed. Learn more about the staking transactions for validators and the staking transactions for delegators.

1.4.2 Governance transactions

With the DPoS consensus mechanism, the CyberMiles validators can make changes to the blockchain network's key parameters, deploy new libENI libraries, create trusted contracts, and make other policy changes. Anyone on the blockchain can propose governance changes, but only the current validators can vote. Learn more about the governance transactions.

1.4.3 "Free" transactions

Most CyberMiles transactions require the caller to pay a gas fee. However, it is sometimes possible to set the `gasPrice` to zero and avoid paying the fee altogether! When the caller sends in a transaction with `gasPrice=0`, the node determines whether to reject the transaction.

- If the transaction `gasLimit` is less than 500,000 (this is an adjustable parameter in the system), the transaction will be accepted and executed for free.
- If the transaction `gasLimit > 500000` and the transaction is a smart contract function call, the function call will be executed by the VM. However, the VM would then require the function to be `freegas`, meaning that the gas fee will come from the contract itself. If the `freegas` requirements are not met, the VM will fail the transaction. Learn more about the freegas transactions

CyberMiles allows only one free transaction per block for the same from / to addresses.

1.5 Virtual Machine

A key feature of the CyberMiles blockchain platform is the Lity programming language and its high performance extensible virtual machine.

Please go to www.LityLang.org for more details on the Lity language and CyberMiles Virtual Machine.

1.6 Travis JSON-RPC

As travis is compatible with Ethereum so most methods especially the ones for normal transaction are the same with Ethereum. Please refer to Ethereum JSON-RPC for more information.

1.6.1 CMT methods

cmt_syncing

Returns the sync object.

Parameters

none

Returns

- `latest_block_hash` Number - The hash of the latest block.
- `latest_app_hash` Number - The hash of latest application state.
- `latest_block_height` Number - The latest block number.
- `latest_block_time` Number - The latest block time.
- `catching_up` Boolean - Whether the node is syncing or not.

Example

```
// Request
curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method":
↪"cmt_syncing","params":[],"id":1}'

// Result
{
        "jsonrpc": "2.0",
        "id": 1,
        "result": {
                "latest_block_hash": "94C0363F68AD5184A861FAE0010BE0D44FDD3254",
                "latest_app_hash": "BB510006FDB4A907A3C7BEAA4A8A2F493252DDCD",
                "latest_block_height": 115851,
                "latest_block_time": "2018-10-30T04:58:17.895717492Z",
                "catching_up": false
        }
}
```

cmt_getBlockByNumber

Returns a block matching the block number.

Parameters

- `blockNumber` Number - The block number.
- `decodeTx` Boolean - Currently should always be false.

Returns

The block object.

Example

```
// Request
curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method":
↪"cmt_getBlockByNumber","params":[78, false],"id":1}'
```

(continues on next page)

(continued from previous page)

```
// Result
{
        "jsonrpc": "2.0",
        "id": 1,
        "result": {
                "block_meta": {
                        "block_id": {
                                "hash": "E0F9C6439B41E1B80E4D2C4C96EDFD100B4BAEC7",
                                "parts": {
                                        "total": 1,
                                        "hash":
↪"C78D31D2B57749A3C67EC8F04A6A9DF396365588"
                                }
                        },
                        "header": {
                                "chain_id": "CyberMiles",
                                "height": 78,
                                "time": "2018-10-15T13:41:41.109630547Z",
                                "num_txs": 0,
                                "last_block_id": {
                                        "hash":
↪"50E1722AE5E7FA9D2C4E939356FF0F9487C26E03",
                                        "parts": {
                                                "total": 1,
                                                "hash":
↪"94F88571468784DD05B8BB963358D5E47B68EDB6"
                                        }
                                },
                                "total_txs": 0,
                                "last_commit_hash":
↪"E05893F0935E1BC259514BC44B61DD8E8962BE8A",
                                "data_hash": "",
                                "validators_hash":
↪"3760C3CD67AC9A819AF01747476E1B04DABCD05B",
                                "consensus_hash":
↪"D6B74BB35BDFFD8392340F2A379173548AE188FE",
                                "app_hash": "2144AC53826041B1406CB6B8ABEDC37064211CA5
↪",
                                "last_results_hash": "",
                                "evidence_hash": ""
                        }
                },
                "block": {
                        "header": {
                                "chain_id": "CyberMiles",
                                "height": 78,
                                "time": "2018-10-15T13:41:41.109630547Z",
                                "num_txs": 0,
                                "last_block_id": {
                                        "hash":
↪"50E1722AE5E7FA9D2C4E939356FF0F9487C26E03",
                                        "parts": {
                                                "total": 1,
                                                "hash":
↪"94F88571468784DD05B8BB963358D5E47B68EDB6"
                                        }
                                },
```

(continues on next page)

(continued from previous page)


```
                                "total_txs": 0,
                                "last_commit_hash":
↪"E05893F0935E1BC259514BC44B61DD8E8962BE8A",
                                "data_hash": "",
                                "validators_hash":
↪"3760C3CD67AC9A819AF01747476E1B04DABCD05B",
                                "consensus_hash":
↪"D6B74BB35BDFFD8392340F2A379173548AE188FE",
                                "app_hash": "2144AC53826041B1406CB6B8ABEDC37064211CA5
↪",
                                "last_results_hash": "",
                                "evidence_hash": ""
                        },
                        "data": {
                                "txs": null
                        },
                        "evidence": {
                                "evidence": null
                        },
                        "last_commit": {
                                "block_id": {
                                        "hash":
↪"50E1722AE5E7FA9D2C4E939356FF0F9487C26E03",
                                        "parts": {
                                                "total": 1,
                                                "hash":
↪"94F88571468784DD05B8BB963358D5E47B68EDB6"
                                        }
                                },
                                "precommits": [{
                                        "validator_address":
↪"04A515F3B6B9E7FC7E2B5AAC4304D82BE9D6573C",
                                        "validator_index": 0,
                                        "height": 77,
                                        "round": 0,
                                        "timestamp": "2018-10-15T13:41:30.824095471Z",
                                        "type": 2,
                                        "block_id": {
                                                "hash":
↪"50E1722AE5E7FA9D2C4E939356FF0F9487C26E03",
                                                "parts": {
                                                        "total": 1,
                                                        "hash":
↪"94F88571468784DD05B8BB963358D5E47B68EDB6"
                                                }
                                        },
                                        "signature": [246, 31, 74, 206, 79, 252, 63,␣
↪8, 62, 221, 28, 28, 174, 45, 191, 121, 163, 69, 96, 83, 245, 141, 165, 145, 28, 240,
↪ 248, 236, 42, 14, 180, 184, 194, 78, 146, 10, 24, 193, 243, 43, 50, 166, 7, 159,␣
↪99, 23, 155, 56, 35, 167, 152, 4, 86, 107, 14, 51, 9, 203, 38, 149, 248, 147, 226,␣
↪7]
                                }]
                        }
                }
        }
}
```

cmt_getTransactionByHash

Returns a transaction matching the given transaction hash.

Parameters

- `transactionHash` String - The transaction hash.

Returns

The transaction object.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_getTransactionByHash","params":["1F64261396674A1A7328B250EC3043E5512010D8"],
↪"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "blockNumber": "0x1c6f6",
                    "from": "0x245323885234fd5adc48ffb546a54c5df99e9ace",
                    "gas": "0x0",
                    "gasPrice": "0x0",
                    "hash":
↪"0xa73b917243b5d3fb810dfb5f1880daab71564aafbb183c5f1e1f40665832aad5",
                    "cmtHash": "1F64261396674A1A7328B250EC3043E5512010D8",
                    "input":
↪"0x7b2274797065223a227374616b655c2f64656c6567617465222c2264617461223a7b2276616c696461746f725f61646
↪",
                    "cmtInput": {
                            "type": "stake/delegate",
                            "data": {
                                    "validator_address":
↪"0xf9fd397486ac5516eea2304fa701cb9767c465d2",
                                    "amount": "3437100000000000000000",
                                    "cube_batch": "02",
                                    "sig":
↪"25c896e1b505cd28bdcc26ee90d934e5ca115852f209775f5cd43b0caf96ea14d99b63e0448087d526848179baebd3ad05
↪"
                            }
                    },
                    "nonce": "0x0",
                    "to": null,
                    "transactionIndex": "0x0",
                    "value": "0x0",
                    "v": "0x48",
                    "r":
↪"0x224015941f4373e5aee27a1173b9ae112317dfdc3b2a1a86cf557c2446c255e4",
                    "s":
↪"0x2798b6ab9f403b938fea0b640476de20a6f09d1e12f86f0cd5e18369164e56ef",
                    "txResult": {
                            "fee": {}
                    }
            }
    }
```

cmt_getTransactionFromBlock

Returns a transaction based on a block hash or number and the transactions index position

Parameters

- `blockNumber` Number - The block number.
- `indexNumber` Number - The transactions index position.

Returns

The transaction object.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_getTransactionFromBlock","params":[116470, 0],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "blockNumber": "0x1c6f6",
                    "from": "0x245323885234fd5adc48ffb546a54c5df99e9ace",
                    "gas": "0x0",
                    "gasPrice": "0x0",
                    "hash":
↪"0xa73b917243b5d3fb810dfb5f1880daab71564aafbb183c5f1e1f40665832aad5",
                    "cmtHash": "1F64261396674A1A7328B250EC3043E5512010D8",
                    "input":
↪"0x7b2274797065223a227374616b655c2f64656c6567617465222c2264617461223a7b2276616c696461746f725f616464
↪",
                    "cmtInput": {
                            "type": "stake/delegate",
                            "data": {
                                    "validator_address":
↪"0xf9fd397486ac5516eea2304fa701cb9767c465d2",
                                    "amount": "34371000000000000000000",
                                    "cube_batch": "02",
                                    "sig":
↪"25c896e1b505cd28bdcc26ee90d934e5ca115852f209775f5cd43b0caf96ea14d99b63e0448087d526848179baebd3ad05
↪"
                            }
                    },
                    "nonce": "0x0",
                    "to": null,
                    "transactionIndex": "0x0",
                    "value": "0x0",
                    "v": "0x48",
                    "r":
↪"0x224015941f4373e5aee27a1173b9ae112317dfdc3b2a1a86cf557c2446c255e4",
                    "s":
↪"0x2798b6ab9f403b938fea0b640476de20a6f09d1e12f86f0cd5e18369164e56ef",
                    "txResult": {
                            "fee": {}
                    }
```

(continues on next page)

(continued from previous page)

```
            }
    }
```

1.6.2 Staking Validator methods

cmt_declareCandidacy

Allows a potential validator declares its candidacy.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. It will be associated with this validator (for self-staking and in order to get paid).
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `pubKey` String - Validator node public key.
- `maxAmount` String - Max amount of CMTs in Wei to be staked.
- `compRate` String - Validator compensation. That is the percentage of block awards to be distributed back to the validators.
- **`description` Object - (optional) Description object as follows:**
  - `name` String - Validator name.
  - `website` String - Web page link.
  - `location` String - Location(network and geo).
  - `email` String - Email.
  - `profile` String - Detailed description.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_declareCandidacy","params":[{"from":
↪"0xc4abd0339eb8d57087278718986382264244252f", "pubKey":
↪"051FUvSNJmVL4UiFL7ucBr3TnGqG6a5JgUIgKf4UOIA=", "maxAmount":"0xF4240", "compRate":
↪"0.2"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
```

(continues on next page)

(continued from previous page)

```
                    },
                    deliver_tx: {
                            "gasUsed": "1000000",
                            "fee": {
                                    "key": "R2FzRmVl",
                                    "value": "2000000000000000"
                            }
                    },
                    hash: '1573F39376D8C10C6B890861CD25FD0BA917556F',
                    height: 271
            }
    }
```

cmt_updateCandidacy

Allows a validator candidate to change its candidacy.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `pubKey` String - (optional) Validator node public key.
- `maxAmount` String - (optional) New max amount of CMTs in Wei to be staked.
- `compRate` String - (optional) Validator compensation. That is the percentage of block awards to be distributed back to the validators.
- **`description` Object - (optional) When updated, the verified status will set to false:**
  - `name` String - Validator name.
  - `website` String - Web page link.
  - `location` String - Location(network and geo).
  - `email` String - Email.
  - `profile` String - Detailed description.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_updateCandidacy","params":[{"from":
↪"0xc4abd0339eb8d57087278718986382264244252f", "maxAmount":"0xF4240", "description":
↪{"website": "https://www.example.com"}}],"id":1}'

// Result
```

(continues on next page)

(continued from previous page)

```
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            "gasUsed": "1000000",
                            "fee": {
                                    "key": "R2FzRmVl",
                                    "value": "2000000000000000"
                            }
                    },
                    hash: '1B11C4D5EA9664DB1DD3A9CDD86741D6C8E226E9',
                    height: 297
            }
    }
```

cmt_withdrawCandidacy

Allows a validator to withdraw.

Parameters

- `from` String - The address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_withdrawCandidacy","params":[{"from":
↪"0xc4abd0339eb8d570872787189863822644244252f"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: '4A723894821166EFC7DDD4FD92BE8D855B3FDBAC',
```

(continues on next page)

(continued from previous page)

```
                    height: 311
            }
    }
```

cmt_verifyCandidacy

Allows the foundation to "verify" a validator's information.

Parameters

- `from` String - A special address the foundation owns. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `candidateAddress` String - The address of validator to verfify.
- `verified` Boolean - (optional) Verified true or false, default to false.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_verifyCandidacy","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "candidateAddress":
↪"0xc4abd0339eb8d57087278718986382264244252f", "verified":true}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: 'EADC546C764AFF6C176B843321B5AB090FBEC0DA',
                    height: 334
            }
    }
```

cmt_activateCandidacy

Allows a "removed" validator to re-activate itself.

Parameters

- `from` String - The address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_activateCandidacy","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: 'FB70A78AD62A0E0B24194CA951725770B2EFBC0A',
                    height: 393
            }
    }
```

cmt_deactivateCandidacy

Allows a validator to deactivate itself.

Parameters

- `from` String - The address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_deactivateCandidacy","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc"}],"id":1}'
```

(continues on next page)

(continued from previous page)

```
// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: 'FB70A78AD62A0E0B24194CA951725770B2EFBC0A',
                    height: 393
            }
    }
```

cmt_setCompRate

Allows a validator to update the compensation rate for its delegators.

Parameters

- `from` String - The address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `delegatorAddress` String - The adddress of delegator.
- `compRate` String - New compensation rate to set for the delegator. Compensation rate is the percentage of block awards to be distributed back to the validators.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_setCompRate","params":[{"from":"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc",
↪"delegatorAddress":"0x38d7b32e7b5056b297baf1a1e950abbaa19ce949", "compRate":"0.3"}],
↪"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
```

(continues on next page)

(continued from previous page)

```
                        "gasUsed": "1000000",
                        "fee": {
                                "key": "R2FzRmVl",
                                "value": "2000000000000000"
                        }
                },
                hash: 'C61BAEEEF637CB554157261DF27F7D1CFE50F251',
                height: 393
        }
    }
```

cmt_updateCandidacyAccount

A validator requests to update its binding address.

Parameters

- `from` String - The address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `newCandidateAccount` String - The new adddress of the validator.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed. If successful, the requestId will be set in the data property(base64 encoded), for the new address to accept later.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_updateCandidacyAccount","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "newCandidateAccount":
↪"0x283ED77f880D87dBdE8721259F80517A38ae5b4f"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            data: "MQ==",
                            gasUsed: "1000000",
                            fee: {
                                    key: "R2FzRmVl",
                                    value: "2000000000000000"
                            }
                    },
                    hash: "34B157D42AFF2D8327FC8CEA8DFFC1E61E9C0D93",
```

(continues on next page)

(continued from previous page)

```
                    height: 105
            }
    }
```

cmt_acceptCandidacyAccountUpdate

A validator uses its new address to accept an account updating request.

Parameters

- `from` String - The new address for the validator. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `accountUpdateRequestId` int64 - The account updating request id.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_acceptCandidacyAccountUpdate","params":[{"from":
↪"0x283ed77f880d87dbde8721259f80517a38ae5b4f", "accountUpdateRequestId":1}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            gasUsed: "1000000",
                            fee: {
                                    key: "R2FzRmVl",
                                    value: "2000000000000000"
                            }
                    },
                    hash: "D343D115C152D1A78B7DB9CAA2160E3BA31A3F63",
                    height: 67
            }
    }
```

cmt_queryValidator

Query the current staking status of a specific validator.

Parameters

- `validatorAddress` String - The validator address.
- `height` Number - The block number. Default to 0, means current head of the blockchain. NOT IMPLEMENTED YET.

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Object - The validator object.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryValidator","params":["0x858578e81a0259338b4d897553afa7b9c363e769", 0],
↪"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "jsonrpc": "2.0",
                    "id": 1,
                    "result": {
                            "height": 116992,
                            "data": {
                                    "id": 29,
                                    "pub_key": {
                                            "type": "tendermint/PubKeyEd25519",
                                            "value":
↪"NEUjQM4EvOkTruH0aufgQM4tLEKrCJSAvEDKwZ771ng="
                                    },
                                    "owner_address":
↪"0x858578e81a0259338b4d897553aFA7b9c363e769",
                                    "shares": "20989543781473532838491O5",
                                    "voting_power": 161882,
                                    "pending_voting_power": 0,
                                    "max_shares": "200000000000000000000000000",
                                    "comp_rate": "1/2",
                                    "created_at": 1539619422,
                                    "description": {
                                            "name": "Rfinex",
                                            "website": "https://www.rfinex.com",
                                            "location": "Geneva, Switzerland",
                                            "email": "",
                                            "profile": "Make Crypto Greater"
                                    },
                                    "verified": "Y",
                                    "active": "Y",
                                    "block_height": 881,
                                    "rank": 15,
                                    "state": "Validator",
                                    "num_of_delegators": 2
                            }
                    }
            }
    }
```

cmt_queryValidators

Returns a list of all current validators and validator candidates.

Parameters

- `height` Number - The block number. Default to 0, means current head of the blockchain. NOT IMPLEMENTED YET.

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Array - An array of all current validators and validator candidates.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryValidators","params":[0],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "jsonrpc": "2.0",
                    "id": 1,
                    "result": {
                            "height": 117008,
                            "data": [{
                                    "id": 9,
                                    "pub_key": {
                                            "type": "tendermint/PubKeyEd25519",
                                            "value": "aIVtdAdQlQ4uuTMmsU+8z9d//
↪+URrPKX2vcobWDO6HA="
                                    },
                                    "owner_address":
↪"0x5c158B32dE3037d5BC6D2Ebff1b9cF099daF1F7D",
                                    "shares": "525829971878780385668796",
                                    "voting_power": 48854,
                                    "pending_voting_power": 0,
                                    "max_shares": "500000000000000000000000",
                                    "comp_rate": "99/100",
                                    "created_at": 0,
                                    "description": {
                                            "name": "Seed Validator",
                                            "website": "https://www.cybermiles.io/
↪seed-validator/",
                                            "location": "HK",
                                            "email": "developer@cybermiles.io",
                                            "profile": "To be replaced by an external␣
↪validator."
                                    },
                                    "verified": "Y",
                                    "active": "Y",
                                    "block_height": 0,
                                    "rank": 18,
                                    "state": "Validator",
                                    "num_of_delegators": 10
```

(continues on next page)

(continued from previous page)

```
                                }]
                        }
                }
        }
```

cmt_queryAwardInfos

Returns award information of all current validators and backup validators.

Parameters

- `height` Number - The block number. Default to 0, means current head of the blockchain.

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Array - An array of award information of all current validators and backup validators.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryAwardInfos","params":[0],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "jsonrpc": "2.0",
                    "id": 1,
                    "result": {
                            "height": 117024,
                            "data": [{
                                    "address":
↪"0x1ac7d4f1d4fa3eaef67d8208a2b1b84670211e75",
                                    "state": "Validator",
                                    "amount": "695443350547290695"
                            }, {
                                    "address":
↪"0x1724d4a82f29d93a1eb96c19b4bb6b219dc18f23",
                                    "state": "Validator",
                                    "amount": "1221147599113753229"
                            }, {
                                    "address":
↪"0x4cdaf011cadba6c3997252738e4d6dd30c8865b9",
                                    "state": "Validator",
                                    "amount": "551560392982316129"
                            }, {
                                    "address":
↪"0xfd0e8e4c4dea053f10e72e8800b08ac875e5ac49",
                                    "state": "Validator",
                                    "amount": "624496861299062028"
                            }, {
                                    "address":
↪"0x8958618332df62af93053cb9c535e26462c959b0",
```

(continues on next page)

(continued from previous page)


```
                                    "state": "Validator",
                                    "amount": "1043204078311188515"
                                }, {
                                    "address":
↪"0x5c158b32de3037d5bc6d2ebff1b9cf099daf1f7d",
                                    "state": "Validator",
                                    "amount": "165901770329933149"
                                }, {
                                    "address":
↪"0x1b92c5bb82972af385d8cd8c1230502083898ba6",
                                    "state": "Validator",
                                    "amount": "3773502902478876527"
                                }, {
                                    "address":
↪"0xcd3090e881170f6d036fdb3ae5a3d36ead5bcf83",
                                    "state": "Validator",
                                    "amount": "1251133119890215962"
                                }, {
                                    "address":
↪"0x3af427d092f9bf934d2127408935c1455170ea8a",
                                    "state": "Validator",
                                    "amount": "837444996842711426"
                                }, {
                                    "address":
↪"0x70a52ff393256f016939ae2926cbd999508a555b",
                                    "state": "Validator",
                                    "amount": "1129557623931196482"
                                }, {
                                    "address":
↪"0x34c5f1c0e10701dbaf0df1ad2a7826be41a3a380",
                                    "state": "Validator",
                                    "amount": "3188316617454823732"
                                }, {
                                    "address":
↪"0xeb65290b802df113300120c52b313f1896e80d38",
                                    "state": "Validator",
                                    "amount": "673088346779289101"
                                }, {
                                    "address":
↪"0xf9fd397486ac5516eea2304fa701cb9767c465d2",
                                    "state": "Validator",
                                    "amount": "696852636065097266"
                                }, {
                                    "address":
↪"0xf9a431660dc8e425018564ce707d44a457301eb9",
                                    "state": "Validator",
                                    "amount": "538679862936435824"
                                }, {
                                    "address":
↪"0x9a3482fd81d706d5aa941f38946af69a448e08c3",
                                    "state": "Validator",
                                    "amount": "1780039672657381152"
                                }, {
                                    "address":
↪"0x858578e81a0259338b4d897553afa7b9c363e769",
                                    "state": "Validator",
                                    "amount": "549733415612243995"
```


(continues on next page)

(continued from previous page)


```
                            }, {
                                    "address":
↪"0x0da518ecf4761a86965c1f77ac4c1bd6e19904e3",
                                    "state": "Validator",
                                    "amount": "654220900184269072"
                            }, {
                                    "address":
↪"0x221507f21aac826263a664538580e57ded401978",
                                    "state": "Validator",
                                    "amount": "3022099031334153293"
                            }, {
                                    "address":
↪"0x654e1dfe66519b9a09305ad58392d9a1c61296b3",
                                    "state": "Validator",
                                    "amount": "434627049560264340"
                            }, {
                                    "address":
↪"0x482a7cbb8f66a9db6b25808861b182c670c79259",
                                    "state": "Backup Validator",
                                    "amount": "460623994544478727"
                            }, {
                                    "address":
↪"0x04ba6cf9a4035294958678dd0f540a195b260d0e",
                                    "state": "Backup Validator",
                                    "amount": "577538694065296547"
                            }, {
                                    "address":
↪"0xe218509490578f75dfc6ed6c8a80158675071a8c",
                                    "state": "Backup Validator",
                                    "amount": "577517942432438752"
                            }, {
                                    "address":
↪"0xae3befdc5d0f5397b9e448fe136f10360dddde28",
                                    "state": "Backup Validator",
                                    "amount": "461557818023079509"
                            }, {
                                    "address":
↪"0x72cf924c62baff2ed74a5ceb885082b814216e55",
                                    "state": "Backup Validator",
                                    "amount": "459544909635873380"
                            }]
                    }
            }
    }
```


1.6.3 Staking Delegator methods

cmt_delegate

Used by a delegator to stake CMTs to a validator.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.

- `validatorAddress` String - The address of validator to delegate.
- `amount` String - Amount of CMTs in Wei to delegate.
- `cubeBatch` String - The batch number of the CMT cube. Use "01" for testing.
- `sig` String - delegator_address|nonce signed by the CMT cube. Check this for how to generate a signature for testing.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_delegate","params":[{"from":"0x38d7b32e7b5056b297baf1a1e950abbaa19ce949",
↪"validatorAddress":"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "amount":"0x186A0",
↪ "cubeBatch":"01", "sig":
↪"036b6dddefdb1d798a9847121dde8c38713721869a24c77abe2249534f6d98622727720994f663ee9cc446c6e246781caa
↪"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: '8A40C44D31316BFB2D417A1985E03DA36145EF5A',
                    height: 319
            }
    }
```

cmt_withdraw

Used by a delegator to unbind staked CMTs from a validator.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `validatorAddress` String - The address of validator to withdraw.
- `amount` String - Amount of CMTs in Wei to withdraw.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.

- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_withdraw","params":[{"from":"0x38d7b32e7b5056b297baf1a1e950abbaa19ce949",
↪"validatorAddress":"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "amount":"0x186A0"}
↪],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: '8A40C44D31316BFB2D417A1985E03DA36145EF5A',
                    height: 319
            }
    }
```

cmt_queryDelegator

Query the current staking status of a specific delegator.

Parameters

- `delegatorAddress` String - The delegator address.
- `height` Number - The block number. Default to 0, means current head of the blockchain. NOT IMPLEMENTED YET.

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Object - The delegator object.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryDelegator","params":["0x3a436deae68b7d4c8ff9f1cb0498913a397472d7", 0],
↪"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "height": 117466,
```

(continues on next page)

(continued from previous page)


```
                    "data": [{
                            "id": 780,
                            "delegator_address":
↪"0xcc64debb948ff9a2cb9ac5cbd292cef1d380221f",
                            "pub_key": {
                                    "type": "tendermint/PubKeyEd25519",
                                    "value":
↪"sbmLYMzeezCgqJKQBXNVAiZtsdSAx75JUzAtwzWv9pw="
                            },
                            "validator_address":
↪"0x70A52fF393256f016939Ae2926CBd999508A555B",
                            "delegate_amount": "34310000000000000000000",
                            "award_amount": "39040378244652451965",
                            "withdraw_amount": "0",
                            "pending_withdraw_amount": "0",
                            "slash_amount": "0",
                            "comp_rate": "1/4",
                            "voting_power": 970,
                            "created_at": 1540551045,
                            "state": "Y",
                            "block_height": 86269,
                            "average_staking_date": 4,
                            "candidate_id": 30
                    }]
            }
    }
```


1.6.4 Governance methods

cmt_proposeTransferFund

Propose a fund recovery proposal.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. Must be a validator.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `transferFrom` String - From account address.
- `transferTo` String - To account address.
- `amount` String - Amount of CMTs in Wei.
- `reason` String - (optional) Reason.
- `expireBlockHeight` Number - (optional) Expiration block height.
- `expireTimestamp` Number - (optional) Timestamp when the proposal will expire.

Note: You can specify expiration block height or timestamp, but not both. If none is specified, a default of 7 days, as measured in block height(7 * 24 * 60 * 60 / 10), will be used.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.

- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed. If successful, the ProposalID will be set in the data property, for validators to vote later.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_proposeTransferFund","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "transferFrom":
↪"0xc4abd0339eb8d57087278718986382264244252f", "transferTo":
↪"0x11f4d0A3c12e86B4b5F39B213F7E19D048276DAe", "amount":"0x186A0"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            data: 'JTUx+ODH0/OSdgfC0Sn66qjn2tX8LfvbiwnArzNpIus=',
                            gasUsed ": '2000000',
                            fee: {
                                    key: 'R2FzRmVl',
                                    value: "4000000000000000'
                            }
                    },
                    hash: '95A004438F89E809657EB119ACBDB42A33725B39',
                    height: 561
            }
    }
```

cmt_proposeChangeParam

Propose a system parameter change.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. Must be a validator.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `name` String - The name of the parameter.
- `value` String - New value of the parameter.
- `reason` String - (optional) Reason.
- `expireBlockHeight` Number - (optional) Expiration block height.
- `expireTimestamp` Number - (optional) Timestamp when the proposal will expire.

Note: You can specify expiration block height or timestamp, but not both. If none is specified, a default of 7 days, as measured in block height(7 * 24 * 60 * 60 / 10), will be used.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.

- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed. If successful, the ProposalID will be set in the data property, for validators to vote later.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_proposeChangeParam","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "name":"gas_price", "value":
↪"3000000000"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            data: 'JTUx+ODH0/OSdgfC0Sn66qjn2tX8LfvbiwnArzNpIus=',
                            gasUsed ": '2000000',
                            fee: {
                                    key: 'R2FzRmVl',
                                    value: "4000000000000000'
                            }
                    },
                    hash: '95A004438F89E809657EB119ACBDB42A33725B39',
                    height: 561
            }
    }
```

cmt_proposeDeployLibEni

Propose a new library for ENI.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. Must be a validator.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `name` String - The name of the library.
- `version` String - Version of the library, data format: vX.Y.Z, where X, Y, and Z are non-negative integers.
- `fileUrl` String - JSON string of key/value pairs. Key is the name of the OS(so far, only ubuntu and centos are supported), value is the URL array to retrieve the library file.
- `md5` String - JSON string of key/value pairs. Key is the name of the OS(so far, only ubuntu and centos are supported), value is the MD5 of the library file.
- `reason` String - (optional) Reason.

- `deployBlockHeight` Number - (optional) The block number where the new ENI library will deploy.
- `deployTimestamp` Number - (optional) Timestamp when the new ENI library will deploy.

Note: You can specify deploy block height or timestamp, but not both. If none is specified, a default of 7 days, as measured in block height(7 * 24 * 60 * 60 / 10), will be used.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed. If successful, the ProposalID will be set in the data property, for validators to vote later.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_proposeDeployLibEni","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "name":"reverse", "version":"v1.0.0",
↪"fileUrl":"{\"ubuntu\": [\"<url1>\", \"<url2>\"], \"centos\": [\"<url1>\", \"<url2>\
↪"]}", "md5":"{\"ubuntu\": \"<md5 text>\", \"centos\": \"<md5 text>\"}"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            data: 'JTUx+ODH0/OSdgfC0Sn66qjn2tX8LfvbiwnArzNpIus=',
                            gasUsed ": '2000000',
                            fee: {
                                    key: 'R2FzRmVl',
                                    value: "4000000000000000'
                            }
                    },
                    hash: '95A004438F89E809657EB119ACBDB42A33725B39',
                    height: 561
            }
    }
```

cmt_proposeRetireProgram

Propose to retire the program.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. Must be a validator.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.

- `preservedValidators` String - A comma seperated validator public key list. Valiators in this list will be preserved, other validators will be deactivated.
- `reason` String - (optional) Reason.
- `retiredBlockHeight` Number - (optional) The block number where the program will retire. If not specified, a default of 7 days, as measured in block height(7 * 24 * 60 * 60 / 10), will be used.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed. If successful, the ProposalID will be set in the data property, for validators to vote later.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_proposeRetireProgram","params":[{"from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc", "preservedValidators":"Esdo0ZN+nHduoi/
↪kNqjdQSNFmNyv2M3Tie/eZeC25gM=,X6qJkoWxW8YkEHquJQM7mZcfpt5r+l8V6C8rbg8dEHQ=", "reason
↪":"System Upgrade"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            data: 'JTUx+ODH0/OSdgfC0Sn66qjn2tX8LfvbiwnArzNpIus=',
                            gasUsed ": '2000000',
                            fee: {
                                    key: 'R2FzRmVl',
                                    value: "4000000000000000'
                            }
                    },
                    hash: '95A004438F89E809657EB119ACBDB42A33725B39',
                    height: 561
            }
    }
```

cmt_vote

Vote on proposals of making changes to the system state.

Here are some use cases:

- Vote to change system wide parameters such as the system inflation rate.
- Vote to accept new native libraries for ENI.
- Vote to recover funds for users.

Parameters

- `from` String - The address for the sending account. Uses the web3.cmt.defaultAccount property, if not specified. Must be a validator.
- `nonce` Number - (optional) The number of transactions made by the sender prior to this one.
- `proposalId` String - The Proposal ID to vote.
- `answer` String - Y or N.

Returns

- `height` Number - The block number where the transaction is in. =0 if failed.
- `hash` String - Hash of the transaction.
- `check_tx` Object - CheckTx result. Contains error code and log if failed.
- `deliver_tx` Object - DeliverTx result. Contains error code and log if failed.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_vote","params":[{"from":"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc",
↪"proposalId":"JTUx+ODH0/OSdgfC0Sn66qjn2tX8LfvbiwnArzNpIus=", "answer":"Y"}],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    check_tx: {
                            fee: {}
                    },
                    deliver_tx: {
                            fee: {}
                    },
                    hash: '95A004438F89E809657EB119ACBDB42A33725B39',
                    height: 561
            }
    }
```

cmt_queryProposals

Returns a list of all proposals.

Parameters

none

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Array - An array of all proposals

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryProposals","params":[],"id":1}'
```

(continues on next page)

(continued from previous page)

```
// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "height": 58,
                    "data": [{
                                    "Id": "/
↪YRNInf2DpWJ6KBcS+Xqa+EUiBH3DMgeM2T57tsMd2E=",
                                    "Type": "transfer_fund",
                                    "Proposer":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc",
                                    "BlockHeight": 15,
                                    "ExpireBlockHeight": 20,
                                    "CreatedAt": "2018-07-03T14:27:11Z",
                                    "Result": "Expired",
                                    "ResultMsg": "",
                                    "ResultBlockHeight": 20,
                                    "ResultAt": "2018-07-03T14:28:01Z",
                                    "Detail": {
                                            "amount": "16",
                                            "from":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc",
                                            "reason": "",
                                            "to":
↪"0xd5bb0351974eca5d116eff840a03a9b96d8ba9e7"
                                    }
                            },
                            {
                                    "Id":
↪"DN6utTAmgX9Iy7naroaKgO2dEbIkwmwRPmmfk35cdEE=",
                                    "Type": "change_param",
                                    "Proposer":
↪"0x7eff122b94897ea5b0e2a9abf47b86337fafebdc",
                                    "BlockHeight": 16,
                                    "ExpireBlockHeight": 60496,
                                    "CreatedAt": "2018-07-03T14:27:21Z",
                                    "Result": "",
                                    "ResultMsg": "",
                                    "ResultBlockHeight": 0,
                                    "ResultAt": "",
                                    "Detail": {
                                            "name": "gas_price",
                                            "reason": "test",
                                            "value": "3000000000"
                                    }
                            }
                    ]
            }
    }
```

cmt_queryParams

Returns current settings of system parameters.

Parameters

- `height` Number - The block number. Default to 0, means current head of the blockchain.

Returns

- `height` Number - Current block number or the block number if specified.
- `data` Array - An array of all proposals.

Example

```
    // Request
    curl -X POST -H "Content-Type: application/json" --data '{"jsonrpc":"2.0","method
↪":"cmt_queryParams","params":[0],"id":1}'

// Result
    {
            "jsonrpc": "2.0",
            "id": 1,
            "result": {
                    "height": 1000,
                    "data": {
                            "max_vals": 19,
                            "backup_vals": 5,
                            "self_staking_ratio": "1/10",
                            "inflation_rate": "2/25",
                            "validator_size_threshold": "3/25",
                            "unstake_waiting_period": 60480,
                            "proposal_expire_period": 60480,
                            "declare_candidacy_gas": 1000000,
                            "update_candidacy_gas": 1000000,
                            "set_comp_rate_gas": 21000,
                            "update_candidate_account_gas": 1000000,
                            "accept_candidate_account_update_request_gas": 1000000,
                            "transfer_fund_proposal_gas": 2000000,
                            "change_params_proposal_gas": 2000000,
                            "deploy_libeni_proposal_gas": 2000000,
                            "retire_program_proposal_gas": 2000000,
                            "upgrade_program_proposal_gas": 2000000,
                            "gas_price": 2000000000
                    }
            }
    }
```

1.7 CMT Wallet - dApp SDK Guideline

1.7.1 Introduction

This document helps DApp developers access the CMT Wallet DApp SDK.

In general, DApp requires a hosting environment to interact with the user's wallet, just like metamask, CMT Wallet provides this environment in the app.

In DApp browser, DApp can do the same and more things in Metamask.

To keep things simple, this document will use DApp browser for CMT Wallet DApp browser, DApp for DApp webpage.

1.7.2 Web3JS

CMT Wallet DApp browser is fully compatible with metamask, you can migrate DApp directly to CMT Wallet without even writing any code. When the DApp is loaded by the DApp browser, we will inject a web3-cmt.js, so the DApp does not have to have its own built-in web3-cmt.js (but you can do the same), the web3 version we are currently injecting is 0.19, You can access this global object window.web3. Dapp browser will set web3.cmt.defaultAccount The value of the user is the current wallet address of the user, and the web3 HttpProvider is set to the same as the node configuration of the CMT Wallet.

1.7.3 API

web3.cmt.sendTransaction For DApp, the most common operation is to send a transaction, usually calling the web3.cmt.sendTransaction method of web3.js, DApp browser will listen to this method call, display a modal layer to let the user see the transaction information. The user can enter the password signature and then send the transaction. After the transaction is successful, it will return a txHash. If it fails, it will return the error value.

Common web3 api:

- Check the current active account on (web3.cmt.coinbase)
- Get the balance of any account (web3.cmt.getBalance)
- Send a transaction (web3.cmt.sendTransaction)
- Sign the message with the private key of the current account (web3.personal.sign)

Error handling

The DApp browser only handles some errors (such as the user entering the wrong password), most of the transaction errors will be returned to the DApp, DApps should handle these errors and prompt the user. We have done i18n processing of the error content, most of the time You can pop up error.message directly.

The user cancels the operation and the Dapp browser returns the error code "1001"

window.cmtwallet.closeDapp() Close the current DApp page and return to the discovery page

window.cmtwallet.getCurrentLanguage() Get the user's current language settings. This information may be useful if the DApp needs to support multiple languages, but we have added the locale parameter to the DApp url. In most cases you don't need to call this API. Available locale: zh-CN en-US

window.cmtwallet.getSdkVersion() Get the current CMT Wallet Dapp SDK version number: 1

window.cmtwallet.getPlatform() Get the current CMT Wallet phone system: android ios

window.cmtwallet.getAssetTokens() Get current address asset token list,return json array data

CMT Code Example

```
Copy CMT Code and goto CMT Wallet App to Open Red Packet!
cmtwallet://dapp?url=http://www.cmtoken.io/envelop/open.html?id=1&cmd=W1pYF
CMT Wallet Download Linkhttps://www.cybermiles.io/cmt-wallet/
```

dApp development sample process

- 1.install Metamask for CMT, switch testnet, get CMT.
- 2.go to Remix for CMT, coding&deploy contract, get contract address/ABI/Binary Codes.
- 3.coding in HTML5 and import web3-cmt functions.

- 4.test dApp and contact CMT Community.

1.7.4 Developer Toolkit

MetaMask for CMT

Remix for CMT

CMT Wallet-android(Testnet)

CMT Wallet-ios(Testnet)

CMTTracking

Get testnet CMT

---

web3-cmt.js

---

dApp SDK Example

Smart Contract source code in SDK Example

```
contract EasyMsg {
 string public msg;
 uint public age;

 function getData() public constant returns (string,uint){
    return (msg,age);
 }

 function setData(string _msg,uint _age) public {
     msg = _msg;
     age = _age;
 }

}
```

1.8 Frequently Asked Questions (FAQ)

In this document, we will list the frequently asked technical and general questions and answers. If you can't find you question in this FAQ, create an issue on CyberMiles.

1.8.1 Technical

1. Hardware Requirement for Validator

Operating System:

Ubuntu 16.04 LTS/Centos 7, docker is not recommended for production environment due to performance monitoring and debug complexity.

Disk Space:

Minimal disk space is 500GB SSD for the first year, more disk space will be needed as applications on travis are widely applied and transactions increase. For cloud server, it is easy to increase the disk space to at most 64T without shutting down servers or restarting service.

Physical Server:

CPU: 8 cores, 16 threads Memory: 14 RAM Port: 26656, 26657 Networking: 10 Gbps

Cloud Server:

Cost:

It is hard to estimate the total expense of physical servers depended on various types of CPU and SSD and different cable services. Yearly cost estimation for 1 validator server + 2 sentry nodes + 500G x 3 in GCP is $12443.89 plus around $1000 network fee(depend on the traffic load):

2. Sentry Node

The Sentry Node Architecture(SNA) is an infrastructure example for DDoS mitigation on travis validator nodes. On the travis network, a supernode can be attacked using the Distributed Denial of Service method. The validator node has a fixed IP address and it opens a RESTful API port facing the Internet. To mitigate the issue, multiple distributed nodes (sentry nodes) are deployed in cloud environments. With the possibility of easy scaling, it is harder to make an impact on the validator node. New sentry nodes can be brought up during a DDoS attack and using the gossip network they can be integrated into the transaction flow.

3. How to check my address, transactions, and validator status ?

Mainnet explorer: https://www.cmttracking.io/

Testnet explorer: https://testnet.cmttracking.io/

4. How could I get some CMTs in testnet/mainnet ?

Mainnet CMT:

get CMTs in crypto exchanges(binance, huobi, okex etc.)

Testnet CMT:

use testnet faucet to get test CMTs. http://travis-faucet.cybermiles.io/index.html

5. What's the chainId for testnet and mainnet ?

Mainnet chainId: 18

Testnet chainId: 19

6. How to access testnet/mainnet ?

Travis cli

```
travis attach http://{your_host}:8545
```

RPC hosts

- mainnet: rpc.cybermiles.io 8545
- testnet: testnet.cmtwallet.io 8545

RPC developer doc: https://travis.readthedocs.io/en/latest/json-rpc.html

7. How to open 8545 port?

Attention: You may lose your asset when 8545 port is turned on and is accessible by hackers.

set rpc = true.

```
vim $HOME/.travis/config/config.toml

# this is an example in mainnet
# at bottom of the config file
[vm]
chainid = 18
rpc = true
rpcapi = "cmt,eth,net,web3"
rpcaddr = "0.0.0.0"
rpcport = 8545
rpccorsdomain = "*"
rpcvhosts = "localhost"
ws = false
verbosity = "3"
```

8. How to use RPC ?

https://travis.readthedocs.io/en/latest/json-rpc.html

9. How to calculate validator's voting power or delegator's rewards ?

English version(page 10 ~ 21): https://www.cybermiles.io/wp-content/uploads/2018/10/EN_CyberMiles_DPoS_1.4.2.pdf

(8 ~ 17): https://www.cybermiles.io/wp-content/uploads/2018/10/CN_CyberMiles_DPoS.pdf

10. How to recover/re-sync my node when it crashed or missed too many blocks?

use snapshot to catch up the block data quickly. https://travis.readthedocs.io/en/latest/connect-mainnet.html#snapshot

11. When did CyberMiles transfer CMT from Ethereum and what's the height?

Ethereum Transaction Hash: https://etherscan.io/tx/0x5ff71c1be6ee3512bb574eb900e0954041031d3e0b7e544535ca7c95c8f4ccf0

Height: 6486489 (https://etherscan.io/block/6486489)

Official CMT Contract address in Ethereum: https://etherscan.io/address/0xf85feea2fdd81d51177f6b8f35f0e6734ce45f5f

12. How to update a new address of existing validator? Does this action affect all the delegators of this validator?

For validator, use web3-cmt.js or rpc to call updateAccount method to update your validator address. follow the command and example in https://cybermiles.github.io/web3-cmt.js/api/validator.html#updateaccount. It will affect your delegators.

But if your validator misses more than 120 blocks when you try to update your address, your validator node will be slashed and all the delegators on this node will be affeced. If so, manual activate your node is needed(https://cybermiles.github.io/web3-cmt.js/api/validator.html#activate).