BitcoinZ - Insight REST API

The BitcoinZ Insight REST API , is a compiled suite of open-source tools allowing for distributed applications to be ran anywhere in the world using a network of Decentralized Insight developers node. With this tool, businesses, researchers, and application developers can seamlessly integrate applications into the BitcoinZ network infrastructure using an Insight Developers node as an out-of-the-box ready integrator through it's RESTful API.

Prerequisites

Quick Install

Synchronization

The initial synchronization process scans the blockchain from the paired Zcashd server to update addresses and balances. insight-api needs exactly one trusted Zcashd node to run. This node must have finished downloading the blockchain before running insight-api.

While insight is synchronizing the website can be accessed (the sync process is embedded in the webserver), but there may be missing data or incorrect balances for addresses. The 'sync' status is shown at the `/api/sync` endpoint.
The blockchain can be read from Zcashd's raw `.dat` files or RPC interface. Reading the information from the `.dat` files is much faster so it's the recommended (and default) alternative. `.dat` files are scanned in the default location for each platform (for example, `~/.bitcoinZ` on Linux). In case a non-standard location is used, it needs to be defined (see the Configuration section). As of June 2014, using `.dat` files the sync process takes 9 hrs. for livenet and 30 mins. for testnet.
While synchronizing the blockchain, insight-api listens for new blocks and transactions relayed by the Zcashd node. Those are also stored on insight-api's database. In case insight-api is shutdown for a period of time, restarting it will trigger a partial (historic) synchronization of the blockchain. Depending on the size of that synchronization task, a reverse RPC or forward `.dat` syncing strategy will be used.
If Zcashd is shutdown, insight-api needs to be stopped and restarted once Zcashd is restarted

BitcoinZ - Insight REST API

DB storage requirement

To store the blockchain and address related information, insight-api uses LevelDB. Two DBs are created: txs and blocks. By default these are stored on

`~/.insight/`

Please note that some older versions of Insight-API store that on `<insight's root>/db`.
This can be changed at config/config.js. As of June 2014, storing the livenet blockchain takes ~35GB of disk space (2GB for the testnet).

Development

API

By default, insight provides a REST API at `/api`, but this prefix is configurable from the var `apiPrefix`in the `config.js` file.
The end-points are:

Block

```
/api/block/[:hash]
/api/block/0007844681f84249ad7829f9673ea4b6d26a139c741c5847926aff944337d908
```

Block index

Get block hash by height

```
/api/block-index/[:height]
/api/block-index/0
```

This would return:

```
{"blockHash":"f499ee3d498b4298ac6a64205b8addb7c43197e2a660229be65db8a4534d75c1"}
```

which is the hash of the Genesis block (0 height)

Transaction

```
/api/tx/[:txid]
/api/tx/0a70d135d95a190435be8cc41ea02cfce61a9c9e39cd8e7515a50d52e49c57d7
```

BitcoinZ - Insight REST API

Address

`/api/addr/[:addr][?noTxList=1&noCache=1]`
`/api/addr/t1Y4QACu5S6udREAEjNZgEFWLT6TASncfcL?noTxList=1`
will return the wallet data and only the last txid associated with the address.

`/api/addr/[:addr]`
`/api/addr/t1Y4QACu5S6udREAEjNZgEFWLT6TASncfcL`
will return the wallet data and all txid's associated with the address

Address Properties

`/api/addr/[:addr]/balance`
`/api/addr/[:addr]/totalReceived`
`/api/addr/[:addr]/totalSent`
`/api/addr/[:addr]/unconfirmedBalance`
The response contains the value in Satoshis. ### Unspent Outputs

`/api/addr/[:addr]/utxo[?noCache=1]`
Sample return:


```
[
    {
        "address": "t1Y4QACu5S6udREAEjNZgEFWLT6TASncfcL",
        "txid": "dbfdc2a0d22a8282c4e7be0452d595695f3a39173bed4f48e590877382b112fc",
        "vout": 0,
        "ts": 1401276201,
        "scriptPubKey": "76a914e50575162795cd77366fb80d728e3216bd52deac88ac",
        "amount": 0.001,
        "confirmations": 3
    },
    {
        "address": "t1Y4QACu5S6udREAEjNZgEFWLT6TASncfcL",
        "txid": "e2b82af55d64f12fd0dd075d0922ee7d6a300f58fe60a23cbb5831b31d1d58b4",
        "vout": 0,
        "ts": 1401226410,
        "scriptPubKey": "76a914e50575162795cd77366fb80d728e3216bd52deac88ac",
        "amount": 0.001,
        "confirmation": 6,
        "confirmationsFromCache": true
    }
]
```

Please note that in case confirmations are cached (which happens by default when the number of confirmations is bigger that INSIGHT_SAFE_CONFIRMATIONS) the response will include the pair

confirmationsFromCache:true, and confirmations will equal INSIGHT_SAFE_CONFIRMATIONS. See noCache and INSIGHT_IGNORE_CACHE options for details.

Unspent Outputs for multiple addresses

GET method:

```
/api/addrs/[:addrs]/utxo
/api/addrs/2NF2baYuJAkCKo5onjUKEPdARQkZ6SYyKd5,2NAre8sX2povnjy4aeiHKeEh97Qhn97tB1f/utxo
```

POST method:

```
/api/addrs/utxo
```

POST params:

```
addrs: 2NF2baYuJAkCKo5onjUKEPdARQkZ6SYyKd5,2NAre8sX2povnjy4aeiHKeEh97Qhn97tB1f
```

Transactions by Block

```
/api/txs/?block=HASH
/api/txs/?block=0007844681f84249ad7829f9673ea4b6d26a139c741c5847926aff944337d908
```

Transactions by Address

```
/api/txs/?address=addr
/api/txs/?address=t1Y4QACu5S6udREAEjNZgEFWLT6TASncfcL
```

Transactions for multiple addresses

GET method:

```
/api/addrs/[:addrs]/txs[?from=&to=]
/api/addrs/2NF2baYuJAkCKo5onjUKEPdARQkZ6SYyKd5,2NAre8sX2povnjy4aeiHKeEh97Qhn97tB1f/txs?from=0&to=20
```

POST method:

```
/api/addrs/txs
```

POST params:

```
addrs: 2NF2baYuJAkCKo5onjUKEPdARQkZ6SYyKd5,2NAre8sX2povnjy4aeiHKeEh97Qhn97tB1f
from (optional): 0
to (optional): 20
```

Sample output:

```
{ totalItems: 100,
  from: 0,
  to: 20,
  items:
```

```
    [ { txid: '3e81723d069b12983b2ef694c9782d32fca26cc978de744acbc32c3d3496e915',
        version: 1,
        locktime: 0,
        vin: [Object],
        vout: [Object],
        blockhash: '00000000011a135e5277f5493c52c66829792392632b8b65429cf07ad3c47a6c
',
        confirmations: 109367,
        time: 1393659685,
        blocktime: 1393659685,
        valueOut: 0.3453,
        size: 225,
        firstSeenTs: undefined,
        valueIn: 0.3454,
        fees: 0.0001 },
      { ... },
      { ... },
      ...
      { ... }
    ]
}
```


Note: if pagination params are not specified, the result is an array of transactions.

Transaction broadcasting

POST method:

```
/api/tx/send
```

POST params:

```
rawtx: "signed transaction as hex string"

eg

rawtx: 01000000017b1eabe0209b1fe794124575ef807057c77ada2138ae4fa8d6c4de0398a14f3
f000000004948304502210089 49f0cb400094ad2b5eb399d59d01c14d73d8fe6e96df1a7150deb38
8ab8935022079656090d7f6bac4c9a94e0aad311a4268e082a725f8aeae0573fb12ff866a5f01ffffffff
01f0ca052a010000001976a914cbc20a7664f2f69e5355aa427045bc15e7c6c77288ac00000000
```

POST response:

```
{
    txid: [:txid]
}

eg

{
    txid: "c7736a0a0046d5a8cc61c8c3c2821d4d7517f5de2bc66a966011aaa79965ffba"
}
```

BitcoinZ - Insight REST API

Historic blockchain data sync status

`/api/sync`

Live network p2p data sync status

`/api/peer`

Status of the bitcoinZ network

`/api/status?q=xxx`

Where "xxx" can be:

- •getInfo
- •getDifficulty
- •getTxOutSetInfo
- •getBestBlockHash
- •getLastBlockHash

Web Socket API

The web socket API is served using socket.io.

The following are the events published by insight:

'tx': new transaction received from network. This event is published in the 'inv' room. Data will be a app/models/Transaction object. Sample output:

```
{
  "txid":"00c1b1acb310b87085c7deaaeba478cef5dc9519fab87a4d943ecbb39bd5b053",
  "processed":false
  ...
}
```

'block': new block received from network. This event is published in the 'inv' room. Data will be a app/models/Block object. Sample output:

```
{
  "hash":"000000004a3d187c430cd6a5e988aca3b19e1f1d1727a50dead6c8ac26899b96",
  "time":1389789343,
  ...
}
```

'': new transaction concerning received from network. This event is published in the '' room.

'status': every 1% increment on the sync task, this event will be triggered. This event is published in the 'sync' room.

Sample output:

```
{
   blocksToSync: 164141,
   syncedBlocks: 475,
   upToExisting: true,
   scanningBackward: true,
   isEndGenesis: true,
   end: "000000000933ea01ad0ee984209779baaec3ced90fa3f408719526f8d77f4943",
   isStartGenesis: false,
   start: "000000009f929800556a8f3cfdbe57c187f2f679e351b12f7011bfc276c41b6d"
}
```

Example Usage

The following html page connects to the socket.io insight API and listens for new transactions.

html

```
<html>
<body>
   <script src="http://<insight-server>:<port>/socket.io/socket.io.js"></script>
   <script>
      eventToListenTo = 'tx'
      room = 'inv'

      var socket = io("http://<insight-server>:<port>/");
      socket.on('connect', function() {
         // Join the room.
         socket.emit('subscribe', room);
      })
      socket.on(eventToListenTo, function(data) {
         console.log("New transaction received: " + data.txid)
      })
   </script>
</body>
</html>
```

License

(The MIT License)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the 'Software'), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sub-license,

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED 'AS IS', WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Street Cred

A big thanks to the folks over at NPM for putting together the core Bitcoin framework of this document. Please see https://www.npmjs.com/package/insight-bitcore-api for the original document.