Introduction

In order for your Web3 application to interact with ETH (by reading blockchain data or sending transactions to the network) - it must be connected to an ETH node. Therefore, regardless of the specific node or client implementation, applications can rely on a unified set of methods, namely: JSON-RPC.

BNS (Blockchain Node Service) is a service that provides full-node RPC call capabilities based on blockchain nodes for all web3 customers. BNS helps customers achieve full node hosting (customers no longer need to send requests to their own local nodes, but only need to send requests to the provider that provides the same API - BNS, to access the information on the blockchain), thereby exempting customers from The cost of node operation and maintenance, time-consuming development, and services are unavailable/scalable.

Authentication

To call the BNS apis, you need to use the apikey and secretkey issued by the platform for authentication；Currently, apikey and secretkey only support background issuance, and you need to provide your Alibaba Cloud uid information

Examples

In this section we provide examples of how to use individual JSON_RPC API endpoints using the command line tool, curl.

```
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"web3_clientVersion","params":[],"id":67}'
```

Ehereum JSON-RPC API Methods

web3_clientVersion

Returns the current client version.

Parameters

None

Returns

`String` - The current client version

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"web3_clientVersion","params":[],"id":67}'
// Result
{
  "id":67,
  "jsonrpc":"2.0",
  "result": "Mist/v0.9.3/darwin/go1.4.1"
}
```

web3_sha3

Returns Keccak-256 (*not* the standardized SHA3-256) of the given data.

Parameters

1. `DATA` - the data to convert into a SHA3 hash

```
params: ["0x68656c6c6f20776f726c64"]
```

Returns

`DATA` - The SHA3 result of the given string.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"web3_sha3","params":
["0x68656c6c6f20776f726c64"],"id":64}'
// Result
{
  "id":64,
  "jsonrpc": "2.0",
  "result":
"0x47173285a8d7341e5e972fc677286384f802f8ef42a5ec5f03bbfa254cb01fad"
}
```

net_version

Returns the current network id.

Parameters

None

Returns

`String` - The current network id.

The full list of current network IDs is available at chainlist.org. Some common ones are:

- `1`: Ethereum Mainnet
- `5`: Goerli testnet
- `11155111`: Sepolia testnet

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"net_version","params":[],"id":67}'
// Result
{
  "id":67,
  "jsonrpc": "2.0",
  "result": "3"
}
```

net_listening

Returns `true` if client is actively listening for network connections.

Parameters

None

Returns

`Boolean` - `true` when listening, otherwise `false`.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"net_listening","params":[],"id":67}'
// Result
{
  "id":67,
  "jsonrpc":"2.0",
  "result":true
}
```

net_peerCount

Returns number of peers currently connected to the client.

Parameters

None

Returns

`QUANTITY` - integer of the number of connected peers.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"net_peerCount","params":[],"id":74}'
// Result
{
  "id":74,
  "jsonrpc": "2.0",
  "result": "0x2" // 2
}
```

eth_protocolVersion

Returns the current Ethereum protocol version. Note that this method is not available in Geth.

Parameters

None

Returns

`String` - The current Ethereum protocol version

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_protocolVersion","params":[],"id":67}'
// Result
{
  "id":67,
  "jsonrpc": "2.0",
  "result": "54"
}
```

eth_syncing

Returns an object with data about the sync status or `false`.

Parameters

None

Returns

`Object|Boolean`, An object with sync status data or `FALSE`, when not syncing:

- `startingBlock`: `QUANTITY` - The block at which the import started (will only be reset, after the sync reached his head)
- `currentBlock`: `QUANTITY` - The current block, same as eth_blockNumber
- `highestBlock`: `QUANTITY` - The estimated highest block

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_syncing","params":[],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": {
    startingBlock: '0x384',
    currentBlock: '0x386',
    highestBlock: '0x454'
  }
}
// Or when not syncing
{
  "id":1,
  "jsonrpc": "2.0",
  "result": false
}
```

eth_coinbase

Returns the client coinbase address.

Parameters

None

Returns

`DATA`, 20 bytes - the current coinbase address.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_coinbase","params":[],"id":64}'
// Result
{
  "id":64,
  "jsonrpc": "2.0",
  "result": "0x407d73d8a49eeb85d32cf465507dd71d507100c1"
}
```

eth_chainId

Returns the chain ID used for signing replay-protected transactions.

Parameters

None

Returns

`chainId`, hexadecimal value as a string representing the integer of the current chain id.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_chainId","params":[],"id":67}'
// Result
{
  "id":67,
  "jsonrpc": "2.0",
  "result": "0x1"
}
```

eth_mining

Returns `true` if client is actively mining new blocks.

Parameters

None

Returns

`Boolean` - returns `true` of the client is mining, otherwise `false`.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_mining","params":[],"id":71}'
//
{
  "id":71,
  "jsonrpc": "2.0",
  "result": true
}
```

eth_hashrate

Returns the number of hashes per second that the node is mining with.

Parameters

None

Returns

`QUANTITY` - number of hashes per second.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_hashrate","params":[],"id":71}'
// Result
{
  "id":71,
  "jsonrpc": "2.0",
  "result": "0x38a"
}
```

eth_gasPrice

Returns the current price per gas in wei.

Parameters

None

Returns

`QUANTITY` - integer of the current gas price in wei.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_gasPrice","params":[],"id":73}'
// Result
{
  "id":73,
  "jsonrpc": "2.0",
  "result": "0x1dfd14000" // 8049999872 Wei
}
```

eth_accounts

Returns a list of addresses owned by client.

Parameters

None

Returns

`Array of DATA`, 20 Bytes - addresses owned by the client.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_accounts","params":[],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": ["0x407d73d8a49eeb85d32cf465507dd71d507100c1"]
}
```

eth_blockNumber

Returns the number of most recent block.

Parameters

None

Returns

`QUANTITY` - integer of the current block number the client is on.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_blockNumber","params":[],"id":83}'
// Result
{
  "id":83,
  "jsonrpc": "2.0",
  "result": "0x4b7" // 1207
}
```

eth_getBalance

Returns the balance of the account of given address.

Parameters

1. `DATA`, 20 Bytes - address to check for balance.
2. `QUANTITY|TAG` - integer block number, or the string `"latest"`, `"earliest"` or `"pending"`, see the default block parameter

```
params: ["0x407d73d8a49eeb85d32cf465507dd71d507100c1", "latest"]
```

Returns

`QUANTITY` - integer of the current balance in wei.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0x407d73d8a49eeb85d32cf465507dd71d507100c1", "latest"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x0234c8a3397aab58" // 158972490234375000
}
```

eth_getStorageAt

Returns the value from a storage position at a given address.

Parameters

1. DATA, 20 Bytes - address of the storage.
2. QUANTITY - integer of the position in the storage.
3. QUANTITY|TAG - integer block number, or the string "latest", "earliest" or "pending", see the default block parameter

Returns

DATA - the value at this storage position.

Example Calculating the correct position depends on the storage to retrieve. Consider the following contract deployed at 0x295a70b2de5e3953354a6a8344e616ed314d7251 by address 0x391694e7e0b0cce554cb130d723a9d27458f9298.

```
contract Storage {
    uint pos0;
    mapping(address => uint) pos1;
    function Storage() {
        pos0 = 1234;
        pos1[msg.sender] = 5678;
    }
}
```

Retrieving the value of pos0 is straight forward:

```
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0", "method": "eth_getStorageAt", "params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251", "0x0", "latest"], "id": 1}'
localhost:8545
{"jsonrpc":"2.0","id":1,"result":"0x00000000000000000000000000000000000000
000000000000000000000004d2"}
```

Retrieving an element of the map is harder. The position of an element in the map is calculated with:

```
keccack(LeftPad32(key, 0), LeftPad32(map position, 0))
```

This means to retrieve the storage on pos1["0x391694e7e0b0cce554cb130d723a9d27458f9298"] we need to calculate the position with:

```
keccak(
  decodeHex(
    "000000000000000000000000391694e7e0b0cce554cb130d723a9d27458f9298" +
      "0000000000000000000000000000000000000000000000000000000000000001"
  )
)
```

The geth console which comes with the web3 library can be used to make the calculation:

```
> var key =
"000000000000000000000000391694e7e0b0cce554cb130d723a9d27458f9298" +
"0000000000000000000000000000000000000000000000000000000000000001"
undefined
> web3.sha3(key, {"encoding": "hex"})
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9"
```

Now to fetch the storage:

```
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0", "method": "eth_getStorageAt", "params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251",
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9",
"latest"], "id": 1}' localhost:8545
{"jsonrpc":"2.0","id":1,"result":"0x00000000000000000000000000000000000000
0000000000000000000000162e"}
```

eth_getTransactionCount

Returns the number of transactions *sent* from an address.

Parameters

1. `DATA`, 20 Bytes - address.
2. `QUANTITY|TAG` - integer block number, or the string `"latest"`, `"earliest"` or `"pending"`, see the default block parameter

```
params: [
  "0x407d73d8a49eeb85d32cf465507dd71d507100c1",
  "latest", // state at the latest block
]
```

Returns

`QUANTITY` - integer of the number of transactions send from this address.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getTransactionCount","params":
```

```
["0x407d73d8a49eeb85d32cf465507dd71d507100c1","latest"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x1" // 1
}
```

eth_getBlockTransactionCountByHash

Returns the number of transactions in a block from a block matching the given block hash.

Parameters

1. `DATA`, 32 Bytes - hash of a block

```
params:
["0xb903239f8543d04b5dc1ba6579132b143087c68db1b2168786408fcbce568238"]
```

Returns

`QUANTITY` - integer of the number of transactions in this block.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getBlockTransactionCountByHash","params":
["0xb903239f8543d04b5dc1ba6579132b143087c68db1b2168786408fcbce568238"],"id
":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0xb" // 11
}
```

eth_getBlockTransactionCountByNumber

Returns the number of transactions in a block matching the given block number.

Parameters

1. `QUANTITY|TAG` - integer of a block number, or the string `"earliest"`, `"latest"` or `"pending"`, as in the default block parameter.

```
params: [
  "0xe8", // 232
]
```

Returns

`QUANTITY` - integer of the number of transactions in this block.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getBlockTransactionCountByNumber","params"
:["0xe8"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0xa" // 10
}
```

eth_getUncleCountByBlockHash

Returns the number of uncles in a block from a block matching the given block hash.

Parameters

1. `DATA`, 32 Bytes - hash of a block

```
params:
["0xb903239f8543d04b5dc1ba6579132b143087c68db1b2168786408fcbce568238"]
```

Returns

`QUANTITY` - integer of the number of uncles in this block.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getUncleCountByBlockHash","params":
["0xb903239f8543d04b5dc1ba6579132b143087c68db1b2168786408fcbce568238"],"id
":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x1" // 1
}
```

eth_getUncleCountByBlockNumber

Returns the number of uncles in a block from a block matching the given block number.

Parameters

1. `QUANTITY|TAG` - integer of a block number, or the string "latest", "earliest" or "pending", see the default block parameter

```
params: [
  "0xe8", // 232
]
```

Returns

`QUANTITY` - integer of the number of uncles in this block.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getUncleCountByBlockNumber","params":
["0xe8"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x1" // 1
}
```

eth_getCode

Returns code at a given address.

Parameters

1. `DATA`, 20 Bytes - address
2. `QUANTITY|TAG` - integer block number, or the string `"latest"`, `"earliest"` or `"pending"`, see the default block parameter

```
params: [
  "0xa94f5374fce5edbc8e2a8697c15331677e6ebf0b",
  "0x2", // 2
]
```

Returns

`DATA` - the code from the given address.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getCode","params":
["0xa94f5374fce5edbc8e2a8697c15331677e6ebf0b", "0x2"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
"0x600160008035811a818181146012578301005b601b6001356025565b806000526020600
0f25b600060078202905091905056"
}
```

eth_sign

The sign method calculates an Ethereum specific signature with: `sign(keccak256("\x19Ethereum Signed Message:\n" + len(message) + message)))`.

By adding a prefix to the message makes the calculated signature recognizable as an Ethereum specific signature. This prevents misuse where a malicious dapp can sign arbitrary data (e.g. transaction) and use the signature to impersonate the victim.

Note: the address to sign with must be unlocked.

Parameters

1. `DATA`, 20 Bytes - address
2. `DATA`, N Bytes - message to sign

Returns

`DATA`: Signature

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
```

```
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_sign","params":
["0x9b2055d370f73ec7d8a03e965129118dc8f5bf83", "0xdeadbeaf"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
"0xa3f20717a250c2b0b729b7e5becbff67fdaef7e0699da4de7ca5895b02a170a12d887fd
3b17bfdce3481f10bea41f45ba9f709d39ce8325427b57afcfc994cee1b"
}
```

eth_signTransaction

Signs a transaction that can be submitted to the network at a later time using with eth_sendRawTransaction.

Parameters

1. `Object` - The transaction object

- `from`: `DATA`, 20 Bytes - The address the transaction is sent from.
- `to`: `DATA`, 20 Bytes - (optional when creating new contract) The address the transaction is directed to.
- `gas`: `QUANTITY` - (optional, default: 90000) Integer of the gas provided for the transaction execution. It will return unused gas.
- `gasPrice`: `QUANTITY` - (optional, default: To-Be-Determined) Integer of the gasPrice used for each paid gas, in Wei.
- `value`: `QUANTITY` - (optional) Integer of the value sent with this transaction, in Wei.
- `data`: `DATA` - The compiled code of a contract OR the hash of the invoked method signature and encoded parameters.
- `nonce`: `QUANTITY` - (optional) Integer of a nonce. This allows to overwrite your own pending transactions that use the same nonce.

Returns

`DATA`, The signed transaction object.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"id": 1,"jsonrpc": "2.0","method": "eth_signTransaction","params":
[{"data":"0xd46e8dd67c5d32be8d46e8dd67c5d32be8058bb8eb970870f072445675058b
b8eb970870f072445675","from":
"0xb60e8dd61c5d32be8058bb8eb970870f07233155","gas": "0x76c0","gasPrice":
"0x9184e72a000","to":
"0xd46e8dd67c5d32be8058bb8eb970870f07244567","value": "0x9184e72a"}]}'
// Result
```

```
{
    "id": 1,
    "jsonrpc": "2.0",
    "result":
"0xa3f20717a250c2b0b729b7e5becbff67fdaef7e0699da4de7ca5895b02a170a12d887fd
3b17bfdce3481f10bea41f45ba9f709d39ce8325427b57afcfc994cee1b"
}
```

eth_sendTransaction

Creates new message call transaction or a contract creation, if the data field contains code.

Parameters

1. `Object` - The transaction object

- `from`: `DATA`, 20 Bytes - The address the transaction is sent from.
- `to`: `DATA`, 20 Bytes - (optional when creating new contract) The address the transaction is directed to.
- `gas`: `QUANTITY` - (optional, default: 90000) Integer of the gas provided for the transaction execution. It will return unused gas.
- `gasPrice`: `QUANTITY` - (optional, default: To-Be-Determined) Integer of the gasPrice used for each paid gas.
- `value`: `QUANTITY` - (optional) Integer of the value sent with this transaction.
- `data`: `DATA` - The compiled code of a contract OR the hash of the invoked method signature and encoded parameters.
- `nonce`: `QUANTITY` - (optional) Integer of a nonce. This allows to overwrite your own pending transactions that use the same nonce.

```
params: [
  {
    from: "0xb60e8dd61c5d32be8058bb8eb970870f07233155",
    to: "0xd46e8dd67c5d32be8058bb8eb970870f07244567",
    gas: "0x76c0", // 30400
    gasPrice: "0x9184e72a000", // 10000000000000
    value: "0x9184e72a", // 2441406250
    data:
"0xd46e8dd67c5d32be8d46e8dd67c5d32be8058bb8eb970870f072445675058bb8eb97087
0f072445675",
  },
]
```

Returns

`DATA`, 32 Bytes - the transaction hash, or the zero hash if the transaction is not yet available.

Use eth_getTransactionReceipt to get the contract address, after the transaction was mined, when you created a contract.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_sendTransaction","params":[{see
above}],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
"0xe670ec64341771606e55d6b4ca35a1a6b75ee3d5145a99d05921026d1527331"
}
```

eth_sendRawTransaction

Creates new message call transaction or a contract creation for signed transactions.

Parameters

1. DATA, The signed transaction data.

```
params: [

"0xd46e8dd67c5d32be8d46e8dd67c5d32be8058bb8eb970870f072445675058bb8eb97087
0f072445675",
]
```

Returns

DATA, 32 Bytes - the transaction hash, or the zero hash if the transaction is not yet available.

Use eth_getTransactionReceipt to get the contract address, after the transaction was mined, when you created a contract.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_sendRawTransaction","params":[{see
above}],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
```

```
"0xe670ec64341771606e55d6b4ca35a1a6b75ee3d5145a99d05921026d1527331"
}
```

eth_call

Executes a new message call immediately without creating a transaction on the block chain.

Parameters

1. `Object` - The transaction call object

- `from`: `DATA`, 20 Bytes - (optional) The address the transaction is sent from.
- `to`: `DATA`, 20 Bytes - The address the transaction is directed to.
- `gas`: `QUANTITY` - (optional) Integer of the gas provided for the transaction execution. eth_call consumes zero gas, but this parameter may be needed by some executions.
- `gasPrice`: `QUANTITY` - (optional) Integer of the gasPrice used for each paid gas
- `value`: `QUANTITY` - (optional) Integer of the value sent with this transaction
- `data`: `DATA` - (optional) Hash of the method signature and encoded parameters. For details see Ethereum Contract ABI in the Solidity documentation

2. `QUANTITY|TAG` - integer block number, or the string `"latest"`, `"earliest"` or `"pending"`, see the default block parameter

Returns

`DATA` - the return value of executed contract.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_call","params":[{see above}],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x"
}
```

eth_estimateGas

Generates and returns an estimate of how much gas is necessary to allow the transaction to complete. The transaction will not be added to the blockchain. Note that the estimate may be significantly more than the amount of gas actually used by the transaction, for a variety of reasons including EVM mechanics and node performance.

Parameters

See eth_call parameters, except that all properties are optional. If no gas limit is specified geth uses the block gas limit from the pending block as an upper bound. As a result the returned estimate might not be enough to executed the call/transaction when the amount of gas is higher than the pending block gas limit.

Returns

`QUANTITY` - the amount of gas used.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_estimateGas","params":[{see
above}],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x5208" // 21000
}
```

eth_getBlockByHash

Returns information about a block by hash.

Parameters

1. `DATA`, 32 Bytes - Hash of a block.
2. `Boolean` - If `true` it returns the full transaction objects, if `false` only the hashes of the transactions.

```
params: [
  "0xdc0818cf78f21a8e70579cb46a43643f78291264dda342ae31049421c82d21ae",
  false,
]
```

Returns

`Object` - A block object, or `null` when no block was found:

- `number`: `QUANTITY` - the block number. `null` when its pending block.
- `hash`: `DATA`, 32 Bytes - hash of the block. `null` when its pending block.
- `parentHash`: `DATA`, 32 Bytes - hash of the parent block.
- `nonce`: `DATA`, 8 Bytes - hash of the generated proof-of-work. `null` when its pending block.
- `sha3Uncles`: `DATA`, 32 Bytes - SHA3 of the uncles data in the block.
- `logsBloom`: `DATA`, 256 Bytes - the bloom filter for the logs of the block. `null` when its pending block.

- `transactionsRoot`: `DATA`, 32 Bytes - the root of the transaction trie of the block.
- `stateRoot`: `DATA`, 32 Bytes - the root of the final state trie of the block.
- `receiptsRoot`: `DATA`, 32 Bytes - the root of the receipts trie of the block.
- `miner`: `DATA`, 20 Bytes - the address of the beneficiary to whom the mining rewards were given.
- `difficulty`: `QUANTITY` - integer of the difficulty for this block.
- `totalDifficulty`: `QUANTITY` - integer of the total difficulty of the chain until this block.
- `extraData`: `DATA` - the "extra data" field of this block.
- `size`: `QUANTITY` - integer the size of this block in bytes.
- `gasLimit`: `QUANTITY` - the maximum gas allowed in this block.
- `gasUsed`: `QUANTITY` - the total used gas by all transactions in this block.
- `timestamp`: `QUANTITY` - the unix timestamp for when the block was collated.
- `transactions`: `Array` - Array of transaction objects, or 32 Bytes transaction hashes depending on the last given parameter.
- `uncles`: `Array` - Array of uncle hashes.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getBlockByHash","params":
["0xdc0818cf78f21a8e70579cb46a43643f78291264dda342ae31049421c82d21ae",
false],"id":1}'
// Result
{
{
"jsonrpc": "2.0",
"id": 1,
"result": {
    "difficulty": "0x4ea3f27bc",
    "extraData":
"0x476574682f4c5649562f76312e302e302f6c696e75782f676f312e342e32",
    "gasLimit": "0x1388",
    "gasUsed": "0x0",
    "hash":
"0xdc0818cf78f21a8e70579cb46a43643f78291264dda342ae31049421c82d21ae",
    "logsBloom":
"0x00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000",
    "miner": "0xbb7b8287f3f0a933474a79eae42cbca977791171",
    "mixHash":
"0x4fffe9ae21f1c9e15207b1f472d5bbdd68c9595d461666602f2be20daf5e7843",
    "nonce": "0x689056015818adbe",
    "number": "0x1b4",
    "parentHash":
"0xe99e022112df268087ea7eafaf4790497fd21dbeeb6bd7a1721df161a6657a54",
    "receiptsRoot":
"0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
    "sha3Uncles":
"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
    "size": "0x220",
    "stateRoot":
"0xddc8b0234c2e0cad087c8b389aa7ef01f7d79b2570bccb77ce48648aa61c904d",
    "timestamp": "0x55ba467c",
    "totalDifficulty": "0x78ed983323d",
    "transactions": [
    ],
    "transactionsRoot":
"0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
    "uncles": [
    ]
}
}
```

eth_getBlockByNumber

Returns information about a block by block number.

Parameters

1. `QUANTITY|TAG` - integer of a block number, or the string `"earliest"`, `"latest"` or `"pending"`, as in the default block parameter.
2. `Boolean` - If `true` it returns the full transaction objects, if `false` only the hashes of the transactions.

```
params: [
  "0x1b4", // 436
  true,
]
```

Returns See eth_getBlockByHash

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getBlockByNumber","params":["0x1b4",
true],"id":1}'
```

Result see eth_getBlockByHash

eth_getTransactionByHash

Returns the information about a transaction requested by transaction hash.

Parameters

1. `DATA`, 32 Bytes - hash of a transaction

```
params:
["0x88df016429689c079f3b2f6ad39fa052532c56795b733da78a91ebe6a713944b"]
```

Returns

`Object` - A transaction object, or `null` when no transaction was found:

- `blockHash`: `DATA`, 32 Bytes - hash of the block where this transaction was in. `null` when its pending.
- `blockNumber`: `QUANTITY` - block number where this transaction was in. `null` when its pending.
- `from`: `DATA`, 20 Bytes - address of the sender.
- `gas`: `QUANTITY` - gas provided by the sender.
- `gasPrice`: `QUANTITY` - gas price provided by the sender in Wei.
- `hash`: `DATA`, 32 Bytes - hash of the transaction.
- `input`: `DATA` - the data send along with the transaction.
- `nonce`: `QUANTITY` - the number of transactions made by the sender prior to this one.
- `to`: `DATA`, 20 Bytes - address of the receiver. `null` when its a contract creation transaction.
- `transactionIndex`: `QUANTITY` - integer of the transactions index position in the block. `null` when its pending.
- `value`: `QUANTITY` - value transferred in Wei.
- `v`: `QUANTITY` - ECDSA recovery id
- `r`: `QUANTITY` - ECDSA signature r
- `s`: `QUANTITY` - ECDSA signature s

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getTransactionByHash","params":
["0x88df016429689c079f3b2f6ad39fa052532c56795b733da78a91ebe6a713944b"],"id
":1}'
// Result
{
  "jsonrpc":"2.0",
  "id":1,
  "result":{

"blockHash":"0x1d59ff54b1eb26b013ce3cb5fc9dab3705b415a67127a003c3e61eb445b
b8df2",
    "blockNumber":"0x5daf3b", // 6139707
    "from":"0xa7d9ddbe1f17865597fbd27ec712455208b6b76d",
    "gas":"0xc350", // 50000
    "gasPrice":"0x4a817c800", // 20000000000

"hash":"0x88df016429689c079f3b2f6ad39fa052532c56795b733da78a91ebe6a713944b
",
    "input":"0x68656c6c6f21",
    "nonce":"0x15", // 21
    "to":"0xf02c1c8e6114b1dbe8937a39260b5b0a374432bb",
    "transactionIndex":"0x41", // 65
    "value":"0xf3dbb76162000", // 4290000000000000
    "v":"0x25", // 37

"r":"0x1b5e176d927f8e9ab405058b2d2457392da3e20f328b16ddabcebc33eaac5fea",

"s":"0x4ba69724e8f69de52f0125ad8b3c5c2cef33019bac3249e2c0a2192766d1721c"
  }
}
```

eth_getTransactionByBlockHashAndIndex

Returns information about a transaction by block hash and transaction index position.

Parameters

1. `DATA`, 32 Bytes - hash of a block.
2. `QUANTITY` - integer of the transaction index position.

```
params: [
  "0xe670ec64341771606e55d6b4ca35a1a6b75ee3d5145a99d05921026d1527331",
  "0x0", // 0
]
```

Returns See eth_getTransactionByHash

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getTransactionByBlockHashAndIndex","params
":["0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
"0x0"],"id":1}'
```

Result see eth_getTransactionByHash

eth_getTransactionByBlockNumberAndIndex

Returns information about a transaction by block number and transaction index position.

Parameters

1. `QUANTITY|TAG` - a block number, or the string `"earliest"`, `"latest"` or `"pending"`, as in the default block parameter.
2. `QUANTITY` - the transaction index position.

```
params: [
  "0x29c", // 668
  "0x0", // 0
]
```

Returns See eth_getTransactionByHash

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getTransactionByBlockNumberAndIndex","para
ms":["0x29c", "0x0"],"id":1}'
```

Result see eth_getTransactionByHash

eth_getTransactionReceipt

Returns the receipt of a transaction by transaction hash.

Note That the receipt is not available for pending transactions.

Parameters

1. `DATA`, 32 Bytes - hash of a transaction

```
params:
["0x85d995eba9763907fdf35cd2034144dd9d53ce32cbec21349d4b12823c6860c5"]
```

Returns `Object` - A transaction receipt object, or `null` when no receipt was found:

- `transactionHash` : `DATA`, 32 Bytes - hash of the transaction.
- `transactionIndex`: `QUANTITY` - integer of the transactions index position in the block.
- `blockHash`: `DATA`, 32 Bytes - hash of the block where this transaction was in.
- `blockNumber`: `QUANTITY` - block number where this transaction was in.
- `from`: `DATA`, 20 Bytes - address of the sender.
- `to`: `DATA`, 20 Bytes - address of the receiver. null when its a contract creation transaction.
- `cumulativeGasUsed` : `QUANTITY` - The total amount of gas used when this transaction was executed in the block.

- `effectiveGasPrice` : `QUANTITY` - The sum of the base fee and tip paid per unit of gas.
- `gasUsed` : `QUANTITY` - The amount of gas used by this specific transaction alone.
- `contractAddress` : `DATA`, 20 Bytes - The contract address created, if the transaction was a contract creation, otherwise `null`.
- `logs`: `Array` - Array of log objects, which this transaction generated.
- `logsBloom`: `DATA`, 256 Bytes - Bloom filter for light clients to quickly retrieve related logs.
- `type`: `QUANTITY` - integer of the transaction type, `0x0` for legacy transactions, `0x1` for access list types, `0x2` for dynamic fees. It also returns *either* :
- `root` : `DATA` 32 bytes of post-transaction stateroot (pre Byzantium)
- `status`: `QUANTITY` either `1` (success) or `0` (failure)

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getTransactionReceipt","params":
["0x85d995eba9763907fdf35cd2034144dd9d53ce32cbec21349d4b12823c6860c5"],"id
":1}'
// Result
{
  "jsonrpc": "2.0",
  "id": 1,
  "result": {
    "blockHash":

"0xa957d47df264a31badc3ae823e10ac1d444b098d9b73d204c40426e57f47e8c3",
    "blockNumber": "0xeff35f",
    "contractAddress": null, // string of the address if it was created
    "cumulativeGasUsed": "0xa12515",
    "effectiveGasPrice": "0x5a9c688d4",
    "from": "0x6221a9c005f6e47eb398fd867784cacfdcfff4e7",
    "gasUsed": "0xb4c8",
    "logs": [{
      // logs as returned by getFilterLogs, etc.
    }],
    "logsBloom": "0x00...0", // 256 byte bloom filter
    "status": "0x1",
    "to": "0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2",
    "transactionHash":

"0x85d995eba9763907fdf35cd2034144dd9d53ce32cbec21349d4b12823c6860c5",
    "transactionIndex": "0x66",
    "type": "0x2"
  }
}
```

eth_getUncleByBlockHashAndIndex

Returns information about a uncle of a block by hash and uncle index position.

Parameters

1. DATA, 32 Bytes - The hash of a block.
2. QUANTITY - The uncle's index position.

```
params: [
  "0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
  "0x0", // 0
]
```

Returns See eth_getBlockByHash

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getUncleByBlockHashAndIndex","params":
["0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
"0x0"],"id":1}'
```

Result see eth_getBlockByHash

Note: An uncle doesn't contain individual transactions.

eth_getUncleByBlockNumberAndIndex

Returns information about a uncle of a block by number and uncle index position.

Parameters

1. QUANTITY|TAG - a block number, or the string "earliest", "latest" or "pending", as in the default block parameter.
2. QUANTITY - the uncle's index position.

```
params: [
  "0x29c", // 668
  "0x0", // 0
]
```

Returns See eth_getBlockByHash

Note: An uncle doesn't contain individual transactions.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getUncleByBlockNumberAndIndex","params":
["0x29c", "0x0"],"id":1}'
```

Result see eth_getBlockByHash

eth_getCompilers

Returns a list of available compilers in the client.

Parameters None

Returns `Array` - Array of available compilers.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getCompilers","params":[],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": ["solidity", "lll", "serpent"]
}
```

eth_compileSolidity

Returns compiled solidity code.

Parameters

1. `String` - The source code.

```
params: [
  "contract test { function multiply(uint a) returns(uint d) {   return a
* 7;   } }",
]
```

Returns `DATA` - The compiled source code.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_compileSolidity","params":["contract test
{ function multiply(uint a) returns(uint d) {   return a * 7;   }
}"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": {
      "code":
"0x605880600c6000396000f3006000357c0100000000000000000000000000000000000000
0000000000000000000090048063c6888fa114602e57005b603d60048035906020015060475
65b8060005260206000f35b60006007820290506053565b91905056",
      "info": {
        "source": "contract test {\n   function multiply(uint a) constant
returns(uint d) {\n       return a * 7;\n   }\n}\n",
        "language": "Solidity",
        "languageVersion": "0",
        "compilerVersion": "0.9.19",
        "abiDefinition": [
          {
            "constant": true,
            "inputs": [
              {
                "name": "a",
                "type": "uint256"
              }
            ],
            "name": "multiply",
            "outputs": [
              {
                "name": "d",
                "type": "uint256"
              }
            ],
            "type": "function"
          }
        ],
        "userDoc": {
          "methods": {}
        },
        "developerDoc": {
          "methods": {}
        }
      }
}
```

eth_compileLLL

Returns compiled LLL code.

Parameters

1. `String` - The source code.

```
params: ["(returnlll (suicide (caller)))"]
```

Returns `DATA` - The compiled source code.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_compileLLL","params":["(returnlll (suicide
(caller)))"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
"0x603880600c6000396000f3006001600060e060020a600035048063c6888fa1146018570
05b6021600435602b565b8060005260206000f35b600081600702905091905056" // the
compiled source code
}
```

eth_compileSerpent

Returns compiled serpent code.

Parameters

1. `String` - The source code.

```
params: ["/* some serpent */"]
```

Returns `DATA` - The compiled source code.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_compileSerpent","params":["/* some serpent
*/"],"id":1}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result":
"0x603880600c6000396000f3006001600060e060020a600035048063c6888fa1146018570
05b6021600435602b565b8060005260206000f35b600081600702905091905056" // the
compiled source code
}
```

eth_newFilter

Creates a filter object, based on filter options, to notify when the state changes (logs). To check if the state has changed, call eth_getFilterChanges.

A note on specifying topic filters: Topics are order-dependent. A transaction with a log with topics [A, B] will be matched by the following topic filters:

- `[]` "anything"
- `[A]` "A in first position (and anything after)"
- `[null, B]` "anything in first position AND B in second position (and anything after)"
- `[A, B]` "A in first position AND B in second position (and anything after)"
- `[[A, B], [A, B]]` "(A OR B) in first position AND (A OR B) in second position (and anything after)"
- **Parameters**

1. `Object` - The filter options:

- `fromBlock`: `QUANTITY|TAG` - (optional, default: `"latest"`) Integer block number, or `"latest"` for the last mined block or `"pending"`, `"earliest"` for not yet mined transactions.
- `toBlock`: `QUANTITY|TAG` - (optional, default: `"latest"`) Integer block number, or `"latest"` for the last mined block or `"pending"`, `"earliest"` for not yet mined transactions.
- `address`: `DATA|Array`, 20 Bytes - (optional) Contract address or a list of addresses from which logs should originate.
- `topics`: `Array of DATA`, - (optional) Array of 32 Bytes `DATA` topics. Topics are order-dependent. Each topic can also be an array of DATA with "or" options.

```
params: [
  {
    fromBlock: "0x1",
    toBlock: "0x2",
    address: "0x8888f1f195afa192cfee860698584c030f4c9db1",
    topics: [

"0x000000000000000000000000a94f5374fce5edbc8e2a8697c15331677e6ebf0b",
      null,
      [

"0x000000000000000000000000a94f5374fce5edbc8e2a8697c15331677e6ebf0b",

"0x0000000000000000000000000aff3454fce5edbc8cca8697c15331677e6ebccc",
        ],
      ],
    },
  ]
```

Returns `QUANTITY` - A filter id.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_newFilter","params":[{"topics":
["0x12341234"]}],"id":73}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": "0x1" // 1
}
```

eth_newBlockFilter

Creates a filter in the node, to notify when a new block arrives. To check if the state has changed, call eth_getFilterChanges.

Parameters None

Returns `QUANTITY` - A filter id.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_newBlockFilter","params":[],"id":73}'
// Result
{
  "id":1,
  "jsonrpc":  "2.0",
  "result": "0x1" // 1
}
```

eth_newPendingTransactionFilter

Creates a filter in the node, to notify when new pending transactions arrive. To check if the state has changed, call eth_getFilterChanges.

Parameters None

Returns `QUANTITY` - A filter id.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_newPendingTransactionFilter","params":
[],"id":73}'
// Result
{
  "id":1,
  "jsonrpc":  "2.0",
  "result": "0x1" // 1
}
```

eth_uninstallFilter

Uninstalls a filter with given id. Should always be called when watch is no longer needed. Additionally Filters timeout when they aren't requested with eth_getFilterChanges for a period of time.

Parameters

1. `QUANTITY` - The filter id.

```
params: [
  "0xb", // 11
]
```

Returns `Boolean` - `true` if the filter was successfully uninstalled, otherwise `false`.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_uninstallFilter","params":
["0xb"],"id":73}'
// Result
{
  "id":1,
  "jsonrpc": "2.0",
  "result": true
}
```

eth_getFilterChanges

Polling method for a filter, which returns an array of logs which occurred since last poll.

Parameters

1. `QUANTITY` - the filter id.

```
params: [
  "0x16", // 22
]
```

Returns `Array` - Array of log objects, or an empty array if nothing has changed since last poll.

- For filters created with `eth_newBlockFilter` the return are block hashes (`DATA`, 32 Bytes), e.g. `["0x3454645634534..."]`.
- For filters created with `eth_newPendingTransactionFilter` the return are transaction hashes (`DATA`, 32 Bytes), e.g. `["0x6345343454645..."]`.
- For filters created with `eth_newFilter` logs are objects with following params:
  - `removed`: `TAG` - `true` when the log was removed, due to a chain reorganization. `false` if its a valid log.
  - `logIndex`: `QUANTITY` - integer of the log index position in the block. `null` when its pending log.
  - `transactionIndex`: `QUANTITY` - integer of the transactions index position log was created from. `null` when its pending log.
  - `transactionHash`: `DATA`, 32 Bytes - hash of the transactions this log was created from. `null` when its pending log.
  - `blockHash`: `DATA`, 32 Bytes - hash of the block where this log was in. `null` when its pending. `null` when its pending log.
  - `blockNumber`: `QUANTITY` - the block number where this log was in. `null` when its pending. `null` when its pending log.
  - `address`: `DATA`, 20 Bytes - address from which this log originated.
  - `data`: `DATA` - contains one or more 32 Bytes non-indexed arguments of the log.
  - `topics`: `Array of DATA` - Array of 0 to 4 32 Bytes `DATA` of indexed log arguments. (In *solidity*: The first topic is the *hash* of the signature of the event (e.g. `Deposit(address,bytes32,uint256)`), except you declared the event with the `anonymous` specifier.)
- **Example**

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":
["0x16"],"id":73}'
// Result
{
  "id":1,
  "jsonrpc":"2.0",
  "result": [{
    "logIndex": "0x1", // 1
    "blockNumber":"0x1b4", // 436
    "blockHash":
"0x8216c5785ac562ff41e2dcfdf5785ac562ff41e2dcfdf829c5a142f1fccd7d",
    "transactionHash":
"0xdf829c5a142f1fccd7d8216c5785ac562ff41e2dcfdf5785ac562ff41e2dcf",
    "transactionIndex": "0x0", // 0
    "address": "0x16c5785ac562ff41e2dcfdf829c5a142f1fccd7d",

"data":"0x0000000000000000000000000000000000000000000000000000000000000000
",
    "topics":
["0x59ebeb90bc63057b6515673c3ecf9438e5058bca0f92585014eced636878c9a5"]
    },{
      ...
    }]
}
```

eth_getFilterLogs

Returns an array of all logs matching filter with given id.

Parameters

1. QUANTITY - The filter id.

```
params: [
  "0x16", // 22
]
```

Returns See eth_getFilterChanges

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getFilterLogs","params":["0x16"],"id":74}'
```

Result see eth_getFilterChanges

eth_getLogs

Returns an array of all logs matching a given filter object.

Parameters

1. `Object` - The filter options:

- `fromBlock`: `QUANTITY|TAG` - (optional, default: `"latest"`) Integer block number, or `"latest"` for the last mined block or `"pending"`, `"earliest"` for not yet mined transactions.
- `toBlock`: `QUANTITY|TAG` - (optional, default: `"latest"`) Integer block number, or `"latest"` for the last mined block or `"pending"`, `"earliest"` for not yet mined transactions.
- `address`: `DATA|Array`, 20 Bytes - (optional) Contract address or a list of addresses from which logs should originate.
- `topics`: `Array of DATA`, - (optional) Array of 32 Bytes `DATA` topics. Topics are order-dependent. Each topic can also be an array of DATA with "or" options.
- `blockhash`: `DATA`, 32 Bytes - (optional, **future**) With the addition of EIP-234, `blockHash` will be a new filter option which restricts the logs returned to the single block with the 32-byte hash `blockHash`. Using `blockHash` is equivalent to `fromBlock` = `toBlock` = the block number with hash `blockHash`. If `blockHash` is present in the filter criteria, then neither `fromBlock` nor `toBlock` are allowed.

```
params: [
  {
    topics: [
"0x000000000000000000000000a94f5374fce5edbc8e2a8697c15331677e6ebf0b",
    ],
  },
]
```

Returns See eth_getFilterChanges

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getLogs","params":[{"topics":
["0x000000000000000000000000a94f5374fce5edbc8e2a8697c15331677e6ebf0b"]}],"
id":74}'
```

Result see eth_getFilterChanges

eth_getWork

Returns the hash of the current block, the seedHash, and the boundary condition to be met ("target").

Parameters None

Returns `Array` - Array with the following properties:

1. `DATA`, 32 Bytes - current block header pow-hash
2. `DATA`, 32 Bytes - the seed hash used for the DAG.

3. DATA, 32 Bytes - the boundary condition ("target"), 2^256 / difficulty.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0","method":"eth_getWork","params":[],"id":73}'
// Result
{
  "id":1,
  "jsonrpc":"2.0",
  "result": [

"0x1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef",

"0x5EED00000000000000000000000000005EED0000000000000000000000000000",
      "0xd1ff1c01710000000000000000000000d1ff1c01710000000000000000000000"
    ]
}
```

eth_submitWork

Used for submitting a proof-of-work solution.

Parameters

1. DATA, 8 Bytes - The nonce found (64 bits)
2. DATA, 32 Bytes - The header's pow-hash (256 bits)
3. DATA, 32 Bytes - The mix digest (256 bits)

```
params: [
  "0x0000000000000001",
  "0x1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef",
  "0xD1FE5700000000000000000000000000D1FE5700000000000000000000000000",
]
```

Returns Boolean - returns true if the provided solution is valid, otherwise false.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0", "method":"eth_submitWork", "params":
["0x0000000000000001",
"0x1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef",
"0xD1GE5700000000000000000000000000D1GE5700000000000000000000000000"],"id"
:73}'
// Result
{
  "id":73,
  "jsonrpc":"2.0",
  "result": true
}
```

eth_submitHashrate

Used for submitting mining hashrate.

Parameters

1. `Hashrate`, a hexadecimal string representation (32 bytes) of the hashrate
2. `ID`, String - A random hexadecimal(32 bytes) ID identifying the client

```
params: [
  "0x0000000000000000000000000000000000000000000000000000000000500000",
  "0x59daa26581d0acd1fce254fb7e85952f4c09d0915afd33d3886cd914bc7d283c",
]
```

Returns `Boolean` - returns `true` if submitting went through successfully and `false` otherwise.

Example

```
// Request
curl https://www.blockchainnodeservice.cc/eth/{apikey} -X POST -H
'Content-Type: application/json' -H 'X-BNS-AUTH-SECRET: {secretkey}' -d
'{"jsonrpc":"2.0", "method":"eth_submitHashrate", "params":
["0x0000000000000000000000000000000000000000000000000000000000500000",
"0x59daa26581d0acd1fce254fb7e85952f4c09d0915afd33d3886cd914bc7d283c"],"id"
:73}'
// Result
{
  "id":73,
  "jsonrpc":"2.0",
  "result": true
}
```