Web 3.0 Node Engine Service (NES)

Developer Guide

Issue 01

Date 2023-12-18

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Ethereum Node Engine

1.1 Introduction

Ethereum

Ethereum is a blockchain with a computer embedded in it. It is the foundation for building apps and organizations in a decentralized, permissionless, censorship-resistant way. PoS underlies Ethereum's **consensus mechanism**. Ethereum switched on its proof-of-stake mechanism in 2022 because it is more secure, less energy-intensive, and better for implementing new scaling solutions compared to the previous **PoW** architecture.

Learn more about **Ethereum** and **PoS** at **Github** and **Ethereum official website**.

NES can enhance the stability and privacy of your blockchain usage and development, while also boosting its overall performance. **Note that Huawei Cloud will never collect your blockchain addresses.**

NOTE

- Supported networks
  - Ethereum mainnet: HTTP/WSS
  - Goerli testnet: HTTP/WSS
  - Sepolia testnet: HTTP/WSS
- **Supported APIs at the execution layer**
- **Supported APIs at the consensus layer**

1.2 JSON-RPC API Request Examples

1.2.1 Using cURL to Send JSON-RPC API Requests

1.2.1.1 Execution Layer

Request example (with credential)

```
curl -X POST https://your-http-endpoint/your-credential \
-H 'Content-Type: application/json' \
-d '{
"jsonrpc": "2.0",
"method":  "eth_blockNumber",
"params": [],
"id": 1
}'
```

Request example (with IAM token)

```
curl -X POST -H 'X-Auth-Token:your-iam-token' https://your-http-endpoint \
-H 'Content-Type: application/json' \
-d '{
"jsonrpc": "2.0",
"method":  "eth_blockNumber",
"params": [],
"id": 1
}'
```

Response example

```
{
"jsonrpc": "2.0",
"id": 1,
"result": "00f3c34b"
}
```

1.2.1.2 Consensus Layer

Request example (with credential)

```
curl -X GET -H 'Content-Type: application/json' https://your-http-endpoint/your-credential/eth/v1/beacon/genesis
```

Request example (with IAM token)

```
curl -X GET -H 'Content-Type: application/json' -H 'X-Auth-Token:your-iam-token' https://your-http-endpoint/eth/v1/beacon/genesis
```

Response example

```
{"data":
{"genesis_time":"1606824023","genesis_validators_root":"0x4b363db94e286120d76eb905340fdd4e54bfe9f06bf33ff6cf5ad27f511bfe95","genesis_fork_version":"0x00000000"}}
```

1.3 Application Development

1.3.1 Using Web3.js to Send JSON-RPC API Requests

Request example:

```
const Web3 = require('web3');
const url = 'https://your-http-endpoint/your-credential'
const web3 = new Web3(new Web3.providers.HttpProvider(url));
web3.eth.getBlockNumber((error, blockNumber) => {
    if(!error) {
        console.log(blockNumber);
    } else {
        console.log(error);
    }
});
```

Response example:

```
{
"jsonrpc": "2.0",
"id": 1,
"result": "00f3c34b"
}
```

1.3.2 Using Ethers.js to Send JSON-RPC API Requests

Request example:

```
const ethers = require('ethers');
const url = 'https://your-http-endpoint/your-credential'
const provider = new ethers.providers.JsonRpcProvider(url)
provider.getBlockNumber((error, blockNumber) => {
    if(!error) {
        console.log(blockNumber);
    } else {
        console.log(error);
    }
});
```

Response example:

```
{
"jsonrpc": "2.0",
"id": 1,
"result": "00f3c34b"
}
```

1.4 Smart Contract Development

1.4.1 Smart Contract Introduction

A smart contract is simply a program that runs on the Ethereum blockchain. It is a collection of code (its functions) and data (its state) that resides at a specific address on the Ethereum blockchain.

Smart contracts are a type of **Ethereum account**. This means they have a balance and can be the target of transactions. However they are not controlled by a user, instead they are deployed to the network and run as programmed. User accounts can then interact with a smart contract by submitting transactions that execute a function defined on the smart contract. Smart contracts can define rules, like regular contracts, and automatically enforce them via the code. Smart contracts cannot be deleted by default, and interactions with them are irreversible.

1.5 Ethereum APIs

In order for a software application to interact with the Ethereum blockchain, either by reading blockchain data or sending transactions to the network, it must connect to an Ethereum node.

For this purpose, every **Ethereum client** implements a **JSON-RPC specification**, so there is a uniform set of methods that applications can rely on regardless of the specific node or client implementation.

JSON-RPC is a stateless, light-weight remote procedure call (RPC) protocol. It defines several data structures and the rules around their processing. It is transport agnostic in that the concepts can be used within the same process, over HTTP, or in many various message passing environments. It uses JSON (RFC 4627) as data format.

For details, see **JSON-RPC API**.

1.5.1 Dedicated Edition

1.5.1.1 Common Ethereum APIs

A handful of core JSON-RPC methods require data from the Ethereum network, and fall neatly into three main categories: Gossip, State, and History. You can use the links in these sections to jump to each method, or check **Supported Ethereum APIs** to explore the whole list of methods.

1.5.1.1.1 Gossip Methods

These methods track the head of the chain. This is how transactions make their way around the network, find their way into blocks, and how clients find out about new blocks.

- **eth_blockNumber**
- **eth_sendRawTransaction**

1.5.1.1.2 State Methods

These methods report the current state of all the data stored. The "state" is like one big shared piece of RAM, and includes account balances, contract data, and gas estimations.

- **eth_getBalance**
- **eth_getStorageAt**
- **eth_getTransactionCount**
- **eth_getCode**
- **eth_call**
- **eth_estimateGas**

1.5.1.1.3 History Methods

These methods fetch historical records of every block back to genesis. This is like one large append-only file, and includes all block headers, block bodies, uncle blocks, and transaction receipts.

- **eth_getBlockTransactionCountByHash**
- **eth_getBlockTransactionCountByNumber**
- **eth_getUncleCountByBlockHash**
- **eth_getUncleCountByBlockNumber**
- **eth_getBlockByHash**
- **eth_getBlockByNumber**

- **eth_getTransactionByHash**
- **eth_getTransactionByBlockHashAndIndex**
- **eth_getTransactionByBlockNumberAndIndex**
- **eth_getTransactionReceipt**
- **eth_getUncleByBlockHashAndIndex**
- **eth_getUncleByBlockNumberAndIndex**

1.5.1.2 Supported Ethereum APIs

1.5.1.2.1 Client APIs

Introduction of Ethereum APIs: **JSON-RPC API**

Introduction of Go-Ethereum APIs: **JSON-RPC Server**

API Method	Throughput (Time/s)		
	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
debug_traceBlock	10	20	50
debug_traceBlockByHash	10	10	50
debug_traceBlockByNumber	10	10	15
debug_traceCall	1000	4000	10000
debug_traceTransaction	50	90	300
eth_blockNumber	7000	30000	60000
eth_call	2000	12000	30000
eth_chainId	3000	20000	50000
eth_createAccessList	200	300	500
eth_estimateGas	700	1500	5000

API Method	Throughput (Time/s)		
	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
eth_feeHistory	The throughput is determined by multiplying BLOCKCOUNT and LEN(REWARDPERCENTILES). The following lists the throughputs for different multiplication results. 50: 2700 100: 2500 200: 2300 300: 2200 400: 1900 500: 1800 600: 1500 700: 1500 800: 1500 900: 1000 1000: 1000 2000: 700 3000: 600 5000: 400 10000: 200	The throughput is determined by multiplying BLOCKCOUNT and LEN(REWARDPERCENTILES). The following lists the throughputs for different multiplication results. 50: 22000 100: 20000 200: 10000 300: 9500 400: 9000 500: 8000 600: 7000 700: 6000 800: 5000 900: 5000 1000: 4000 2000: 3000 3000: 2000 5000: 1000 10000: 600	The throughput is determined by multiplying BLOCKCOUNT and LEN(REWARDPERCENTILES). The following lists the throughputs for different multiplication results. 50: 42000 100: 35000 200: 29000 300: 24000 400: 20000 500: 18000 600: 15000 700: 14000 800: 12000 900: 11000 1000: 10000 2000: 5000 3000: 3000 5000: 2000 10000: 1000
eth_gasPrice	3000	20000	40000
eth_getBalance	3000	15000	40000
eth_getBlockByHash	● 200 if a complete block object is returned ● 1500 if a complete block object is not returned	● 600 if a complete block object is returned ● 5000 if a complete block object is not returned	● 1500 if a complete block object is returned ● 16000 if a complete block object is not returned

API Method	Throughput (Time/s)		
	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
eth_getBlockByNumber	● 300 if a complete block object is returned ● 1500 if a complete block object is not returned	● 600 if a complete block object is returned ● 5000 if a complete block object is not returned	● 1500 if a complete block object is returned ● 20000 if a complete block object is not returned
eth_getBlockTransactionCountByHash	3000	1000	40000
eth_getBlockTransactionCountByNumber	3000	20000	40000
eth_getCode	1000	4000	8000
eth_getFilterChanges	400	1000	2000
eth_getFilterLogs	50	1000	2000
eth_getLogs	40	100	200
eth_getProof	1000	1000	3000
eth_getStorageAt	3000	15000	40000
eth_getTransactionByBlockHashAndIndex	3000	15000	40000
eth_getTransactionByBlockNumberAndIndex	2500	15000	40000
eth_getTransactionByHash	600	1500	4000
eth_getTransactionCount	3000	15000	40000
eth_getTransactionReceipt	500	1500	3000
eth_getUncleByBlockHashAndIndex	3000	15000	40000
eth_getUncleByBlockNumberAndIndex	3000	15000	40000
eth_getUncleCountByBlockHash	3000	15000	40000
eth_getUncleCountByBlockNumber	3000	15000	40000

API Method	Throughput (Time/s)		
	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
eth_getWork	100	2000	5500
eth_maxPriorityFeePerGas	3000	15000	40000
eth_newBlockFilter	600	800	1800
eth_newFilter	100	500	1000
eth_newPendingTransactionFilter	20	50	80
eth_sendRawTransaction	500	1000	2500
eth_subscribe	100	1000	1000
eth_syncing	3000	20000	50000
eth_uninstallFilter	500	2000	3000
eth_unsubscribe	100	1000	1000
net_listening	3000	20000	40000
net_version	3000	20000	40000
txpool_inspect	20	40	90
txpool_status	2000	8000	15000
web3_clientVersion	3000	20000	40000
web3_sha3	3000	20000	40000

1.5.1.2.2 Beacon Node APIs

Table 1-1 Beacon node APIs

API Method	Type	Description	Throughput (Time/s)		
			4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
/eth/v1/beacon/genesis	GET	Get details of the chain's genesis.	5000	10000	23000

API Method	Type	Description	Throughput (Time/s)		
			4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
/eth/v1/beacon/states/{state_id}/root	GET	Get the HashTreeRoot for a given state_id.	3000	6000	13000
/eth/v1/beacon/states/{state_id}/fork	GET	Get the fork object for a given state_id.	3000	7000	17000
/eth/v1/beacon/states/{state_id}/finality_checkpoints	GET	Get finality checkpoints for a state.	3000	7000	17000
/eth/v1/beacon/states/{state_id}/validators	GET	Get the validator information.	5	5	5
/eth/v1/beacon/states/{state_id}/validators/{validator_id}	GET	Get the information for a given validator_id.	300	600	1000
/eth/v1/beacon/states/{state_id}/validator_balances	GET	Get the validator balance.	5	5	5
/eth/v1/beacon/states/{state_id}/committees	GET	Get the committees for a state.	5	6	15
/eth/v1/beacon/states/{state_id}/sync_committees	GET	Get sync committees for a state.	1100	2800	5000
/eth/v1/beacon/headers	GET	Get block headers matching given query.	1800	4000	8000
/eth/v1/beacon/headers/{block_id}	GET	Get the block header for a given block ID.	1400	2000	6000
/eth/v2/beacon/blocks/{block_id}	GET	Get the block information for the given block ID.	50	90	300
/eth/v1/beacon/blocks/{block_id}/root	GET	Get the block root information for a given block ID.	5000	9000	22000

API Method	Type	Description	Throughput (Time/s)		
			4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
/eth/v1/beacon/blocks/{block_id}/attestations	GET	Get the attestation included in for given block ID.	300	700	1800
/eth/v1/beacon/rewards/blocks/{block_id}	GET	Get the block reward information.	90	110	120
/eth/v1/beacon/rewards/attestations/{epoch}	POST	Get the validator rewards of the corresponding epoch.	5	5	5
/eth/v1/beacon/blinded_blocks/{block_id}	GET	Get the blinded block for given block ID.	300	600	1400
/eth/v1/beacon/pool/attestations	GET	Get attestations from the operations pool.	2200	3000	4000
/eth/v1/beacon/pool/attester_slashings	GET	Get attester slashings from the operations pool.	5000	10000	23000
/eth/v1/beacon/pool/proposer_slashings	GET	Get proposer slashings from the operations pool.	6000	11000	24000
/eth/v1/beacon/pool/voluntary_exits	GET	Get SignedVuntaryExit from the operations pool.	5000	11000	24000
/eth/v1/beacon/pool/bls_to_execution_changes	GET	Get BLS to execution changes known by the node but not necessarily incorporated into any block.	6000	11000	24000
/eth/v1/builder/states/{state_id}/expected_withdrawals	GET	Get the withdrawals that are to be included for the block built on the specified state.	2000	3000	6000
/eth/v1/config/fork_schedule	GET	Get all forks.	6000	10000	21000

API Method	Type	Description	Throughput (Time/s)		
			4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
/eth/v1/config/spec	GET	Get the specification configurations used on this node.	1700	4000	9000
/eth/v1/config/deposit_contract	GET	Get the Eth1 deposit contract address and chain ID.	7000	10000	24000
/eth/v2/debug/beacon/states/{state_id}	GET	Get the full BeaconState object.	5	5	5
/eth/v2/debug/beacon/heads (Deprecated)	GET	Get all chain headers.	6000	10000	13000
/eth/v1/debug/fork_choice	GET	Get the fork choice array.	600	1000	2000
/eth/v1/events	GET	Subscribe to beacon node events.	-	-	-
/eth/v1/node/version	GET	Get the beacon node version.	5000	10000	23000
/eth/v1/node/syncing	GET	Get the beacon node syncing status.	5000	10000	23000
/eth/v1/node/health	GET	Get the health check result of the beacon node.	5000	11000	24000
/eth/v1/validator/duties/attester/{epoch}	POST	Get validator duties.	5	5	5
/eth/v1/validator/duties/proposer/{epoch}	GET	Get block proposer duties.	5	5	5
/eth/v1/validator/duties/sync/{epoch}	POST	Get sync committee duties.	5	5	5

API Method	Type	Description	Throughput (Time/s)		
			4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 32 GB
/eth/v1/validator/aggregate_attestation	GET	Get aggregated attestation.	4000	8000	15000
/eth/v1/validator/sync_committee_contribution	GET	Produce a sync committee contribution.	5000	11000	18000
/eth/v1/validator/liveness/{epoch}	POST	Return if a validator has been observed on the network.	5	5	5

1.5.2 Shared Edition

The following introduces the JSON-RPC APIs supported by the shared edition.

1.5.2.1 eth_blocknumber

Introduction

This API returns the latest block number of the blockchain. It consumes 15 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An integer value of the latest block number encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_blockNumber","params":[],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.2 eth_getBlockByNumber

Introduction

This API returns information about the block by block number. It consumes 49 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.3 eth_getUncleByBlockNumberAndIndex

Introduction

This API returns information about an uncle of a block by number and uncle index position. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Block number or tag	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.

  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockNumberAndIndex","params":["latest","0x0"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.4 eth_getUncleByBlockHashAndIndex

Introduction

This API returns information about an uncle of a block by hash and uncle index position. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.

- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockHashAndIndex","params":
["0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.5 eth_getUncleCountByBlockNumber

Introduction

This API returns the number of uncles for the block by block number. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.6 eth_getUncleCountByBlockHash

Introduction

This API returns the number of uncles for the block by block hash. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.7 eth_getBlockByHash

Introduction

This API returns information about the block by block hash. It consumes 47 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.

- totalDifficulty: the total difficulty of the chain until this block.
- extraData: the "extra data" field of this block.
- size: the size of this block in bytes.
- gasLimit: the maximum gas allowed in this block.
- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec",false],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.8 eth_getTransactionByHash

Introduction

This API returns the information about a transaction by transaction hash. It consumes 25 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
  - hash: the hash of the transaction.

- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByHash","params":
["0xb142342a7fd70602b7a0ba3688a41bfcbb4fbc3490c252ca48af2594619d220c"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.9 eth_getTransactionCount

Introduction

This API returns the number of transactions sent from an address. It consumes 26 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address from which the transaction count to be checked.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions sent from an address encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionCount","params":
["0x8D97689C9818892B700e27F316cc3E41e17fBeb9", "latest"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.10 eth_getTransactionByBlockHashAndIndex

Introduction

This API returns information about a transaction by a block hash and transaction index position. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
  - hash: the hash of the transaction.
  - input: the data sent along with the transaction.
  - nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
  - to: the address of the receiver. It is null for a contract creation transaction.
  - transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
  - value: the value transferred in wei encoded as hexadecimal.
  - type: the transaction type.
  - accessList: a list of addresses and storage keys that the transaction plans to access.

  - chainId: the transaction chain ID, if any.
  - v: the standard V field of the signature.
  - r: the R field of the signature.
  - s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockHashAndIndex","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec","0x0"],"id":1,"jsonrpc":"2.0"}
'
```

1.5.2.11 eth_getTransactionByBlockNumberAndIndex

Introduction

This API returns information about a transaction by a block number and transaction index position. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.

- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockNumberAndIndex","params":["0xc5043f",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.12 eth_getBlockTransactionCountByHash

Introduction

This API returns the number of transactions for the block by block hash. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.13 eth_getBlockTransactionCountByNumber

Introduction

This API returns the number of transactions for the block by block number. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.14 eth_getTransactionReceipt

Introduction

This API returns the receipt of a transaction by transaction hash. It consumes 15 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

- Object: A transaction receipt object with the following fields, or null when no transaction receipt was found:
  - blockHash: the hash of the block where this transaction was in.
  - blockNumber: the block number where this transaction was added encoded as a hexadecimal.
  - contractAddress: the contract address created for contract creation. It is null for a transaction that is not for contract creation.

- cumulativeGasUsed: the total gas used when this transaction was executed in the block.
- effectiveGasPrice: the total base charge plus tip paid for each unit of gas.
- from: the address of the sender.
- gasUsed: the amount of gas used by this specific transaction alone.
- logs: an array of log objects that generated this transaction.
  - address: the address from which this log was generated.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - data: the 32-byte non-indexed argument of the log.
  - blockNumber: the number of the block where this log was in.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the transactions index position from which this log was created. It is null for a pending log.
  - blockHash: the hash of the block where this log was in.
  - logIndex: the integer of log index position in the block encoded as hexadecimal. It is null for a pending log.
  - removed: true if log was removed due to a chain reorganization and false if the log is valid.
- logsBloom: the bloom filter which is used to retrieve related logs.
- status: 1 (success) or 0 (failure) encoded as a hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionHash: the hash of the transaction.
- transactionIndex: the transaction index position encoded as a hexadecimal.
- type: the value type.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionReceipt","params":
["0x6d755989f51032147484162c4dc3d6550552dbd8d3b094fe3c221bfa3c5942b2"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.15 eth_sendRawTransaction

Introduction

This API creates a new message call transaction or creates a contract for signed transactions. It consumes 250 CUs.

Parameter Description

Parameter	Type	Description
Signed transaction data	String	The transaction generated using the private key.

Return Value

The transaction hash, or the zero hash if the transaction is not yet available.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"jsonrpc":"2.0","method":"eth_sendRawTransaction","params":["signed transaction"],"id":1}'
```

1.5.2.16 eth_call

Introduction

This API executes a new message call immediately without creating a transaction on the blockchain. It consumes 30 CUs.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	(Optional) The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	(Optional) The integer of gas provided for the transaction execution.
gasPrice	Integer	(Optional) The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	(Optional) The integer of value sent with this transaction encoded as hexadecimal.

Parameter	Type	Description
data	String	(Optional) The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The return value of the executed contract method.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_call","params":
[{"from":null,"to":"0x6b175474e89094c44da98b954eedeac495271d0f","data":"0x70a0823100000000000000
00000000006E0d01A76C3Cf4288372a29124A26D4353EE51BE"}, "latest"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.17 eth_createAccessList

Introduction

This API creates an EIP-2930 type accessList based on a given Transaction object. It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles). It consumes 300 CUs.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	The integer of gas provided for the transaction execution.

Parameter	Type	Description
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles), plus the estimated gas consumed when the access list is added.

- accessList: a list of objects with the following fields:
  - address: the addresses to be accessed by the transaction.
  - storageKeys: the storage keys to be accessed by the transaction.
- gasUsed: a hexadecimal string representing the approximate gas cost for the transaction if the access list is included.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"method":"eth_createAccessList","params":[{"from":
"0xaeA8F8f781326bfE6A7683C2BD48Dd6AA4d3Ba63", "data": "0x608060806080608155"},
"pending"],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.18 eth_estimateGas

Introduction

This API returns an estimation of gas for a given transaction. It consumes 90 CUs.

Parameter Description

The parameters are the same as those of eth_call, but they are all optional. If no gas 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 execute the call/transaction when the amount of actual gas needed is higher than the pending block gas limit.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
Gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

An estimation of gas for a given transaction.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_estimateGas","params":
[{"from":"0x8D97689C9818892B700e27F316cc3E41e17fBeb9","to":"0xd3CdA913deB6f67967B99D67aCDFa1
712C293601","value":"0x186a0"}],"id":1,"jsonrpc":"2.0"}'
```

1.5.2.19 eth_feeHistory

Introduction

This API returns historical gas information. It consumes 16 CUs.

Parameter Description

Parameter	Type	Description
Number of blocks	String/Integer	Number of blocks in the requested range. 1 to 1024 blocks can be requested in a single query. Less than requested may be returned if not all blocks are available.
Newest block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Reward percentiles	Integer	(Optional) A monotonically increasing list of percentile values to sample from each block's effective priority fees per gas in ascending order, weighted by gas used.

Return Value

- oldestBlock: the lowest number block of the returned range encoded as hexadecimal.
- baseFeePerGas: an array of block base fees per gas, including an extra block value. The extra value is the next block after the newest block in the returned range. Zeroes are returned for blocks created before EIP-1559.
- gasUsedRatio: an array of block gas used ratios. These are calculated as the ratio of gasUsed and gasLimit.
- reward: an array of effective priority fees per gas data points from a single block. All zeroes are returned if the block is empty.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"id": 1, "jsonrpc": "2.0", "method": "eth_feeHistory", "params": ["0x5", "latest", [20,30]] }'
```

1.5.2.20 eth_maxPriorityFeePerGas

Introduction

This API returns a fee per gas that is an estimate of how much you can pay as a priority fee, or a tip, to get a transaction included in the current block. It consumes 16 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the priority fee needed to be included in a block.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_maxPriorityFeePerGas","id":1}'
```

1.5.2.21 eth_gasPrice

Introduction

This API returns the current gas price in wei. It consumes 19 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current gas price in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_gasPrice","params": [],"id":1}'
```

1.5.2.22 eth_getBalance

Introduction

This API returns the balance of the given account address. It consumes 23 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address to check for balance.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

A hexadecimal value of the current balance in the account at the given address, in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0xc94770007dda54cF92009BFF0dE90c06F603a09f", "latest"],"id":1}'
```

1.5.2.23 eth_subscribe

Introduction

This API creates a new subscription for particular events. The node returns a subscription ID. For each event that matches the subscription, a notification with relevant data is sent together with the subscription ID. It consumes 10 CUs.

Parameter Description

Parameter	Type	Description
Event type	String	The type of event to listen to.
Optional parameters	String	Optional parameters to include to describe the type of event to listen to (e.g. newHeads, newPendingTransactions, logs).

Return Value

While the subscription is active, you will receive events formatted as an object described below:

Event Object:

- jsonrpc: always **2.0**.
- method: always **eth_subscription**.
- params: an object with the following fields:
  - subscription: the subscription ID returned by the API that creates this subscription. This ID will be attached to all received events and can also be used to cancel the subscription using eth_unsubscribe.
  - result: an object whose contents vary depending on the event type.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_subscribe",
"params": ["logs"]}'
```

1.5.2.24 eth_unsubscribe

Introduction

This API cancels subscriptions with the subscription ID. It returns a boolean indicating that the subscription was canceled successfully. It consumes 10 CUs.

Parameter Description

Parameter	Type	Description
Subscription ID	String	The ID of the subscription you want to unsubscribe.

Return Value

true is returned if a subscription was successfully canceled, or false is returned.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_unsubscribe",
"params": ["0x9cef478923ff08bf67fde6c64013158d"]}'
```

1.5.2.25 eth_getStorageAt

Introduction

This API returns the value from a storage position at a given address. It consumes 23 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address.
Storage position	String	A hexadecimal code of the position in the storage.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns the value at this storage position.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getStorageAt","params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251",
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9", "0x65a8db"],"id":1}'
```

1.5.2.26 eth_getCode

Introduction

This API returns the compiled byte code of a smart contract, if any, at a given address. It consumes 22 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The compiled byte code of the smart contract at the given address.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getCode","params":
["0x06012c8cf97bead5deae237070f9587f8e7a266d", "0x65a8db"],"id":1}'
```

1.5.2.27 eth_getProof

Introduction

This API returns the account and storage values, including the Merkle proof, of the specified account. It consumes 43 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Storage keys	Array	An array of 32-byte storage keys to be proofed and included.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

- address: the address related to the account.
- accountProof: an array of RLP-serialized MerkleTree-Nodes, starting with the stateRoot-Node, following the path of the SHA3 (address) as key.
- balance: a hexadecimal value of the current balance in wei.
- codeHash: the 32-byte hash of the code of the account.
- nonce: the nonce of the account.
- storageHash: 32 bytes. The SHA3 of the StorageRoot. All storage will deliver a Merkle proof starting with this rootHash.
- storageProof: an array of storage-entries as requested. Each entry is an object with these properties:
  - key: the requested storage key.
  - value: the storage value.
  - proof: an array of RLP-serialized MerkleTree-Nodes, starting with the storageHash-Node, following the path of the SHA3 (key) as path.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc": "2.0","method": "eth_getProof","id": 1,"params":
["0x7F0d15C7FAae65896648C8273B6d7E43f58Fa842",
["0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421"], "latest"]}'
```

1.5.2.28 eth_getLogs

Introduction

This API returns an array of all the logs matching the given filter object. It consumes 75 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

An array of log objects, or an empty array if nothing has changed since last poll. Log objects contain the following keys and their values:

- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.

- blockHash: 32 bytes. The hash of the block where this log was in. It is null for a pending log.
- blockNumber: the block number where this log was in. It is null for a pending log.
- address: 20 bytes. The address from which this log originated.
- data: It contains one or more 32-byte non-indexed arguments of the log.
- topics: an array of 0 to 4 indexed log arguments, each 32 bytes. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getLogs","params":[{"blockHash":
"0x7c5a35e9cb3e8ae0e221ab470abae9d446c3a5626ce6689fc777dcffcab52c70", "topics":
["0x241ea03ca20251805084d27d4440371c34a0b85ff108f6bb5611248f73818b80"]}],"id":74}'
```

1.5.2.29 eth_getFilterChanges

Introduction

The polling method for a filter, which returns an array of logs which occurred since last poll. Call eth_newFilter, eth_newBlockFilter, or eth_newPendingTransactionFilter to create a filter. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

- log object array: an array of log objects, or an empty array if nothing has changed since last poll.
- For filters created with eth_newBlockFilter, the return values are block hashes (32 bytes), for example, ["0x3454645634534..."].
- For filters created with eth_newFilter, the logs are objects with the following parameters:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.

- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":73}'
```

1.5.2.30 eth_getFilterLogs

Introduction

This API returns an array of all the logs matching the given filter ID. It consumes 75 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

- Log object array: an array of log objects that match the filter. For an array of logs that occurred since the last poll, use eth_getFilterChanges. Log objects contain the following keys and their values:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
  - removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
   -X POST \
   -H "Content-Type: application/json" \
   -d '{"jsonrpc":"2.0","method":"eth_getFilterLogs","params":["0x16"],"id":74}'
```

1.5.2.31 eth_newBlockFilter

Introduction

This API creates a filter in the node to notify when a new block arrives. It consumes 20 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
   -X POST \
   -H "Content-Type: application/json" \
   -d '{"jsonrpc":"2.0","method":"eth_newBlockFilter","params":[],"id":73}'
```

1.5.2.32 eth_newFilter

Introduction

This API creates a filter object based on the given filter options to notify when the state changes (logs). It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newFilter","params":[{"topics":
["0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef"]}],"id":73}'
```

1.5.2.33 eth_newPendingTransactionFilter

Introduction

This API creates a filter in the node to notify when new pending transactions arrive at Polygon. It consumes 20 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newPendingTransactionFilter","params":[],"id":73}'
```

1.5.2.34 eth_uninstallFilter

Introduction

This API uninstalls a filter with the given filter ID. It should always be called when watching is no longer needed. Additionally, filters time out when they are not requested with eth_getFilterChanges for a period of time. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

true is returned if a filter was successfully uninstalled, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_uninstallFilter","params":["0xb"],"id":73}'
```

1.5.2.35 eth_chainId

Introduction

This API returns the currently configured chain ID. It consumes 1 CU.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current chain ID.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_chainId","params": [],"id":1}'
```

1.5.2.36 web3_sha3

Introduction

This API returns Keccak-256 (not the standardized SHA3-256) hash of the given data. It consumes 15 CUs.

Parameter Description

Parameter	Type	Description
Data	String	Data to be converted.

Return Value

The SHA3 hash of the given string.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"web3_sha3","params": ["0x68656c6c6f20776f726c64"],"id":64}'
```

1.5.2.37 web3_clientVersion

Introduction

This API returns the current version of the chain client. It consumes 15 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

The current client version.

1.5.2.38 net_version

Introduction

This API returns the current version of the network. It consumes 1 CU.

Parameter Description

This method does not accept any parameters.

Return Value

The current network version.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"net_version","params": [],"id":1}'
```

1.5.2.39 net_listening

Introduction

This API returns if the client is listening for network connections. It consumes 1 CU.

Parameter Description

This method does not accept any parameters.

Return Value

If the client is listening for network connections.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"net_listening","params": [], "id":1}'
```

2 TRON Node Engine

2.1 Introduction

TRON is an open-source public blockchain platform that supports smart contracts. TRON is compatible with Ethereum, which means that you can migrate smart contracts on Ethereum to TRON directly or with minor modifications. TRON relies on a unique consensus mechanism to realize the high TPS of the TRON network that is far beyond Ethereum, bringing developers a good experience of faster transactions.

Learn more about TRON at their **Developer Hub** and from their **Whitepaper**.

NES can enhance the stability and privacy of your blockchain usage and development, while also boosting its overall performance. **Note that Huawei Cloud will never collect your blockchain addresses.**

NOTE

- Supported networks
  - TRON mainnet: HTTP
  - Nile testnet: HTTP
- **Supported TRON APIs**

2.2 HTTP Request Examples

2.2.1 Using cURL to Send HTTP API Requests

Request example (with credential)

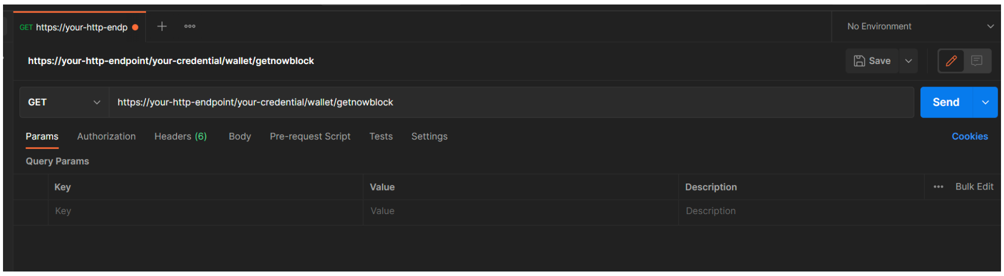
```
curl -X GET https://your-http-endpoint/your-credential/wallet/getnowblock
Request example (with IAM token)
curl -X GET 'X-Auth-Token:your-iam-token' https://your-http-endpoint/wallet/getnowblock
```

Response example

```
{
  "blockID": "000000000204b46379ebc1d66a41a816e1b8d0c3e5f917a6af5e4471288715ef",
  "block_header": {
    "raw_data": {
      "number": 33862755,
      "txTrieRoot":
"0000000000000000000000000000000000000000000000000000000000000000",
      "witness_address": "TVK7n5qks4vNrkfBnjTzEs8XWF3gbv9oNr",
      "parentHash":
"000000000204b462a2825cafbae6838e8ede6e223cc3ef52ef63ee29679123b7",
      "version": 27,
      "timestamp": 1684719705000
    },
    "witness_signature":
"dbf106c634b5e7cb2710bd93dcdd38ef0be012c197866a2c3f0ceb99cbce76073cda4af405f0e1236
214bd1cd9f4a3122c42ae94e727912a847a760b6c9ca91000"
  }
}
```

2.2.2 Using Postman to Send HTTP API Requests

Request example:



GET https://your-http-endp

No Environment

https://your-http-endpoint/your-credential/wallet/getnowblock

Save

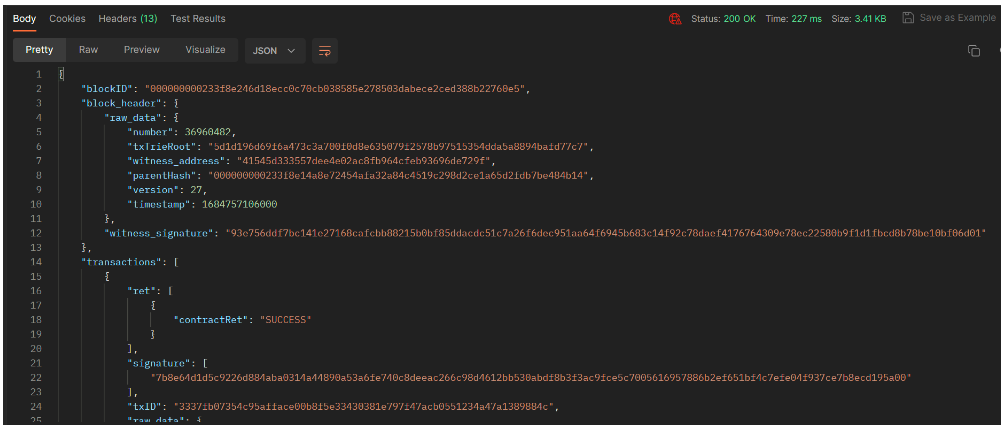
GET https://your-http-endpoint/your-credential/wallet/getnowblock Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Response example:



Body Cookies Headers (13) Test Results

Status: 200 OK Time: 227 ms Size: 3.41 KB

```
{
    "blockID": "000000000233f8e246d18ecc0c70cb038585e278503dabece2ced388b22760e5",
    "block_header": {
        "raw_data": {
            "number": 36960482,
            "txTrieRoot": "5d1d196d69f6a473c3a700f0d8e635079f2578b97515354dda5a8894bafd77c7",
            "witness_address": "41545d333557dee4e02ac8fb964cfeb93696de729f",
            "parentHash": "000000000233f8e14a8e72454afa32a84c4519c298d2ce1a65d2fdb7be484b14",
            "version": 27,
            "timestamp": 1684757106000
        },
        "witness_signature": "93e756ddf7bc141e27168cafcbb88215b0bf85ddacdc51c7a26f6dec951aa64f6945b683c14f92c78daef4176764309e78ec22580b9f1d1fbcd8b78be10bf06d01"
    },
    "transactions": [
        {
            "ret": [
                {
                    "contractRet": "SUCCESS"
                }
            ],
            "signature": [
                "7b8e64d1d5c9226d884aba0314a44890a53a6fe740c8deeac266c98d4612bb530abdf8b3f3ac9fce5c7005616957886b2ef651bf4c7efe04f937ce7b8ecd195a00"
            ],
            "txID": "3337fb07354c95afface00b8f5e33430381e797f47acb0551234a47a1389884c",
            "raw_data": {
```

2.3 JSON-RPC Request Examples

2.3.1 Using cURL to Send JSON-RPC API Requests

Request example (with credential)

curl -k -X POST https://*your-http-endpoint*/*your-credential*/jsonrpc --data
'{"jsonrpc":"2.0","method":"eth_syncing","params":[],"id":64}'
Request example (with IAM token)

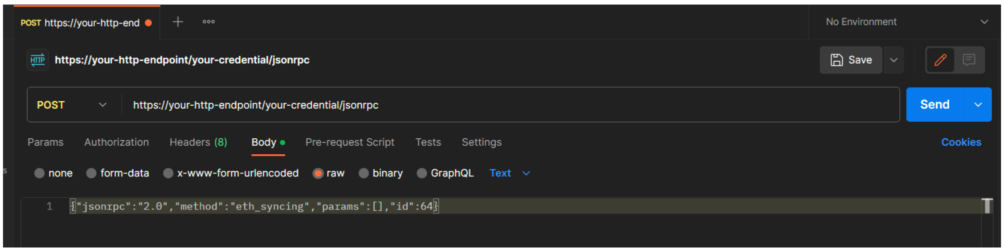
```
curl -k -X POST 'X-Auth-Token:your-iam-token' https://your-http-endpoint/jsonrpc --data
'{"jsonrpc":"2.0","method":"eth_syncing","params":[],"id":64}'
```

Response example

```
{
   "jsonrpc": "2.0",
   "id": 64,
   "result": {
      "startingBlock": "0x32e92c",
      "currentBlock": "0x32e92e",
      "highestBlock": "0x32e92e"
   }
}
```

2.3.2 Using Postman to Send JSON-RPC API Requests

Request example



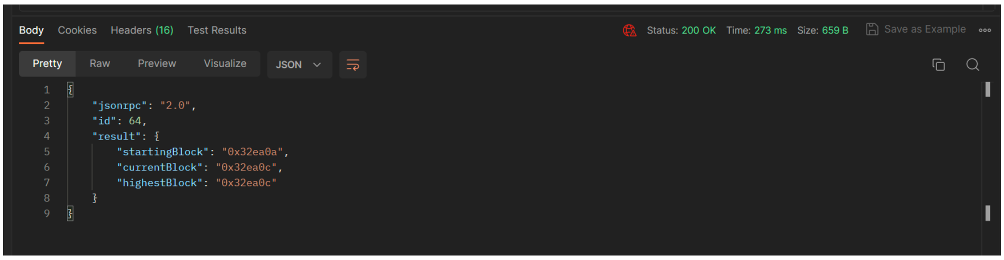
https://your-http-endpoint/your-credential/jsonrpc

POST https://your-http-endpoint/your-credential/jsonrpc

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL Text

```
{"jsonrpc":"2.0","method":"eth_syncing","params":[],"id":64}
```

Response example



Status: 200 OK Time: 273 ms Size: 659 B

```
{
    "jsonrpc": "2.0",
    "id": 64,
    "result": {
        "startingBlock": "0x32ea0a",
        "currentBlock": "0x32ea0c",
        "highestBlock": "0x32ea0c"
    }
}
```

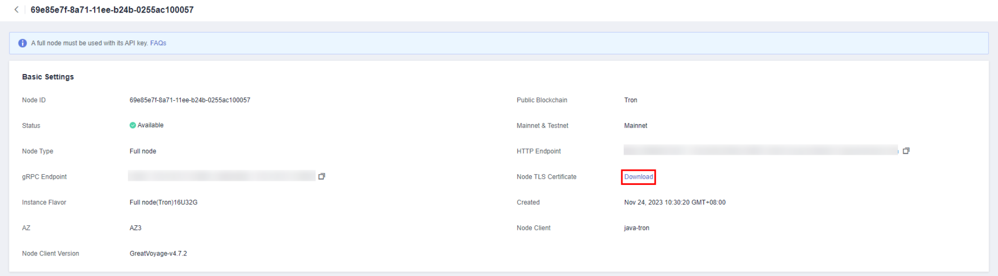
2.4 gRPC Request Examples

2.4.1 Using TridentSDK to Send gRPC Requests

⚠ CAUTION

TridentSDK of the version 0.7.0 and earlier is supported. Currently, you cannot send gRPC requests using TLS. Therefore, perform node interconnection with Java reflection according to the sample code.

Download the certificate on the node details page and place the ca.crt certificate in the package under the project directory.



< 69e85e7f-8a71-11ee-b24b-0255ac100057

A full node must be used with its API key. FAQs

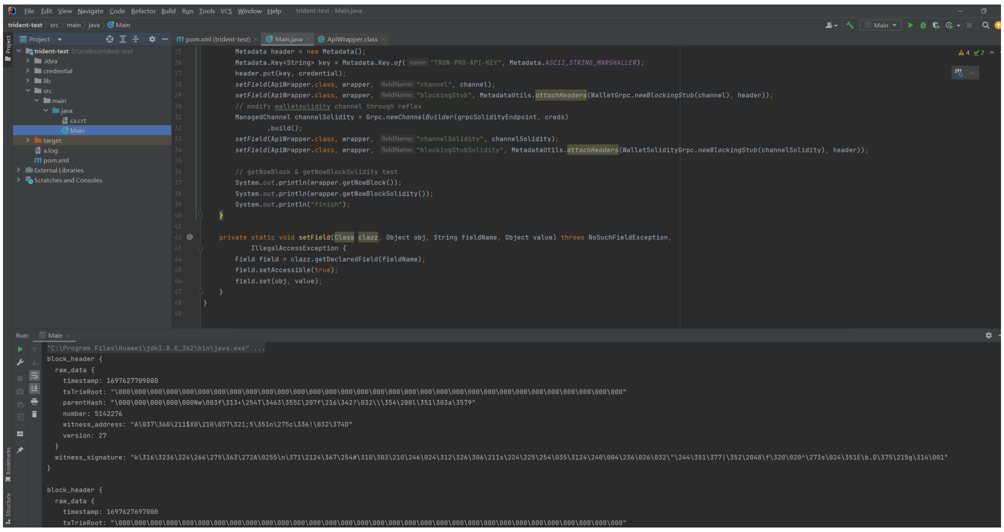
Basic Settings

Node ID	69e85e7f-8a71-11ee-b24b-0255ac100057	Public Blockchain	Tron
Status	Available	Mainnet & Testnet	Mainnet
Node Type	Full node	HTTP Endpoint	[redacted]
gRPC Endpoint	[redacted]	Node TLS Certificate	Download
Instance Flavor	Full node(Tron)16U32G	Created	Nov 24, 2023 10:30:20 GMT+08:00
AZ	AZ3	Node Client	java-tron
Node Client Version	GreatVoyage-v4.7.2		

Configure the gRPC endpoint in the SDK. The sample code is as follows:

```
import io.gRPC.*;
import io.gRPC.stub.MetadataUtils;
import org.tron.trident.api.WalletgRPC;
import org.tron.trident.api.WalletSoliditygRPC;
import org.tron.trident.core.ApiWrapper;
import org.tron.trident.core.exceptions.IllegalException;
import org.tron.trident.core.key.KeyPair;

import java.io.File;
import java.io.IOException;
import java.lang.reflect.Field;

public class Main {
    public static void main(String[] args) throws IllegalException, IOException,
NoSuchFieldException,
            IllegalAccessException {
        String gRPCEndpoint = "your-gRPC-endpoint";
        String gRPCSolidityEndpoint = "your-gRPCsolidity-endpoint";
        String credential = "your-credential";
        String hexPrivateKey = "your-hex-private-key";
        ApiWrapper wrapper = new ApiWrapper(gRPCEndpoint, gRPCSolidityEndpoint,
hexPrivateKey, credential);
        // modify wallet channel through reflex
        ChannelCredentials creds = TlsChannelCredentials.newBuilder()
                .trustManager(new File("your-ca.crt-file-path"))
                .build();
        ManagedChannel channel = gRPC.newChannelBuilder(gRPCEndpoint, creds)
                .build();
        Metadata header = new Metadata();
        Metadata.Key<String> key = Metadata.Key.of("TRON-PRO-API-KEY",
Metadata.ASCII_STRING_MARSHALLER);
        header.put(key, credential);
        setField(ApiWrapper.class, wrapper, "channel", channel);
        setField(ApiWrapper.class, wrapper, "blockingStub",
MetadataUtils.attachHeaders(WalletgRPC.newBlockingStub(channel), header));
        // modify walletsolidity channel through reflex
        ManagedChannel channelSolidity = gRPC.newChannelBuilder(gRPCSolidityEndpoint, creds)
                .build();
        setField(ApiWrapper.class, wrapper, "channelSolidity", channelSolidity);
        setField(ApiWrapper.class, wrapper, "blockingStubSolidity",
MetadataUtils.attachHeaders(WalletSoliditygRPC.newBlockingStub(channelSolidity), header));
        // getNowBlock & getNowBlockSolidity test
        System.out.println(wrapper.getNowBlock());
        System.out.println(wrapper.getNowBlockSolidity());
        System.out.println("finish");
    }
    private static void setField(Class clazz, Object obj, String fieldName, Object value) throws
NoSuchFieldException,
            IllegalAccessException {
        Field field = clazz.getDeclaredField(fieldName);
        field.setAccessible(true);
        field.set(obj, value);
    }
}
```

Response example



```
        Metadata header = new Metadata();
        Metadata.Key<String> key = Metadata.Key.of( name: "TRON-PRO-API-KEY", Metadata.ASCII_STRING_MARSHALLER);
        header.put(key, credential);
        setField(ApiWrapper.class, wrapper,  fieldName: "channel", channel);
        setField(ApiWrapper.class, wrapper,  fieldName: "blockingStub", MetadataUtils.attachHeaders(WalletGrpc.newBlockingStub(channel), header));
        // modify walletsolidity channel through reflex
        ManagedChannel channelSolidity = Grpc.newChannelBuilder(grpcSolidityEndpoint, creds)
                .build();
        setField(ApiWrapper.class, wrapper,  fieldName: "channelSolidity", channelSolidity);
        setField(ApiWrapper.class, wrapper,  fieldName: "blockingStubSolidity", MetadataUtils.attachHeaders(WalletSolidityGrpc.newBlockingStub(channelSolidity), header));

        // getNowBlock & getNowBlockSolidity test
        System.out.println(wrapper.getNowBlock());
        System.out.println(wrapper.getNowBlockSolidity());
        System.out.println("finish");
    }

    private static void setField(Class clazz, Object obj, String fieldName, Object value) throws NoSuchFieldException,
            IllegalAccessException {
        Field field = clazz.getDeclaredField(fieldName);
        field.setAccessible(true);
        field.set(obj, value);
    }
}
```

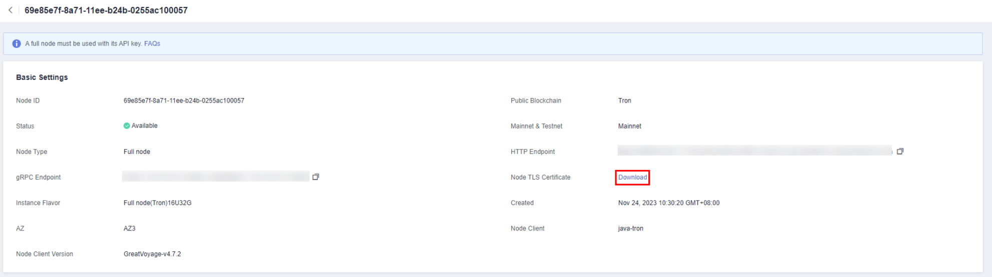
```
"C:\Program Files\Huawei\jdk1.8.0_362\bin\java.exe" ...
block_header {
  raw_data {
    timestamp: 1697627709000
    txTrieRoot: "\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000"
    parentHash: "\000\000\000\000\000Nw\003f\313+\254T\3463\355C\207f\216\342?\032\\\354\200l\351\303a\3579"
    number: 5142276
    witness_address: "A\037\360\211$X0\210\037\321;5\351n\275c\336!\032\374D"
    version: 27
  }
  witness_signature: "k\316\3236\324\266\275\363\272A\0255\n\371\2124\367\254#\310\303\210\246\024\312\326\306\211s\224\225\254\035\3124\240\004\236\026\032\"\244\351\377|\352\2048\f\320\020^\273s\024\351E\b.D\375\215g\314\001"
}

block_header {
  raw_data {
    timestamp: 1697627697000
    txTrieRoot: "\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000\000"
```

2.4.2 Using gotron-sdk to Send gRPC Requests

CAUTION

gotron-sdk of the version 2.3.0 and earlier is supported. The gRPC client created in default mode does not support the connection to the gRPC solidity node.

Download the certificate on the node details page and place the ca.crt certificate in the package under the project directory.



69e85e7f-8a71-11ee-b24b-0255ac100057

A full node must be used with its API key. FAQs

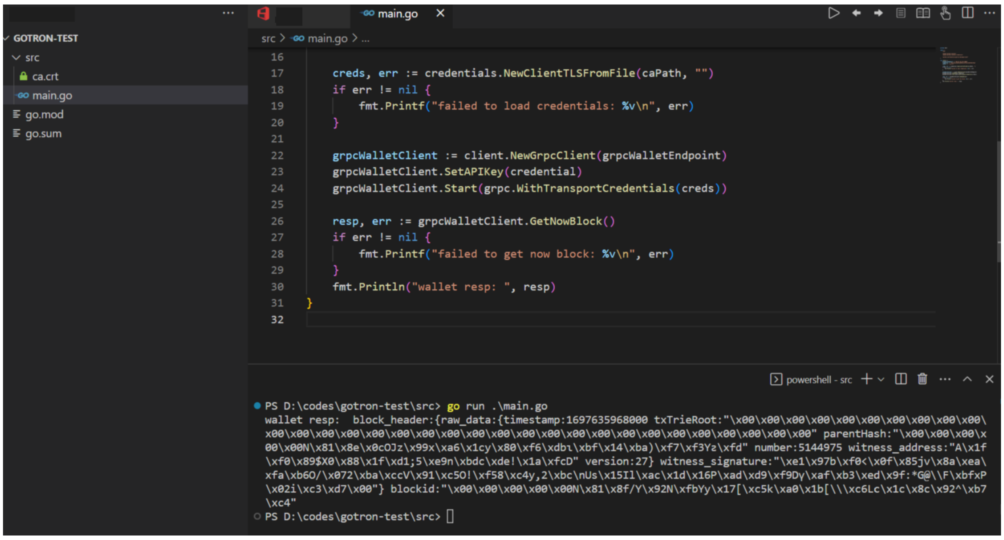
Basic Settings

Node ID	69e85e7f-8a71-11ee-b24b-0255ac100057	Public Blockchain	Tron
Status	Available	Mainnet & Testnet	Mainnet
Node Type	Full node	HTTP Endpoint	[redacted]
gRPC Endpoint	[redacted]	Node TLS Certificate	Download
Instance Flavor	Full node(Tron)16U32G	Created	Nov 24, 2023 10:30:20 GMT+08:00
AZ	AZ3	Node Client	java-tron
Node Client Version	GreatVoyage-v4.7.2		

Configure the gRPC endpoint in the SDK. The sample code is as follows:

```
import (
    "fmt"
    "google.golang.org/gRPC"
    "google.golang.org/gRPC/credentials"
    "github.com/fbsobreira/gotron-sdk/pkg/client"
)
func main() {
    gRPCWalletEndpoint := "your-gRPC-endpoint"
    credential := "your-credential"
    caPath := "your-ca.crt-file-path"
    creds, err := credentials.NewClientTLSFromFile(caPath, "")
    if err != nil {
        fmt.Printf("failed to load credentials: %v\n", err)
    }
    gRPCWalletClient := client.NewgRPCClient(gRPCWalletEndpoint)
    gRPCWalletClient.SetAPIKey(credential)
    gRPCWalletClient.Start(gRPC.WithTransportCredentials(creds))
    resp, err := gRPCWalletClient.GetNowBlock()
    if err != nil {
        fmt.Printf("failed to get now block: %v\n", err)
    }
    fmt.Println("wallet resp: ", resp)
}
```

Response example



```

    creds, err := credentials.NewClientTLSFromFile(caPath, "")
    if err != nil {
        fmt.Printf("failed to load credentials: %v\n", err)
    }

    grpcWalletClient := client.NewGrpcClient(grpcWalletEndpoint)
    grpcWalletClient.SetAPIKey(credential)
    grpcWalletClient.Start(grpc.WithTransportCredentials(creds))

    resp, err := grpcWalletClient.GetNowBlock()
    if err != nil {
        fmt.Printf("failed to get now block: %v\n", err)
    }
    fmt.Println("wallet resp: ", resp)
}

```

```
PS D:\codes\gotron-test\src> go run .\main.go
wallet resp:  block_header:{raw_data:{timestamp:1697635968000 txTrieRoot:"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00" parentHash:"\x00\x00\x00\x
00\x00N\x81\x8e\x0cOJz\x99x\xa6\x1cy\x80\xf6\xdbı\xbf\x14\xba)\xf7\xf3Yz\xfd" number:5144975 witness_address:"A\x1f
\xf0\x89$X0\x88\x1f\xd1;5\xe9n\xbdc\xde!\x1a\xfcD" version:27} witness_signature:"\xe1\x97b\xf0<\x0f\x85jv\x8a\xea\
xfa\xb6O/\x072\xba\xccV\x91\xc5O!\xf58\xc4y,2\xbc\nUs\x15Il\xac\x1d\x16P\xad\xd9\xf9Dy\xaf\xb3\xed\x9f:*G@\\F\xbfxP
\x02i\xc3\xd7\x00"} blockid:"\x00\x00\x00\x00\x00N\x81\x8f/Y\x92N\xfbYy\x17[\xc5k\xa0\x1b[\\\xc6Lc\x1c\x8c\x92^\xb7
\xc4"
PS D:\codes\gotron-test\src> 
```

2.5 Application Development

2.5.1 Using TronWeb to Send HTTP Requests

Request example

```
const TronWeb = require('tronweb');
const tronWeb = new TronWeb({
fullHost: ' your-http-endpoint / your-credential / ', // TRON HTTP endpoint
  });

let account = " accountAddress "; // The account to be queried

const main = async () => {
    let accountBalance = await tronWeb.trx.getBalance(account);
    console.log("accountBalance:\n", accountBalance);
}

main();
```

Response example

```
accountBalance:
 1998899400
```

2.5.2 Using TridentSDK to Send gRPC Requests

For details, see **Using TridentSDK to Send gRPC Requests**.

2.5.3 Using gotron-sdk to Send gRPC Requests

For details, see **Using gotron-sdk to Send gRPC Requests**.

2.6 Supported TRON APIs

Table 2-1 JSON-RPC APIs

API Method	Type	Description	Throughput (Time/s)
eth_accounts	POST	Return an array of addresses owned by the client. An empty list will be returned for tron.	1000
eth_blocknumber	POST	Return the latest block number.	1000
eth_call	POST	Execute a message call immediately without creating a transaction on the blockchain (triggerConstantContract).	1000
eth_chainId	POST	Return the chainId of the TRON network which is the last four bytes of the genesis block hash.	1000
eth_coinbase	POST	Return the witness address of the current node.	1000
eth_estimateGas	POST	Return the required energy using triggerConstantContract.	1000
eth_gasPrice	POST	Return the current energy price in sun.	1000
eth_getBalance	POST	Return the balance of the given account address.	1000
eth_getBlockByHash	POST	Return block information for the given block hash.	50
eth_getBlockByNumber	POST	Return block information for the given block number.	50

API Method	Type	Description	Throughput (Time/s)
eth_getBlockTransactionCountByHash	POST	Return the number of transactions in a block by the given block hash.	1000
eth_getBlockTransactionCountByNumber	POST	Return the number of transactions in a block by the given block number.	1000
eth_getCode	POST	Return the runtime code of a given smart contract address.	400
eth_getStorageAt	POST	Return the value from a storage position at a given address. It can be used to get the value of a variable in a contract.	1000
eth_getTransactionByBlockHashAndIndex	POST	Return information about a transaction by block hash and transaction index position.	1000
eth_getTransactionByBlockNumberAndIndex	POST	Return information about a transaction by block number and transaction index position.	1000
eth_getTransactionByHash	POST	Return the information about a transaction by transaction hash.	1000
eth_getTransactionReceipt	POST	Return transaction information, including the transaction fee, block height, and VM logs.	1000
eth_getWork	POST	Return the hash of the current block.	1000
eth_protocolVersion	POST	Return the java-tron block version.	1000
eth_syncing	POST	Return the sync status of a node.	1000
eth_newFilter	POST	Create an event filter object to listen to events.	1000
eth_newBlockFilter	POST	Create a filter to notify when a new block arrives.	1000
eth_getFilterChanges	POST	Return an array of events that have occurred since the last poll.	1000
eth_getFilterLogs	POST	Return all logs matching a given filter object.	10

API Method	Type	Description	Throughput (Time/s)
eth_uninstallFilter	POST	Uninstall a filter if monitoring is no longer required.	1000
eth_getLogs	POST	Return all logs matching a given filter object.	10
net_listening	POST	Return if the client is listening for network connections.	1000
net_peerCount	POST	Return the number of peers currently connected to the node.	1000
net_version	POST	Return the hash of the genesis block.	1000
web3_clientVersion	POST	Return the current version of the node.	1000
web3_sha3	POST	Return Keccak-256 (not the standardized SHA3-256) of the given data.	1000
buildTransaction	POST	Create a transaction. Different transaction types have different parameters.	1000

Table 2-2 JSON-RPC APIs

API Method	Description	Throughput (Time/s)
BroadcastTransaction	Broadcast the signed transaction.	1000
CreatetTansaction	Create a transaction. (Use CreateTransaction2.)	1000
CreateTransaction2	Create a transaction.	1000
CreateAccount	Activate an account. (Use CreateAccount2.)	1000

API Method	Description	Throughput (Time/s)
CreateAccount2	Activate an account.	1000
GetAccount	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	600
UpdateAccount	Modify the account name. (Use UpdateAccount2.)	1000
UpdateAccount2	Modify the account name.	1000
VoteWitnessAccount	Vote for super representatives (SRs or witnesses). Return the transaction. The transaction needs to be broadcasted after signed. (Use VoteWitnessAccount2.)	1000
VoteWitnessAccount2	Vote for witnesses. Return the transaction. The transaction needs to be broadcasted after signed.	1000
UpdateSetting	Update the consume_user_resource_percent parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
UpdatEenergyLimit	Update the origin_energy_limit parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
CreateAssetIssue	Issue a TRC-10 token. (Use CreateAssetIssue2.)	1000
CreateAssetIssue2	Issue a TRC-10 token.	1000
UpdateWitness	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed. (Use UpdateWitness2.)	1000
UpdateWitness2	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed.	1000

API Method	Description	Throughput (Time/s)
CreateWitness	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed. (Use CreateWitness2.)	1000
CreateWitness2	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed.	1000
TransferAsset	Transfer TRC-10 tokens. (Use TransferAsset2.)	1000
TransferAsset2	Transfer TRC-10 tokens.	1000
ParticipateAssetIssue	Participate in issuing a TRC-10 token. (Use ParticipateAssetIssue2.)	1000
ParticipateAssetIssue2	Participate in issuing a TRC-10 token.	1000
FreezeBalance2	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount. (Discarded.)	1000
FreezeBalanceV2	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount.	1000
UnfreezeBalance	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes. (Use UnfreezeBalance2.)	1000
UnfreezeBalance2	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes.	1000
UnfreezeBalanceV2	Unstake some TRX staked in Stake 2.0, release the corresponding amount of bandwidth or energy, and voting rights (TRON Power, TP).	1000
UnfreezeAsset	Unstake a TRC-10 token that has passed the minimum freeze duration. (Use UnfreezeAsset2.)	1000
UnfreezeAsset2	Unstake a TRC-10 token that has passed the minimum freeze duration.	1000

API Method	Description	Throughput (Time/s)
WithdrawBalance	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts. (Use WithdrawBalance2.)	1000
WithdrawBalance2	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts.	1000
WithdrawExpireUnfreeze	Withdraw the funds after a waiting period.	1000
DelegateResource	Delegate bandwidth or energy resources to other accounts in Stake 2.0.	1000
CancelAllUnfreezeV2	Cancel unstakings. Unstaked funds still in the waiting period will be re-staked. Unstaked funds that exceeded the waiting period will be automatically withdrawn to the owner's account.	1000
UpdateAsset	Update the basic TRC-10 token information. (Use UpdateAsset2.)	1000
UpdateAsset2	Update the basic TRC-10 token information.	1000
ProposalCreate	Create a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ProposalApprove	Approve a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ProposalDelete	Delete a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000

API Method	Description	Throughput (Time/s)
ExchangeCreate	Create a transaction pair. The transaction needs to be broadcasted after signed. Note that successful execution, signing, and broadcast of this API call will deduct 1024 TRX from the user's account.	1000
ExchangeInject	Inject capital into the transaction pair to prevent price fluctuation. The transaction needs to be broadcasted after signed.	1000
ExchangeWithdraw	Withdraw a transaction pair. The transaction needs to be broadcasted after signed.	1000
ExchangeTransaction	Participate in a transaction pair. The transaction needs to be broadcasted after signed.	1000
GetAssetIssueByAccount	Query the TRC-10 token information issued by an account.	1000
GetAccountNet	Query the bandwidth information of an account.	1000
GetAccountResource	Query the resource information of an account (bandwidth, energy, etc).	1000
GetAssetIssueByName	Query the TRC-10 token information by token name.	200
GetAssetIssueListByName	Query the list of all the TRC-10 tokens with a same name.	200
GetAssetIssueById	Query the TRC-10 token information by ID.	1000
GetNowBlock	Query the latest block information. (Use GetNowBlock2.)	1000
GetNowBlock2	Query the latest block information.	1000
GetBlockByNum	Query the block object corresponding to the block height. (Use GetBlockByNum2.)	15
GetBlockByNum2	Query the block object corresponding to the block height.	15
GetTransactionCountByBlockNum	Return the number of transactions in a specified block.	1000

API Method	Description	Throughput (Time/s)
GetBlockById	Query the block by ID (block hash).	15
GetBlockByLimitNext	Query the block objects in the range specified. (Use GetBlockByLimitNext2.)	10
GetBlockByLimitNext2	Query the block objects in the range specified.	10
GetBlockByLatestNum	Query the latest block objects. (Use GetBlockByLatestNum2.)	10
GetBlockByLatestNum2	Query the latest block objects.	10
GetTransactionById	Query transaction information by transaction hash.	1000
DeployContract	Return TransactionExtention, which contains an unsigned transaction.	1000
GetContract	Query the contract information from the blockchain, including the bytecode of the contract, ABI, configuration parameters, etc.	300
GetContractInfo	Query the contract information from the blockchain. It is different from the wallet/ getcontract API. This API returns not only the bytecode but also the runtime bytecode of the contract. Compared with bytecode, runtime bytecode does not contain constructor and constructor parameter information.	200
TriggerContract	Return TransactionExtention. The transaction needs to be broadcasted after signed.	1000
TriggerConstantContract	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	1000
EstimateEnergy	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	1000
ClearContractAbi	Clear the ABI info of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000

API Method	Description	Throughput (Time/s)
ListWitnesses	List all witnesses.	250
GetDelegatedResource	Query all resources delegations during Stake 1.0 phase from an account to another account. (Use GetDelegatedResourceV2.)	1000
GetDelegatedResourceV2	Query all resources delegations during Stake 1.0 phase from an account to another account.	1000
GetDelegatedResourceAccountIndex	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	1000
GetDelegatedResourceAccountIndexV2	Query the resource delegation index in Stake 2.0.	1000
GetCanDelegatedMaxSize	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	1000
GetAvailableUnfreezeCount	Query the remaining times of executing unstake operations in Stake 2.0.	1000
GetCanWithdrawUnfreezeAmount	Query the withdrawable balance at the specified timestamp.	1000
ListProposals	List all proposals.	300
GetProposalById	Query a proposal based on the ID and return proposal details.	1000
ListExchanges	List all transaction pairs.	400
GetExchangeById	Query a transaction pair based on ID.	1000
GetChainParameters	Query all proposal parameters that the witnesses can set.	1000
GetAssetIssueList	Query the list of all the TRC-10 tokens.	5
GetPaginatedAssetIssueList	Query the list of all the TRC-10 tokens by page.	20
GetNextMaintenanceTime	Return the timestamp of the next voting time in milliseconds.	1000
GetTransactionInfoById	Return transaction information, including the transaction fee, block height, and VM logs.	1000

API Method	Description	Throughput (Time/s)
AccountPermissionUpdate	Update the account's permission.	1000
GetTransactionSignWeight	Query the total weight of a signed transaction.	1000
GetTransactionApprovedList	Query the account address list which signed the transaction, by the transaction content and signature information.	1000
GetNodeInfo	Query node information.	700
GetRewardInfo	Get the rewards that a witness or a user has not yet withdrawn.	1000
GetBrokerageInfo	Query the witness's brokerage ratio.	1000
UpdateBrokerage	Update the witness's brokerage setting. The transaction needs to be broadcasted after signed.	1000
GetTransactionInfoByBlockNum	Query transaction information in a block specified.	150
GetBurnTrx	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	1000
GetTransactionFromPending	Get transaction details from the pending pool.	1000
GetTransactionListFromPending	Get transaction list information from the pending pool.	1000
GetPendingSize	Get the size of the pending pool queue.	1000
GetBlock	Query block header information or entire block information according to block height or block hash.	50
UnDelegateResource	Cancel the delegation of bandwidth or energy resources to other accounts in Stake 2.0.	1000

Table 2-3 gRPC solidity APIs

API Method	Description	Throughput (Time/s)
GetAccount	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	600
ListWitnesses	List all witnesses.	250
GetAssetIssueList	Query the list of all the TRC-10 tokens.	5
GetPaginatedAssetIssueList	Query the list of all the TRC-10 tokens by page.	20
GetAssetIssueByName	Query the TRC-10 token information by token name.	200
GetAssetIssueListByName	Query the list of all the TRC-10 tokens with a same name.	200
GetAssetIssueById	Query the TRC-10 token information by ID.	1000
GetNowBlock	Query the latest block information. (Use GetNowBlock2.)	1000
GetNowBlock2	Query the latest block information.	1000
GetBlockByNum	Query the block object corresponding to the block height. (Use GetBlockByNum2.)	15
GetBlockByNum2	Query the block object corresponding to the block height.	15
GetTransactionCountByBlockNum	Return the number of transactions in a specified block.	1000
GetDelegatedResource	Query all resources delegations during Stake 1.0 phase from an account to another account. (Use GetDelegatedResourceV2.)	1000
GetDelegatedResourceV2	Query all resources delegations during Stake 1.0 phase from an account to another account.	1000
GetDelegatedResourceAccountIndex	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	1000
GetDelegatedResourceAccountIndexV2	Query the resource delegation index in Stake 2.0.	1000

API Method	Description	Throughput (Time/s)
GetCanDelegatedMaxSize	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	1000
GetAvailableUnfreezeCount	Query the remaining times of executing unstake operations in Stake 2.0.	1000
GetCanWithdrawUnfreezeAmount	Query the withdrawable balance at the specified timestamp.	1000
GetExchangeById	Query a transaction pair based on ID.	1000
ListExchanges	List all transaction pairs.	400
GetTransactionById	Query transaction information by transaction hash.	1000
GetTransactionInfoById	Return transaction information, including the transaction fee, block height, and VM logs.	1000
GetRewardInfo	Get the rewards that a witness or a user has not yet withdrawn.	1000
GetBrokerageInfo	Query the witness's brokerage ratio.	1000
TriggerConstantContract	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	1000
EstimateEnergy	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	1000
GetTransactionInfoByBlockNum	Query transaction information in a block specified.	150
GetBurnTrx	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	1000
GetBlock	Query block header information or entire block information according to block height or block hash.	50

2.6.1 Dedicated Edition

Table 2-4 Wallet APIs

API Method	Type	Description	Throughput (Time/s)
/wallet/validateaddress	POST	Validate address, return either true or false.	1000
/wallet/broadcasttransaction	POST	Broadcast the signed transaction.	1000
/wallet/broadcasthex	POST	Broadcast the protobuf encoded transaction hexadecimal string after being signed.	1000
/wallet/createtransaction	POST	Create a transaction.	1000
/wallet/createaccount	POST	Activate an account.	1000
/wallet/getaccount	POST	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	600
/wallet/updateaccount	POST	Modify the account name.	1000
/wallet/accountpermissionupdate	POST	Update the account's permission.	1000
/wallet/getaccountresource	POST	Query the resource information of an account (bandwidth, energy, etc).	1000
/wallet/getaccountnet	POST	Query the bandwidth information of an account.	1000
/wallet/unfreezebalance	POST	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes.	1000

API Method	Type	Description	Throughput (Time/s)
/wallet/getdelegatedresource	POST	Query all resources delegations during Stake 1.0 phase from an account to another account.	1000
/wallet/getdelegatedresourceaccountindex	POST	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	1000
/wallet/freezebalancev2	POST	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount.	1000
/wallet/unfreezebalancev2	POST	Unstake some TRX staked in Stake 2.0, release the corresponding amount of bandwidth or energy, and voting rights (TP).	1000
/wallet/cancelallunfreezev2	POST	Cancel unstakings. Unstaked funds still in the waiting period will be re-staked. Unstaked funds that exceeded the waiting period will be automatically withdrawn to the owner's account.	1000
/wallet/delegateresource	POST	Delegate bandwidth or energy resources to other accounts in Stake 2.0.	1000
/wallet/undelegateresource	POST	Cancel the delegation of bandwidth or energy resources to other accounts in Stake 2.0.	1000
/wallet/withdrawexpireunfreeze	POST	Withdraw the funds after a waiting period.	1000
/wallet/getavailableunfreezecount	POST	Query the remaining times of executing unstake operations in Stake 2.0.	1000
/wallet/getcanwithdrawunfreezeamount	POST	Query the withdrawable balance at the specified timestamp.	1000
/wallet/getcandelegatedmaxsize	POST	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	1000

API Method	Type	Description	Throughput (Time/s)
/wallet/getdelegatedresourcev2	POST	In Stake 2.0, query the detail of resource share delegated from the source to the target address.	1000
/wallet/getdelegatedresourceaccountindexv2	POST	Query the resource delegation index in Stake 2.0.	1000
/wallet/getblock	POST	Query block header information or entire block information according to block height or block hash.	50
/wallet/getblockbynum	POST	Query the block object corresponding to the block height.	15
/wallet/getblockbyid	POST	Query the block by ID (block hash).	15
/wallet/getblockbylatestnum	POST	Query the latest block objects.	10
/wallet/getblockbylimitnext	POST	Query the block objects in the range specified.	10
/wallet/getnowblock	GET	Query the latest block information.	1000
/wallet/gettransactionbyid	POST	Query transaction information by transaction hash.	1000
/wallet/gettransactioninfobyid	POST	Return transaction information, including the transaction fee, block height, and VM logs.	1000
/wallet/gettransactioninfobyblocknum	POST	Query transaction information in a block specified.	150
/wallet/getnodeinfo	GET	Query node information.	700
/wallet/getchainparameters	GET	Query all proposal parameters that the witnesses can set.	1000
/wallet/getenergyprices	GET	Query historical energy unit price.	1000
/wallet/getbandwidthprices	GET	Query historical bandwidth unit price.	1000

API Method	Type	Description	Throughput (Time/s)
/wallet/getburntrx	GET	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	1000
/wallet/getapprovedlist	POST	Query the account address list which signed the transaction, by the transaction content and signature information.	1000
/wallet/getassetissuebyaccount	POST	Query the TRC-10 token information issued by an account.	1000
/wallet/getassetissuebyid	POST	Query the TRC-10 token information by ID.	1000
/wallet/getassetissuebyname	POST	Query the TRC-10 token information by token name.	150
/wallet/getassetissuelist	GET	Query the list of all the TRC-10 tokens.	5
/wallet/getassetissuelistbyname	POST	Query the list of all the TRC-10 tokens with a same name.	150
/wallet/getpaginatedassetissuelist	POST	Query the list of all the TRC-10 tokens by page.	20
/wallet/transferasset	POST	Transfer TRC-10 tokens.	1000
/wallet/createassetissue	POST	Issue a TRC-10 token.	1000
/wallet/participateassetissue	POST	Participate in issuing a TRC-10 token.	1000
/wallet/unfreezeasset	POST	Unstake a TRC-10 token that has passed the minimum freeze duration.	1000
/wallet/updateasset	POST	Update the basic TRC-10 token information.	1000
/wallet/getcontract	POST	Query the contract information from the blockchain, including the bytecode of the contract, ABI, configuration parameters, etc.	300

API Method	Type	Description	Throughput (Time/s)
/wallet/getcontractinfo	POST	Query the contract information from the blockchain. It is different from the wallet/getcontract API. This API returns not only the bytecode but also the runtime bytecode of the contract. Compared with bytecode, runtime bytecode does not contain constructor and constructor parameter information.	200
/wallet/triggersmartcontract	POST	Return TransactionExtention. The transaction needs to be broadcasted after signed.	1000
/wallet/triggerconstantcontract	POST	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	1000
/wallet/deploycontract	POST	Return TransactionExtention, which contains an unsigned transaction.	1000
/wallet/updatesetting	POST	Update the consume_user_resource_percent parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/updateenergylimit	POST	Update the origin_energy_limit parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/clearabi	POST	Clear the ABI info of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/estimateenergy	POST	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	1000
/wallet/createwitness	POST	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed.	1000

API Method	Type	Description	Throughput (Time/s)
/wallet/updatewitness	POST	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed.	1000
/wallet/listwitnesses	GET	List all witnesses.	200
/wallet/votewitnessaccount	POST	Vote for witnesses. Return the transaction. The transaction needs to be broadcasted after signed.	1000
/wallet/updateBrokerage	POST	Update the witness's brokerage setting. The transaction needs to be broadcasted after signed.	1000
/wallet/getBrokerage	POST	Query the witness's brokerage ratio.	1000
/wallet/getReward	POST	Get the rewards that a witness or a user has not yet withdrawn.	1000
/wallet/withdrawbalance	POST	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts.	1000
/wallet/getnextmaintenance-time	GET	Return the timestamp of the next voting time in milliseconds.	1000
/wallet/proposalcreate	POST	Create a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/proposalapprove	POST	Approve a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/proposaldelete	POST	Delete a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
/wallet/listproposals	GET	List all proposals.	300
/wallet/getproposalbyid	POST	Query a proposal based on the ID and return proposal details.	1000

API Method	Type	Description	Throughput (Time/s)
/wallet/exchangecreate	POST	Create a transaction pair. The transaction needs to be broadcasted after signed. Note that successful execution, signing, and broadcast of this API call will deduct 1024 TRX from the user's account.	1000
/wallet/exchangeinject	POST	Inject capital into the transaction pair to prevent price fluctuation. The transaction needs to be broadcasted after signed.	1000
/wallet/exchangewithdraw	POST	Withdraw a transaction pair. The transaction needs to be broadcasted after signed.	1000
/wallet/exchangetransaction	POST	Participate in a transaction pair. The transaction needs to be broadcasted after signed.	1000
/wallet/getexchangebyid	POST	Query a transaction pair based on ID.	1000
/wallet/listexchanges	GET	List all transaction pairs.	400
/wallet/gettransactionlistfrom-pending	GET	Get transaction list information from the pending pool.	1000
/wallet/gettransactionfrom-pending	POST	Get transaction details from the pending pool.	1000
/wallet/getpendingsize	GET	Get the size of the pending pool queue.	1000
/wallet/getsignweight	POST	Query the total weight of a signed transaction.	1000

Table 2-5 Wallet solidity APIs

API Method	Type	Description	Throughput (Time/s)
/walletsolidity/gettransactionbyid	POST	Query transaction information by transaction hash (confirmed state).	1000
/walletsolidity/gettransactioninfobyid	POST	Return transaction information, including the transaction fee, block height, and VM logs (confirmed state).	1000
/walletsolidity/gettransactioninfobyblocknum	POST	Query transaction information in a block specified (confirmed state).	1000
/walletsolidity/gettransactioncountbyblocknum	POST	Query the number of transactions in a block by block number (confirmed state).	1000
/walletsolidity/getblock	POST	Query block header information or entire block information according to block height or block hash (confirmed state).	1000
/walletsolidity/getblockbyid	POST	Query the block by ID, that is, block hash (confirmed state).	1000
/walletsolidity/getblockbylatestnum	POST	Query the latest block objects (confirmed state).	1000
/walletsolidity/getblockbylimitnext	POST	Query the block objects in the range specified (confirmed state).	1000
/walletsolidity/getblockbynum	POST	Query whether a specified block is confirmed.	1000
/walletsolidity/getnowblock	GET	Query the latest block information (confirmed state).	1000
/walletsolidity/getaccount	POST	Query the information about an account (confirmed state).	1000
/walletsolidity/getdelegatedresource	POST	Query all resources delegations during Stake 1.0 phase from an account to another account (confirmed state).	1000
/walletsolidity/getdelegatedresourcev2	POST	Query the resource delegation index in Stake 2.0 (confirmed state).	1000

API Method	Type	Description	Throughput (Time/s)
/walletsolidity/getcandelegatedmaxsize	POST	Query the amount of delegatable resources share of the specified resource type for an address in Stake 2.0 (confirmed state). The unit is sun.	1000
/walletsolidity/getavailableunfreezecount	POST	Query the remaining times of executing unstake operations in Stake 2.0 (confirmed state).	1000
/walletsolidity/getcanwithdrawunfreezeamount	POST	Query the withdrawable balance at the specified timestamp in Stake 2.0 (confirmed state).	1000
/walletsolidity/getdelegatedresourceaccountindexv2	POST	Query the resource delegation index in Stake 2.0 (confirmed state). Two lists will be returned. One lists the target addresses that have received the delegation (toAddress). The other lists the addresses that have delegated resources (fromAddress).	1000
/walletsolidity/getnodeinfo	GET	Query node information (confirmed state).	1000
/walletsolidity/getburntrx	GET	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect (confirmed state).	1000
/walletsolidity/triggerconstantcontract	POST	Invoke the read-only function of a contract (view or pure, for querying confirmed state data), or invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption in confirmed state).	1000
/walletsolidity/estimateenergy	POST	Estimate the energy required for the successful execution of smart contract transactions (confirmed state).	1000
/walletsolidity/getassetissuebyid	POST	Query the TRC-10 token information by ID (confirmed state).	1000
/walletsolidity/getassetissuebyname	POST	Query the TRC-10 token information by token name (confirmed state).	150

API Method	Type	Description	Throughput (Time/s)
/walletsolidity/getassetissuelist	GET	Query the list of all the TRC-10 tokens (confirmed state).	5
/walletsolidity/getassetissuelistbyname	POST	Query the list of all the TRC-10 tokens with a same name (confirmed state).	150
/walletsolidity/getpaginatedassetissuelist	POST	Query the list of all the TRC-10 tokens by page (confirmed state).	20
/walletsolidity/listwitnesses	GET	List all witnesses (confirmed state).	20
/walletsolidity/getBrokerage	POST	Query the witness's brokerage ratio (confirmed state).	1000
/walletsolidity/getReward	POST	Get the rewards that a witness or a user has not yet withdrawn (confirmed state).	1000
/walletsolidity/getexchangebyid	POST	Query a transaction pair by ID (confirmed state).	1000
/walletsolidity/listexchanges	GET	Query transaction pairs (confirmed state).	1000
/walletsolidity/getenergyprices	GET	Query historical energy unit price.	1000
/walletsolidity/getbandwidthprices	GET	Query historical bandwidth unit price.	1000

Table 2-6 JSON-RPC APIs

API Method	Type	Description	Throughput (Time/s)
eth_accounts	POST	Return an array of addresses owned by the client. An empty list will be returned for tron.	1000

API Method	Type	Description	Throughput (Time/s)
eth_blocknumber	POST	Return the latest block number.	1000
eth_call	POST	Execute a message call immediately without creating a transaction on the blockchain (triggerConstantContract).	1000
eth_chainId	POST	Return the chainId of the TRON network which is the last four bytes of the genesis block hash.	1000
eth_coinbase	POST	Return the witness address of the current node.	1000
eth_estimateGas	POST	Return the required energy using triggerConstantContract.	1000
eth_gasPrice	POST	Return the current energy price in sun.	1000
eth_getBalance	POST	Return the balance of the given account address.	1000
eth_getBlockByHash	POST	Return block information for the given block hash.	50
eth_getBlockByNumber	POST	Return block information for the given block number.	50
eth_getBlockTransactionCountByHash	POST	Return the number of transactions in a block by the given block hash.	1000
eth_getBlockTransactionCountByNumber	POST	Return the number of transactions in a block by the given block number.	1000
eth_getCode	POST	Return the runtime code of a given smart contract address.	400
eth_getStorageAt	POST	Return the value from a storage position at a given address. It can be used to get the value of a variable in a contract.	1000
eth_getTransactionByBlockHashAndIndex	POST	Return information about a transaction by block hash and transaction index position.	1000
eth_getTransactionByBlockNumberAndIndex	POST	Return information about a transaction by block number and transaction index position.	1000

API Method	Type	Description	Throughput (Time/s)
eth_getTransactionByHash	POST	Return the information about a transaction by transaction hash.	1000
eth_getTransactionReceipt	POST	Return transaction information, including the transaction fee, block height, and VM logs.	1000
eth_getWork	POST	Return the hash of the current block.	1000
eth_protocolVersion	POST	Return the java-tron block version.	1000
eth_syncing	POST	Return the sync status of a node.	1000
eth_newFilter	POST	Create an event filter object to listen to events.	1000
eth_newBlockFilter	POST	Create a filter to notify when a new block arrives.	1000
eth_getFilterChanges	POST	Return an array of events that have occurred since the last poll.	1000
eth_getFilterLogs	POST	Return all logs matching a given filter object.	10
eth_uninstallFilter	POST	Uninstall a filter if monitoring is no longer required.	1000
eth_getLogs	POST	Return all logs matching a given filter object.	10
net_listening	POST	Return if the client is listening for network connections.	1000
net_version	POST	Return the hash of the genesis block.	1000
web3_clientVersion	POST	Return the current version of the node.	1000
web3_sha3	POST	Return Keccak-256 (not the standardized SHA3-256) of the given data.	1000
buildTransaction	POST	Create a transaction. Different transaction types have different parameters.	1000

Table 2-7 JSON-RPC APIs

API Method	Description	Throughput (Time/s)
BroadcastTransaction	Broadcast the signed transaction.	1000
CreatetTansaction	Create a transaction. (Use CreateTransaction2.)	1000
CreateTransaction2	Create a transaction.	1000
CreateAccount	Activate an account. (Use CreateAccount2.)	1000
CreateAccount2	Activate an account.	1000
GetAccount	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	600
UpdateAccount	Modify the account name. (Use UpdateAccount2.)	1000
UpdateAccount2	Modify the account name.	1000
VoteWitnessAccount	Vote for witnesses. Return the transaction. The transaction needs to be broadcasted after signed. (Use VoteWitnessAccount2.)	1000
VoteWitnessAccount2	Vote for witnesses. Return the transaction. The transaction needs to be broadcasted after signed.	1000
UpdateSetting	Update the consume_user_resource_percent parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
UpdatEenergyLimit	Update the origin_energy_limit parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000

API Method	Description	Throughput (Time/s)
CreateAssetIssue	Issue a TRC-10 token. (Use CreateAssetIssue2.)	1000
CreateAssetIssue2	Issue a TRC-10 token.	1000
UpdateWitness	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed. (Use UpdateWitness2.)	1000
UpdateWitness2	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed.	1000
CreateWitness	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed. (Use CreateWitness2.)	1000
CreateWitness2	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed.	1000
TransferAsset	Transfer TRC-10 tokens. (Use TransferAsset2.)	1000
TransferAsset2	Transfer TRC-10 tokens.	1000
ParticipateAssetIssue	Participate in issuing a TRC-10 token. (Use ParticipateAssetIssue2.)	1000
ParticipateAssetIssue2	Participate in issuing a TRC-10 token.	1000
FreezeBalance2	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount. (Discarded.)	1000
FreezeBalanceV2	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount.	1000
UnfreezeBalance	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes. (Use UnfreezeBalance2.)	1000

API Method	Description	Throughput (Time/s)
UnfreezeBalance2	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes.	1000
UnfreezeBalanceV2	Unstake some TRX staked in Stake 2.0, release the corresponding amount of bandwidth or energy, and voting rights (TP).	1000
UnfreezeAsset	Unstake a TRC-10 token that has passed the minimum freeze duration. (Use UnfreezeAsset2.)	1000
UnfreezeAsset2	Unstake a TRC-10 token that has passed the minimum freeze duration.	1000
WithdrawBalance	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts. (Use WithdrawBalance2.)	1000
WithdrawBalance2	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts.	1000
WithdrawExpireUnfreeze	Withdraw the funds after a waiting period.	1000
DelegateResource	Delegate bandwidth or energy resources to other accounts in Stake 2.0.	1000
CancelAllUnfreezeV2	Cancel unstakings. Unstaked funds still in the waiting period will be re-staked. Unstaked funds that exceeded the waiting period will be automatically withdrawn to the owner's account.	1000
UpdateAsset	Update the basic TRC-10 token information. (Use UpdateAsset2.)	1000
UpdateAsset2	Update the basic TRC-10 token information.	1000

API Method	Description	Throughput (Time/s)
ProposalCreate	Create a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ProposalApprove	Approve a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ProposalDelete	Delete a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ExchangeCreate	Create a transaction pair. The transaction needs to be broadcasted after signed. Note that successful execution, signing, and broadcast of this API call will deduct 1024 TRX from the user's account.	1000
ExchangeInject	Inject capital into the transaction pair to prevent price fluctuation. The transaction needs to be broadcasted after signed.	1000
ExchangeWithdraw	Withdraw a transaction pair. The transaction needs to be broadcasted after signed.	1000
ExchangeTransaction	Participate in a transaction pair. The transaction needs to be broadcasted after signed.	1000
GetAssetIssueByAccount	Query the TRC-10 token information issued by an account.	1000
GetAccountNet	Query the bandwidth information of an account.	1000
GetAccountResource	Query the resource information of an account (bandwidth, energy, etc).	1000
GetAssetIssueByName	Query the TRC-10 token information by token name.	200
GetAssetIssueListByName	Query the list of all the TRC-10 tokens with a same name.	200
GetAssetIssueById	Query the TRC-10 token information by ID.	1000
GetNowBlock	Query the latest block information. (Use GetNowBlock2.)	1000

API Method	Description	Throughput (Time/s)
GetNowBlock2	Query the latest block information.	1000
GetBlockByNum	Query the block object corresponding to the block height. (Use GetBlockByNum2.)	15
GetBlockByNum2	Query the block object corresponding to the block height.	15
GetTransactionCount-ByBlockNum	Return the number of transactions in a specified block.	1000
GetBlockById	Query the block by ID (block hash).	15
GetBlockByLimitNext	Query the block objects in the range specified. (Use GetBlockByLimitNext2.)	10
GetBlockByLimitNext2	Query the block objects in the range specified.	10
GetBlockByLatestNum	Query the latest block objects. (Use GetBlockByLatestNum2.)	10
GetBlockByLatest-Num2	Query the latest block objects.	10
GetTransactionById	Query transaction information by transaction hash.	1000
DeployContract	Return TransactionExtention, which contains an unsigned transaction.	1000
GetContract	Query the contract information from the blockchain, including the bytecode of the contract, ABI, configuration parameters, etc.	300
GetContractInfo	Query the contract information from the blockchain. It is different from the wallet/getcontract API. This API returns not only the bytecode but also the runtime bytecode of the contract. Compared with bytecode, runtime bytecode does not contain constructor and constructor parameter information.	200
TriggerContract	Return TransactionExtention. The transaction needs to be broadcasted after signed.	1000

API Method	Description	Throughput (Time/s)
TriggerConstantContract	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	1000
EstimateEnergy	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	1000
ClearContractAbi	Clear the ABI info of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	1000
ListWitnesses	List all witnesses.	250
GetDelegatedResource	Query all resources delegations during Stake 1.0 phase from an account to another account. (Use GetDelegatedResourceV2.)	1000
GetDelegatedResourceV2	Query all resources delegations during Stake 1.0 phase from an account to another account.	1000
GetDelegatedResourceAccountIndex	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	1000
GetDelegatedResourceAccountIndexV2	Query the resource delegation index in Stake 2.0.	1000
GetCanDelegatedMaxSize	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	1000
GetAvailableUnfreezeCount	Query the remaining times of executing unstake operations in Stake 2.0.	1000
GetCanWithdrawUnfreezeAmount	Query the withdrawable balance at the specified timestamp.	1000
ListProposals	List all proposals.	300
GetProposalById	Query a proposal based on the ID and return proposal details.	1000
ListExchanges	List all transaction pairs.	400

API Method	Description	Throughput (Time/s)
GetExchangeById	Query a transaction pair based on ID.	1000
GetChainParameters	Query all proposal parameters that the witnesses can set.	1000
GetAssetIssueList	Query the list of all the TRC-10 tokens.	5
GetPaginatedAssetIssueList	Query the list of all the TRC-10 tokens by page.	20
GetNextMaintenanceTime	Return the timestamp of the next voting time in milliseconds.	1000
GetTransactionInfoById	Return transaction information, including the transaction fee, block height, and VM logs.	1000
AccountPermissionUpdate	Update the account's permission.	1000
GetTransactionSignWeight	Query the total weight of a signed transaction.	1000
GetTransactionApprovedList	Query the account address list which signed the transaction, by the transaction content and signature information.	1000
GetNodeInfo	Query node information.	700
GetRewardInfo	Get the rewards that a witness or a user has not yet withdrawn.	1000
GetBrokerageInfo	Query the witness's brokerage ratio.	1000
UpdateBrokerage	Update the witness's brokerage setting. The transaction needs to be broadcasted after signed.	1000
GetTransactionInfoByBlockNum	Query transaction information in a block specified.	150
GetBurnTrx	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	1000
GetTransactionFromPending	Get transaction details from the pending pool.	1000

API Method	Description	Throughput (Time/s)
GetTransactionListFromPending	Get transaction list information from the pending pool.	1000
GetPendingSize	Get the size of the pending pool queue.	1000
GetBlock	Query block header information or entire block information according to block height or block hash.	50
UnDelegateResource	Cancel the delegation of bandwidth or energy resources to other accounts in Stake 2.0.	1000
GetBandwidthPrices	Query historical bandwidth unit price.	1000
GetEnergyPrices	Query historical energy unit price.	1000
GetMemoFee	Get transaction memo fee.	1000

Table 2-8 gRPC solidity APIs

API Method	Description	Throughput (Time/s)
GetAccount	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	600
ListWitnesses	List all witnesses.	250
GetAssetIssueList	Query the list of all the TRC-10 tokens.	5
GetPaginatedAssetIssueList	Query the list of all the TRC-10 tokens by page.	20
GetAssetIssueByName	Query the TRC-10 token information by token name.	200

API Method	Description	Throughput (Time/s)
GetAssetIssueListByName	Query the list of all the TRC-10 tokens with a same name.	200
GetAssetIssueById	Query the TRC-10 token information by ID.	1000
GetNowBlock	Query the latest block information. (Use GetNowBlock2.)	1000
GetNowBlock2	Query the latest block information.	1000
GetBlockByNum	Query the block object corresponding to the block height. (Use GetBlockByNum2.)	15
GetBlockByNum2	Query the block object corresponding to the block height.	15
GetTransactionCountByBlockNum	Return the number of transactions in a specified block.	1000
GetDelegatedResource	Query all resources delegations during Stake 1.0 phase from an account to another account. (Use GetDelegatedResourceV2.)	1000
GetDelegatedResourceV2	Query all resources delegations during Stake 1.0 phase from an account to another account.	1000
GetDelegatedResourceAccountIndex	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	1000
GetDelegatedResourceAccountIndexV2	Query the resource delegation index in Stake 2.0.	1000
GetCanDelegatedMaxSize	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	1000
GetAvailableUnfreezeCount	Query the remaining times of executing unstake operations in Stake 2.0.	1000
GetCanWithdrawUnfreezeAmount	Query the withdrawable balance at the specified timestamp.	1000
GetExchangeById	Query a transaction pair based on ID.	1000
ListExchanges	List all transaction pairs.	400

API Method	Description	Throughput (Time/s)
GetTransactionById	Query transaction information by transaction hash.	1000
GetTransactionInfoById	Return transaction information, including the transaction fee, block height, and VM logs.	1000
GetRewardInfo	Get the rewards that a witness or a user has not yet withdrawn.	1000
GetBrokerageInfo	Query the witness's brokerage ratio.	1000
TriggerConstantContract	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	1000
EstimateEnergy	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	1000
GetTransactionInfoByBlockNum	Query transaction information in a block specified.	150
GetBurnTrx	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	1000
GetBlock	Query block header information or entire block information according to block height or block hash.	50
GetBandwidthPrices	Query historical bandwidth unit price.	1000
GetEnergyPrices	Query historical energy unit price.	1000

2.6.2 Shared Edition

Table 2-9 Wallet APIs

API Method	Type	Description	CU
/wallet/validateaddress	POST	Validate address, return either true or false.	5
/wallet/broadcasttransaction	POST	Broadcast the signed transaction.	5
/wallet/broadcasthex	POST	Broadcast the protobuf encoded transaction hexadecimal string after being signed.	4
/wallet/createtransaction	POST	Create a transaction.	9
/wallet/createaccount	POST	Activate an account.	10
/wallet/getaccount	POST	Query information about an account, including TRX balance, TRC-10 balances, stake information, vote information, and permissions.	9
/wallet/updateaccount	POST	Modify the account name.	9
/wallet/accountpermissionupdate	POST	Update the account's permission.	12
/wallet/getaccountresource	POST	Query the resource information of an account (bandwidth, energy, etc).	7
/wallet/getaccountnet	POST	Query the bandwidth information of an account.	7
/wallet/unfreezebalance	POST	Unstake the TRX staked during Stake 1.0, release the obtained bandwidth or energy and TP, and automatically cancel all votes.	5
/wallet/getdelegatedresource	POST	Query all resources delegations during Stake 1.0 phase from an account to another account.	5
/wallet/getdelegatedresourceaccountindex	POST	Query the resource delegation by an account during Stake 1.0 phase and list all addresses that have delegated resources to an account.	40

API Method	Type	Description	CU
/wallet/freezebalancev2	POST	In Stake 2.0, stake an amount of TRX to obtain bandwidth or energy, and obtain equivalent TP according to the staked amount.	10
/wallet/unfreezebalancev2	POST	Unstake some TRX staked in Stake 2.0, release the corresponding amount of bandwidth or energy, and voting rights (TP).	7
/wallet/delegateresource	POST	Delegate bandwidth or energy resources to other accounts in Stake 2.0.	5
/wallet/undelegateresource	POST	Cancel the delegation of bandwidth or energy resources to other accounts in Stake 2.0.	7
/wallet/withdrawexpireunfreeze	POST	Withdraw the funds after a waiting period.	7
/wallet/getavailableunfreezecount	POST	Query the remaining times of executing unstake operations in Stake 2.0.	5
/wallet/getcanwithdrawunfreezeamount	POST	Query the withdrawable balance at the specified timestamp.	6
/wallet/getcandelegatedmaxsize	POST	Query the amount of delegatable resources share of the specified resource type for an address. The unit is sun.	7
/wallet/getdelegatedresourcev2	POST	In Stake 2.0, query the detail of resource share delegated from the source to the target address.	5
/wallet/getdelegatedresourceaccountindexv2	POST	Query the resource delegation index in Stake 2.0.	40
/wallet/getblock	POST	Query block header information or entire block information according to block height or block hash.	392
/wallet/getblockbynum	POST	Query the block object corresponding to the block height.	192
/wallet/getblockbyid	POST	Query the block by ID (block hash).	392

API Method	Type	Description	CU
/wallet/getblockbylatestnum	POST	Query the latest block objects.	941
/wallet/getblockbylimitnext	POST	Query the block objects in the range specified.	392
/wallet/getnowblock	GET	Query the latest block information.	542
/wallet/gettransactionbyid	POST	Query transaction information by transaction hash.	52
/wallet/gettransactioninfobyid	POST	Return transaction information, including the transaction fee, block height, and VM logs.	19
/wallet/gettransactioninfobyblocknum	POST	Query transaction information in a block specified.	139
/wallet/getchainparameters	GET	Query all proposal parameters that the witnesses can set.	13
/wallet/getenergyprices	GET	Query historical energy unit price.	8
/wallet/getbandwidthprices	GET	Query historical bandwidth unit price.	6
/wallet/getburntrx	GET	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect.	6
/wallet/getapprovedlist	POST	Query the account address list which signed the transaction, by the transaction content and signature information.	20
/wallet/getassetissuebyaccount	POST	Query the TRC-10 token information issued by an account.	523
/wallet/getassetissuebyid	POST	Query the TRC-10 token information by ID.	9
/wallet/getassetissuebyname	POST	Query the TRC-10 token information by token name.	570
/wallet/getassetissuelist	GET	Query the list of all the TRC-10 tokens.	4706
/wallet/getassetissuelistbyname	POST	Query the list of all the TRC-10 tokens with a same name.	509

API Method	Type	Description	CU
/wallet/getpaginatedassetissuelist	POST	Query the list of all the TRC-10 tokens by page.	784
/wallet/transferasset	POST	Transfer TRC-10 tokens.	20
/wallet/createassetissue	POST	Issue a TRC-10 token.	20
/wallet/participateassetissue	POST	Participate in issuing a TRC-10 token.	20
/wallet/unfreezeasset	POST	Unstake a TRC-10 token that has passed the minimum freeze duration.	20
/wallet/updateasset	POST	Update the basic TRC-10 token information.	20
/wallet/getcontract	POST	Query the contract information from the blockchain, including the bytecode of the contract, ABI, configuration parameters, etc.	84
/wallet/getcontractinfo	POST	Query the contract information from the blockchain. It is different from the wallet/getcontract API. This API returns not only the bytecode but also the runtime bytecode of the contract. Compared with bytecode, runtime bytecode does not contain constructor and constructor parameter information.	114
/wallet/triggersmartcontract	POST	Return TransactionExtention. The transaction needs to be broadcasted after signed.	20
/wallet/triggerconstantcontract	POST	Invoke the read-only function of a contract, invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption), or estimate the energy consumption of contract deployment.	20
/wallet/deploycontract	POST	Return TransactionExtention, which contains an unsigned transaction.	14

API Method	Type	Description	CU
/wallet/updatesetting	POST	Update the consume_user_resource_percent parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	20
/wallet/updateenergylimit	POST	Update the origin_energy_limit parameter of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	20
/wallet/clearabi	POST	Clear the ABI info of a smart contract. An unsigned transaction is returned and needs to be broadcasted after signed.	20
/wallet/estimateenergy	POST	Estimate the energy required for the successful execution of smart contract transactions or deploying a contract.	13
/wallet/createwitness	POST	Apply to become a witness. Return the transaction. The transaction needs to be broadcasted after signed.	20
/wallet/updatewitness	POST	Edit the URL of the witness's official website. The transaction needs to be broadcasted after signed.	20
/wallet/listwitnesses	GET	List all witnesses.	20
/wallet/votewitnessaccount	POST	Vote for witnesses. Return the transaction. The transaction needs to be broadcasted after signed.	20
/wallet/updateBrokerage	POST	Update the witness's brokerage setting. The transaction needs to be broadcasted after signed.	7
/wallet/getBrokerage	POST	Query the witness's brokerage ratio.	5
/wallet/getReward	POST	Get the rewards that a witness or a user has not yet withdrawn.	5

API Method	Type	Description	CU
/wallet/withdrawbalance	POST	Withdraw rewards by witnesses or users, available every 24 hours. Witnesses can withdraw the balance from the account allowance into the account balance. Users can claim the voting rewards and deposit into their accounts.	20
/wallet/getnextmaintenance-time	GET	Return the timestamp of the next voting time in milliseconds.	20
/wallet/proposalcreate	POST	Create a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	5
/wallet/proposalapprove	POST	Approve a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	20
/wallet/proposaldelete	POST	Delete a proposal transaction. An unsigned transaction is returned and needs to be broadcasted after signed.	20
/wallet/listproposals	GET	List all proposals.	116
/wallet/getproposalbyid	POST	Query a proposal based on the ID and return proposal details.	10
/wallet/exchangecreate	POST	Create a transaction pair. The transaction needs to be broadcasted after signed. Note that successful execution, signing, and broadcast of this API call will deduct 1024 TRX from the user's account.	20
/wallet/exchangeinject	POST	Inject capital into the transaction pair to prevent price fluctuation. The transaction needs to be broadcasted after signed.	20
/wallet/exchangewithdraw	POST	Withdraw a transaction pair. The transaction needs to be broadcasted after signed.	20
/wallet/exchangetransaction	POST	Participate in a transaction pair. The transaction needs to be broadcasted after signed.	20
/wallet/getexchangebyid	POST	Query a transaction pair based on ID.	8

API Method	Type	Description	CU
/wallet/listexchanges	GET	List all transaction pairs.	79
/wallet/gettransactionlistfrom-pending	GET	Get transaction list information from the pending pool.	20
/wallet/gettransactionfrom-pending	POST	Get transaction details from the pending pool.	9
/wallet/getpendingsize	GET	Get the size of the pending pool queue.	7
/wallet/getsignweight	POST	Query the total weight of a signed transaction.	20

Table 2-10 Wallet solidity APIs

API Method	Type	Description	CU
/walletsolidity/gettransactionbyid	POST	Query transaction information by transaction hash (confirmed state).	28
/walletsolidity/gettransactioninfobyid	POST	Return transaction information, including the transaction fee, block height, and VM logs (confirmed state).	10
/walletsolidity/gettransactioninfoby-blocknum	POST	Query transaction information in a block specified (confirmed state).	18
/walletsolidity/getblock	POST	Query block header information or entire block information according to block height or block hash (confirmed state).	392
/walletsolidity/getblockbyid	POST	Query the block by ID, that is, block hash (confirmed state).	376
/walletsolidity/getblockbylatestnum	POST	Query the latest block objects (confirmed state).	965
/walletsolidity/getblockbylimitnext	POST	Query the block objects in the range specified (confirmed state).	376
/walletsolidity/getblockbynum	POST	Query whether a specified block is confirmed.	192
/walletsolidity/getnowblock	GET	Query the latest block information (confirmed state).	542

API Method	Type	Description	CU
/walletsolidity/getaccount	POST	Query the information about an account (confirmed state).	9
/walletsolidity/getdelegatedresourcev2	POST	Query the resource delegation index in Stake 2.0 (confirmed state).	5
/walletsolidity/getavailableunfreezecount	POST	Query the remaining times of executing unstake operations in Stake 2.0 (confirmed state).	5
/walletsolidity/getcanwithdrawunfreezeamount	POST	Query the withdrawable balance at the specified timestamp in Stake 2.0 (confirmed state).	6
/walletsolidity/getdelegatedresourceaccountindexv2	POST	Query the resource delegation index in Stake 2.0 (confirmed state). Two lists will be returned. One lists the target addresses that have received the delegation (toAddress). The other lists the addresses that have delegated resources (fromAddress).	7
/walletsolidity/getburntrx	GET	Query the amount of TRX burned due to on-chain transaction fees since No. 54 Committee Proposal took effect (confirmed state).	5
/walletsolidity/triggerconstantcontract	POST	Invoke the read-only function of a contract (view or pure, for querying confirmed state data), or invoke the non-read-only function of a contract (for predicting whether the transaction can be successfully executed and the estimate energy consumption in confirmed state).	20
/walletsolidity/estimateenergy	POST	Estimate the energy required for the successful execution of smart contract transactions (confirmed state).	13
/walletsolidity/getassetissuebyid	POST	Query the TRC-10 token information by ID (confirmed state).	34
/walletsolidity/getassetissuebyname	POST	Query the TRC-10 token information by token name (confirmed state).	36
/walletsolidity/getassetissuelist	GET	Query the list of all the TRC-10 tokens (confirmed state).	4183
/walletsolidity/getassetissuelistbyname	POST	Query the list of all the TRC-10 tokens with a same name (confirmed state).	495

API Method	Type	Description	CU
/walletsolidity/getpaginatedassetissuelist	POST	Query the list of all the TRC-10 tokens by page (confirmed state).	784
/walletsolidity/listwitnesses	GET	List all witnesses (confirmed state).	111
/walletsolidity/getexchangebyid	POST	Query a transaction pair by ID (confirmed state).	6
/walletsolidity/listexchanges	GET	Query transaction pairs (confirmed state).	79
/walletsolidity/getenergyprices	GET	Query historical energy unit price.	17
/walletsolidity/getcandelegatedmaxsize	POST	Query the amount of delegatable resources share of the specified resource type for an address in Stake 2.0 (confirmed state). The unit is sun.	7
/walletsolidity/gettransactioncountbyblocknum	POST	Query the number of transactions in a block by block number (confirmed state).	18

Table 2-11 JSON-RPC APIs

API Method	Type	Description	CU
eth_accounts	POST	Return an array of addresses owned by the client. An empty list will be returned for tron.	13
eth_blockNumber	POST	Return the latest block number.	10
eth_call	POST	Execute a message call immediately without creating a transaction on the blockchain (triggerConstantContract).	20
eth_chainId	POST	Return the chainId of the TRON network which is the last four bytes of the genesis block hash.	1
eth_estimateGas	POST	Return the required energy using triggerConstantContract.	1000
eth_gasPrice	POST	Return the current energy price in sun.	13

API Method	Type	Description	CU
eth_getBalance	POST	Return the balance of the given account address.	15
eth_getBlockByHash	POST	Return block information for the given block hash.	45
eth_getBlockByNumber	POST	Return block information for the given block number.	20
eth_getBlockTransactionCountByHash	POST	Return the number of transactions in a block by the given block hash.	13
eth_getBlockTransactionCountByNumber	POST	Return the number of transactions in a block by the given block number.	12
eth_getCode	POST	Return the runtime code of a given smart contract address.	40
eth_getStorageAt	POST	Return the value from a storage position at a given address. It can be used to get the value of a variable in a contract.	15
eth_getTransactionByBlockHashAndIndex	POST	Return information about a transaction by block hash and transaction index position.	17
eth_getTransactionByBlockNumberAndIndex	POST	Return information about a transaction by block number and transaction index position.	18
eth_getTransactionByHash	POST	Return the information about a transaction by transaction hash.	40
eth_getTransactionReceipt	POST	Return transaction information, including the transaction fee, block height, and VM logs.	19
eth_syncing	POST	Return the sync status of a node.	1
eth_getLogs	POST	Return all logs matching a given filter object.	75
net_listening	POST	Return if the client is listening for network connections.	1
net_version	POST	Return the hash of the genesis block.	1
web3_clientVersion	POST	Return the current version of the node.	1

API Method	Type	Description	CU
web3_sha3	POST	Return Keccak-256 (not the standardized SHA3-256) of the given data.	5
buildTransaction	POST	Create a transaction. Different transaction types have different parameters.	13

3 Polygon PoS

3.1 Polygon PoS Introduction

Polygon PoS is an L2 solution or a side chain, with its PoS Chains interconnected and communicating with the Ethereum mainnet Polygon PoS supports the most widely used Ethereum scaling ecosystem that offers Ethereum Virtual Machine (EVM) compatibility and an ultimate user experience with fast transactions at near-zero gas fees.

Learn more about Polygon at their **Developer Hub** and from their **Whitepaper**.

NES can enhance the stability and privacy of your blockchain usage and development, while also boosting its overall performance. **Note that Huawei Cloud will never collect your blockchain addresses.**

NOTE

- Supported network
  - HTTP and WebSocket
- **Polygon PoS APIs**

3.2 HTTP Request Examples

3.2.1 Using cURL to Send HTTP API Requests

Request example

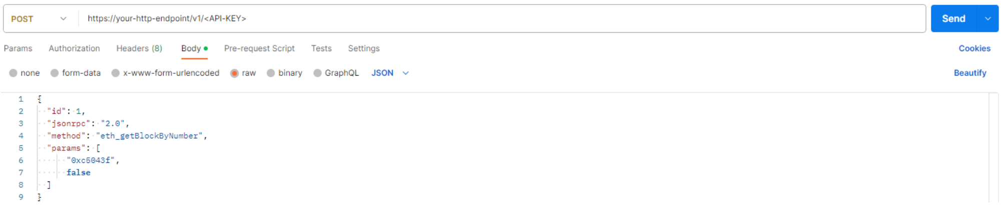
```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

Response example

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "number": "0xc5043f",
        "hash": "0x6c2a069ee47f1fcb83a64c8f3193944199545e91928097d4bc79265f62135040",
        "transactions": [
            "0xd3cee91cc3f05d9228bf2ae99629af0bef22ced9c00fabb7d90be3b1dd029f0b",
            "0xc00960955a51f1d6ea2c10ef0a8fc8b102566d31280e7b22180fe63a9040f178",
            "0x609751af0720d9d12be04ddf7e0aa84c2cc68dcfb7de935325df92bf981a4ccb",
            "0xbbbae84c142947dc8b592d8a69d6cd64249f0514b1736e4f8ae15846171350bc"
        ],
        "difficulty": "0x12",
        "extraData":
"0xd78301091883626f7288676f312e31352e35856c696e7578000000000000000002f70172f7f490
653665c9bfac0666147c8af1f500000000000000000000000000000000000000001127685d6dd6683
085da4b6a041efcef1681e5c9c000000000000000000000000000000000000000021ca971963bdb4b
a2bf337c90660674acff5beb3f00000000000000000000000000000000000000044f856f79f54592a
48c8a1a1fafa1b0a3ac053f9900000000000000000000000000000000000000045973918275c01f5
0555d44e92c9d9b353cadad5400000000000000000000000000000000000000016776786c6590dc
d66b2018db5425749f6334ae6900000000000000000000000000000000000000000172f93a2740e00
112d5f2cef404c0aa16fae21fa400000000000000000000000000000000000000047b5000af8ab69f
d59eb0d4f5762bff57c9c043850000000000000000000000000000000000000000d7c7379531b2aee
82e4ca06d4175d13b9cbeafd4900000000000000000000000000000000000000037fcd58c2d53d98
0b247f1612fdba93e9a76193e6000000000000000000000000000000000000000028e9700392f9246
a6c5b32ee3ecef586f156ed68300000000000000000000000000000000000000000198c27cc3f0301b6
272049dc3f972e2f54278062900000000000000000000000000000000000000001b702f1c9154ac9c
08da247a8e30ee6f2f3373f410000000000000000000000000000000000000001b8bb158b93c94ed
35c1970d610d1e2b34e26652c00000000000000000000000000000000000000003c35649ae99be82
0c7b200a0add09b96d7032d2320000000000000000000000000000000000000001c6869257205e2
0c2a43cb31345db534aecb49f6e000000000000000000000000000000000000001e77bbfd8ed657
20f187efdd109e38d75eaca73850000000000000000000000000000000000000004f84c74dea96df
0ec22e11e7c33996c73fcc2d8220000000000000000000000000000000000000003191dfbf8a52b5a
55b94a6c3a3ffe34a7a8bc395cf1ec37b25f3aa2b585b21cd8375a82941671d5a907a52146f1f3bfd7
56b4b797b844ac9b91c899b887651ff700",
        "gasLimit": "0x1312d00",
        "gasUsed": "0xc6aec",
        "logsBloom":
"0x20208000000400080000000080000400000050008000042002040000000004000000000000400
00000000800000000000000080000000000000000000080240040000000000000000000001000880
0000a000000000004000810001000000000001060000000020000000208000000000860010000000
00000000a0000010240000004000000000000000000020000a00080000025c8000002000800000004
0000000002200000000000000000000080402020000000000020000200000000000000004000000
40200000000000100000000100004000000000000810101000000000000000811001001080000080
800400000000080100100000008000000000000000500000",
        "miner": "0x0000000000000000000000000000000000000000",
        "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
        "nonce": "0x0000000000000000",
        "parentHash":
"0x3d36f634935155891ebcda31688b3b9eda26a339cd7d0ac0b1d22f1d6f0e1ef7",
        "receiptsRoot":
"0x1db35a47efd801643c885906c30dc81430218228690bd7dc23952f9cc2e163d8",
        "sha3Uncles":
"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
        "size": "0xdb8",
        "stateRoot":
"0xfe50dacb41e2de9119bca1ee4f2345c8dc2da79b9171b6f61d858fba40f4cabe",
        "timestamp": "0x606b7a3a",
        "totalDifficulty": "0x6e6d156",
        "transactionsRoot":
"0x89889dccc152914e70a3675868d7784a1a9dbdfb1be7f6943f873ca9d07942a7",
        "uncles": []
    }
}
```

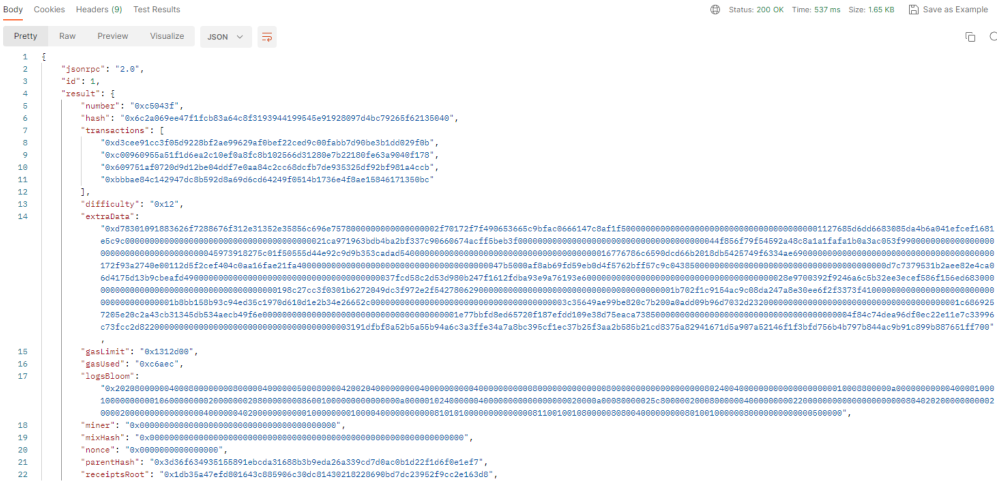
3.2.2 Using Postman to Send HTTP API Requests

Request example



```
{
  "id": 1,
  "jsonrpc": "2.0",
  "method": "eth_getBlockByNumber",
  "params": [
      "0xc5043f",
      false
  ]
}
```

Response example

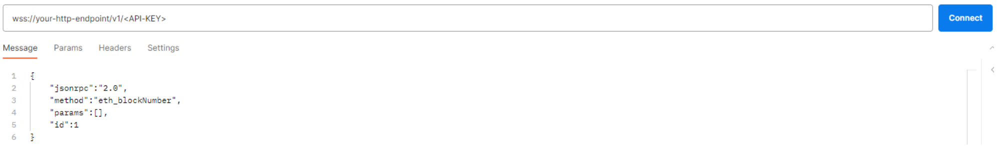


```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "number": "0xc5043f",
        "hash": "0x6c2a069ee47f1fcb83a64c8f3193944199545e91928097d4bc79265f62135040",
        "transactions": [
            "0xd3cee91cc3f05d9228bf2ae99629af0bef22ced9c00fabb7d90be3b1dd029f0b",
            "0xc00960955a51f1d6ea2c10ef0a8fc8b102566d31280e7b22180fe63a9040f178",
            "0x609751af0720d9d12be04ddf7e0aa84c2cc68dcfb7de935325df92bf981a4ccb",
            "0xbbbae84c142947dc8b592d8a69d6cd64249f0514b1736e4f8ae15846171350bc"
        ],
        "difficulty": "0x12",
        "extraData":
            "0xd78301091883626f7288676f312e31352e35856c696e75780000000000000000002f70172f7f490653665c9bfac0666147c8af1f50000000000000000000000000000000000000001127685d6dd6683085da4b6a041efcef1681
            e5c9c00000000000000000000000000000000000000021ca971963bdb4ba2bf337c90660674acff5beb3f00000000000000000000000000000000000000044f856f79f54592a48c8a1a1fafa1b0a3ac053f9900000000000000000
            000000000000000000000045973918275c01f50555d44e92c9d9b353cadad5400000000000000000000000000000000000000016776786c6590dcd66b2018db5425749f6334ae69000000000000000000000000000000000000000
            172f93a2740e00112d5f2cef404c0aa16fae21fa400000000000000000000000000000000000000047b5000af8ab69fd59eb0d4f5762bff57c9c04385000000000000000000000000000000000000000d7c7379531b2aee82e4ca0
            6d4175d13b9cbeafd4900000000000000000000000000000000000000037fcd58c2d53d980b247f1612fdba93e9a76193e600000000000000000000000000000000000000028e9700392f9246a6c5b32ee3ecef586f156ed683000
            000000000000000000000000000000000000198c27cc3f0301b6272049dc3f972e2f5427806290000000000000000000000000000000000000001b702f1c9154ac9c08da247a8e30ee6f2f3373f410000000000000000000000000
            000000000000001b8bb158b93c94ed35c1970d610d1e2b34e26652c0000000000000000000000000000000000000003c35649ae99be820c7b200a0add09b96d7032d2320000000000000000000000000000000000000001c686925
            7205e20c2a43cb31345db534aecb49f6e0000000000000000000000000000000000000001e77bbfd8ed65720f187efdd109e38d75eaca73850000000000000000000000000000000000000004f84c74dea96df0ec22e11e7c33996
            c73fcc2d8220000000000000000000000000000000000000003191dfbf8a52b5a55b94a6c3a3ffe34a7a8bc395cf1ec37b25f3aa2b585b21cd8375a82941671d5a907a52146f1f3bfd756b4b797b844ac9b91c899b887651ff700"
            ,
        "gasLimit": "0x1312d00",
        "gasUsed": "0xc6aec",
        "logsBloom":
            "0x2020800000004000800000000080000040000000500080000420020400000000000040000000000004000080000000000000080000000000000000080240040000000000000000000000100088000000a00000000000400081000
            1000000000010600000000020000000020800000000086001000000000000000a000001024000000040000000000000000020000a0008000025c8000002000800000040000000002200000000000000000000008040202000000000002
            0000200000000000000000040000004020000000000100000000010000400000000081010100000000000000811001001080000008080040000000000801001000080000000000000500000",
        "miner": "0x0000000000000000000000000000000000000000",
        "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
        "nonce": "0x0000000000000000",
        "parentHash": "0x3d36f634935155891ebcda31688b3b9eda26a339cd7d0ac0b1d22f1d6f0e1ef7",
        "receiptsRoot": "0x1db35a47efd801643c885906c30dc81430218228690bd7dc23952f9cc2e163d8",
```

3.3 WebSocket Request Examples

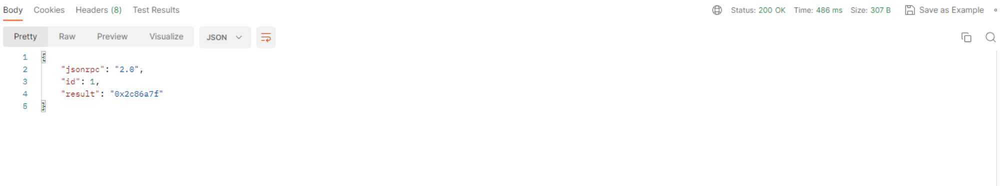
3.3.1 Using Postman to Send JSON-RPC API Requests

Request example



```
{
    "jsonrpc":"2.0",
    "method":"eth_blockNumber",
    "params":[],
    "id":1
}
```

Response example



```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": "0x2c86a7f"
}
```

3.4 Polygon PoS APIs

3.4.1 Ethereum JSON-RPC APIs

3.4.1.1 eth_blocknumber

Introduction

This API returns the latest block number of the blockchain. It consumes 10 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 60,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

An integer value of the latest block number encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_blockNumber","params":[],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.2 eth_getBlockByNumber

Introduction

This API returns information about the block by block number. It consumes 20 CUs. In the dedicated edition, the throughput is 4000 per second for 8 vCPUs and 32 GB memory and 35,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:

- number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
- hash: the hash of the block. It is null if the block is pending.
- parentHash: the hash of the parent block.
- nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
- sha3Uncles: SHA3 of the uncles data in the block.
- logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
- transactionsRoot: the root of the transaction trie of the block.
- stateRoot: the root of the final state trie of the block.
- receiptsRoot: the root of the receipts trie of the block.
- miner: the address of the miner receiving the reward.
- difficulty: the difficulty for this block.
- totalDifficulty: the total difficulty of the chain until this block.
- extraData: the "extra data" field of this block.
- size: the size of this block in bytes.
- gasLimit: the maximum gas allowed in this block.
- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.3 eth_getUncleByBlockNumberAndIndex

Introduction

This API returns information about an uncle of a block by number and uncle index position. It consumes 14 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 51,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number or tag	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Parameter	Type	Description
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockNumberAndIndex","params":["latest","0x0"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.4 eth_getUncleByBlockHashAndIndex

Introduction

This API returns information about an uncle of a block by hash and uncle index position. It consumes 12 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 60,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockHashAndIndex","params":
[0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.5 eth_getUncleCountByBlockNumber

Introduction

This API returns the number of uncles for the block by block number. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.6 eth_getUncleCountByBlockHash

Introduction

This API returns the number of uncles for the block by block hash. It consumes 12 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 58,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.7 eth_getBlockByHash

Introduction

This API returns information about the block by block hash. It consumes 45 CUs. In the dedicated edition, the throughput is 9000 per second for 8 vCPUs and 32 GB memory and 15,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.

  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec",false],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.8 eth_getTransactionByHash

Introduction

This API returns the information about a transaction by transaction hash. It consumes 40 CUs. In the dedicated edition, the throughput is 1500 per second for 8 vCPUs and 32 GB memory and 4000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
  - hash: the hash of the transaction.
  - input: the data sent along with the transaction.
  - nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
  - to: the address of the receiver. It is null for a contract creation transaction.
  - transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
  - value: the value transferred in wei encoded as hexadecimal.

- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://polygon-mainnet.shared-fullnode.bcs.ap-southeast-3.myhuaweicloud.com/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByHash","params":
["0xb142342a7fd70602b7a0ba3688a41bfcbb4fbc3490c252ca48af2594619d220c"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.9 eth_getTransactionCount

Introduction

This API returns the number of transactions sent from an address. It consumes 15 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 46,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Address	String	The address from which the transaction count to be checked.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions sent from an address encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionCount","params":
["0x8D97689C9818892B700e27F316cc3E41e17fBeb9", "latest"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.10 eth_getTransactionByBlockHashAndIndex

Introduction

This API returns information about a transaction by a block hash and transaction index position. It consumes 17 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 43,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
  - hash: the hash of the transaction.
  - input: the data sent along with the transaction.
  - nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
  - to: the address of the receiver. It is null for a contract creation transaction.
  - transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
  - value: the value transferred in wei encoded as hexadecimal.
  - type: the transaction type.
  - accessList: a list of addresses and storage keys that the transaction plans to access.
  - chainId: the transaction chain ID, if any.

- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockHashAndIndex","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec","0x0"],"id":1,"jsonrpc":"2.0"}
'
```

3.4.1.11 eth_getTransactionByBlockNumberAndIndex

Introduction

This API returns information about a transaction by a block number and transaction index position. It consumes 18 CUs. In the dedicated edition, the throughput is 20,000 per second for 8 vCPUs and 32 GB memory and 41,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

- Object: A transaction object with the following fields, or null when no transaction was found:
  - blockHash: the hash of the block where this transaction was in. It is null for a pending log.
  - blockNumber: the number of the block where this transaction was in. It is null for a pending log.
  - from: the address of the sender.
  - gas: the gas provided by the sender in hexadecimal.
  - gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
  - maxFeePerGas: the maximum fee per gas set in the transaction.
  - maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.

- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://polygon-mainnet.shared-fullnode.bcs.ap-southeast-3.myhuaweicloud.com/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockNumberAndIndex","params":["0xc5043f",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.12 eth_getBlockTransactionCountByHash

Introduction

This API returns the number of transactions for the block by block hash. It consumes 13 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 54,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.13 eth_getBlockTransactionCountByNumber

Introduction

This API returns the number of transactions for the block by block number. It consumes 12 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 57,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://polygon-mainnet.shared-fullnode.bcs.ap-southeast-3.myhuaweicloud.com/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.14 eth_getTransactionReceiptsByBlock

Introduction

This API returns all transaction receipts for the given block number or hash. It consumes 1100 CUs. In the dedicated edition, the throughput is 200 per second for 8 vCPUs and 32 GB memory and 650 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number or hash	String	A hexadecimal block number, or the string latest, or the block hash.

Return Value

An array of objects with the following fields:

- Object: A transaction receipt object with the following fields, or null when no transaction receipt was found:

- blockHash: the hash of the block where this transaction was in.
- blockNumber: the block number where this transaction was added encoded as a hexadecimal.
- contractAddress: the contract address created for contract creation. It is null for a transaction that is not for contract creation.
- cumulativeGasUsed: the total gas used when this transaction was executed in the block.
- from: the address of the sender.
- gasUsed: the amount of gas used by this specific transaction alone.
- logs: an array of log objects that generated this transaction.
  - address: the address from which this log was generated.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - data: the 32-byte non-indexed argument of the log.
  - blockNumber: the number of the block where this log was in.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the transactions index position from which this log was created. It is null for a pending log.
  - blockHash: the hash of the block where this log was in.
  - logIndex: the integer of log index position in the block encoded as hexadecimal. It is null for a pending log.
  - removed: true if log was removed due to a chain reorganization and false if the log is valid.
- logsBloom: the bloom filter which is used to retrieve related logs.
- status: 1 (success) or 0 (failure) encoded as a hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionHash: the hash of the transaction.
- transactionIndex: the transaction index position encoded as a hexadecimal.
- type: the value type.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionReceiptsByBlock","params":["latest"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.15 eth_getTransactionReceipt

Introduction

This API returns the receipt of a transaction by transaction hash. It consumes 15 CUs. In the dedicated edition, the throughput is 6000 per second for 8 vCPUs and 32 GB memory and 17,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

- Object: A transaction receipt object with the following fields, or null when no transaction receipt was found:
  - blockHash: the hash of the block where this transaction was in.
  - blockNumber: the block number where this transaction was added encoded as a hexadecimal.
  - contractAddress: the contract address created for contract creation. It is null for a transaction that is not for contract creation.
  - cumulativeGasUsed: the total gas used when this transaction was executed in the block.
  - effectiveGasPrice: the total base charge plus tip paid for each unit of gas.
  - from: the address of the sender.
  - gasUsed: the amount of gas used by this specific transaction alone.
  - logs: an array of log objects that generated this transaction.
    - address: the address from which this log was generated.
    - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
    - data: the 32-byte non-indexed argument of the log.
    - blockNumber: the number of the block where this log was in.
    - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
    - transactionIndex: the transactions index position from which this log was created. It is null for a pending log.
    - blockHash: the hash of the block where this log was in.
    - logIndex: the integer of log index position in the block encoded as hexadecimal. It is null for a pending log.

        - removed: true if log was removed due to a chain reorganization and false if the log is valid.
    - logsBloom: the bloom filter which is used to retrieve related logs.
    - status: 1 (success) or 0 (failure) encoded as a hexadecimal.
    - to: the address of the receiver. It is null for a contract creation transaction.
    - transactionHash: the hash of the transaction.
    - transactionIndex: the transaction index position encoded as a hexadecimal.
    - type: the value type.

Request

```
curl https://polygon-mainnet.shared-fullnode.bcs.ap-southeast-3.myhuaweicloud.com/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionReceipt","params":
["0x6d755989f51032147484162c4dc3d6550552dbd8d3b094fe3c221bfa3c5942b2"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.16 eth_sendRawTransaction

Introduction

This API creates a new message call transaction or creates a contract for signed transactions. It consumes 300 CUs. In the dedicated edition, the throughput is 1000 per second for 8 vCPUs and 32 GB memory and 2400 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Signed transaction data	String	The transaction generated using the private key.

Return Value

The transaction hash, or the zero hash if the transaction is not yet available.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"jsonrpc":"2.0","method":"eth_sendRawTransaction","params":["signed transaction"],"id":1}'
```

3.4.1.17 eth_call

Introduction

This API executes a new message call immediately without creating a transaction on the blockchain. It consumes 20 CUs. In the dedicated edition, the throughput is

15,000 per second for 8 vCPUs and 32 GB memory and 37,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	(Optional) The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	(Optional) The integer of gas provided for the transaction execution.
gasPrice	Integer	(Optional) The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	(Optional) The integer of value sent with this transaction encoded as hexadecimal.
data	String	(Optional) The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The return value of the executed contract method.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_call","params":
[{"from":null,"to":"0x6b175474e89094c44da98b954eedeac495271d0f","data":"0x70a0823100000000000000
0000000006E0d01A76C3Cf4288372a29124A26D4353EE51BE"}, "latest"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.18 eth_createAccessList

Introduction

This API creates an EIP-2930 type accessList based on a given Transaction object. It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles). It consumes 300 CUs. In the dedicated edition, the throughput is 1000 per second for 8 vCPUs and 32 GB memory and 2400 per second for 16 vCPUs and 64 GB memory.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles), plus the estimated gas consumed when the access list is added.

- accessList: a list of objects with the following fields:
  - address: the addresses to be accessed by the transaction.
  - storageKeys: the storage keys to be accessed by the transaction.
- gasUsed: a hexadecimal string representing the approximate gas cost for the transaction if the access list is included.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"method":"eth_createAccessList","params":[{"from":
"0xaeA8F8f781326bfE6A7683C2BD48Dd6AA4d3Ba63", "data": "0x608060806080608155"},
"pending"],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.19 eth_estimateGas

Introduction

This API returns an estimation of gas for a given transaction. It consumes 1000 CUs. In the dedicated edition, the throughput is 50 per second for 8 vCPUs and 32 GB memory and 720 per second for 16 vCPUs and 64 GB memory.

Parameter Description

The parameters are the same as those of eth_call, but they are all optional. If no gas 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 execute the call/transaction when the amount of actual gas needed is higher than the pending block gas limit.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.

Parameter	Type	Description
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

An estimation of gas for a given transaction.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_estimateGas","params":
[{"from":"0x8D97689C9818892B700e27F316cc3E41e17fBeb9","to":"0xd3CdA913deB6f67967B99D67aCDFa1
712C293601","value":"0x186a0"}],"id":1,"jsonrpc":"2.0"}'
```

3.4.1.20 eth_feeHistory

Introduction

This API returns historical gas information. It consumes 17 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 42,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Number of blocks	String/Integer	Number of blocks in the requested range. 1 to 1024 blocks can be requested in a single query. Less than requested may be returned if not all blocks are available.
Newest block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Parameter	Type	Description
Reward percentiles	Integer	(Optional) A monotonically increasing list of percentile values to sample from each block's effective priority fees per gas in ascending order, weighted by gas used.

Return Value

- oldestBlock: the lowest number block of the returned range encoded as hexadecimal.
- baseFeePerGas: an array of block base fees per gas, including an extra block value. The extra value is the next block after the newest block in the returned range. Zeroes are returned for blocks created before EIP-1559.
- gasUsedRatio: an array of block gas used ratios. These are calculated as the ratio of gasUsed and gasLimit.
- reward: an array of effective priority fees per gas data points from a single block. All zeroes are returned if the block is empty.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"id": 1, "jsonrpc": "2.0", "method": "eth_feeHistory", "params": ["0x5", "latest", [20,30]] }'
```

3.4.1.21 eth_maxPriorityFeePerGas

Introduction

This API returns a fee per gas that is an estimate of how much you can pay as a priority fee, or a tip, to get a transaction included in the current block. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 54,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the priority fee needed to be included in a block.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_maxPriorityFeePerGas","id":1}'
```

3.4.1.22 eth_gasPrice

Introduction

This API returns the current gas price in wei. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current gas price in wei.

Request

```
curl https://polygon-mainnet.shared-fullnode.bcs.ap-southeast-3.myhuaweicloud.com/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_gasPrice","params": [],"id":1}'
```

3.4.1.23 eth_getBalance

Introduction

This API returns the balance of the given account address. It consumes 15 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 48,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Address	String	The address to check for balance.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

A hexadecimal value of the current balance in the account at the given address, in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0xc94770007dda54cF92009BFF0dE90c06F603a09f", "latest"],"id":1}'
```

3.4.1.24 eth_getRootHash

Introduction

This API returns the root hash of a specified block range. It consumes 30 CUs. In the dedicated edition, the throughput is 7000 per second for 8 vCPUs and 32 GB memory and 20,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Start block	Integer	The number of the start block.
End block	Integer	The number of the end block.

Return Value

The root hash of a specified block range.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getRootHash","params":[1000, 1032], "id":1}'
```

3.4.1.25 eth_subscribe

Introduction

This API creates a new subscription for particular events. The node returns a subscription ID. For each event that matches the subscription, a notification with relevant data is sent together with the subscription ID. It consumes 10 CUs. In the dedicated edition, the throughput is 1000 per second for 8 vCPUs and 32 GB memory and 1000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Event type	String	The type of event to listen to.

Parameter	Type	Description
Optional parameters	String	Optional parameters to include to describe the type of event to listen to (e.g. newHeads, newPendingTransactions, logs).

Return Value

While the subscription is active, you will receive events formatted as an object described below:

Event Object:

- jsonrpc: always **2.0**.
- method: always **eth_subscription**.
- params: an object with the following fields:
  - subscription: the subscription ID returned by the API that creates this subscription. This ID will be attached to all received events and can also be used to cancel the subscription using eth_unsubscribe.
  - result: an object whose contents vary depending on the event type.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_subscribe",
"params": ["logs"]}'
```

3.4.1.26 eth_unsubscribe

Introduction

This API cancels subscriptions with the subscription ID. It returns a boolean indicating that the subscription was canceled successfully. It consumes 10 CUs. In the dedicated edition, the throughput is 1000 per second for 8 vCPUs and 32 GB memory and 1000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Subscription ID	String	The ID of the subscription you want to unsubscribe.

Return Value

true is returned if a subscription was successfully canceled, or false is returned.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_unsubscribe", "params": ["0x9cef478923ff08bf67fde6c64013158d"]}'
```

3.4.1.27 eth_getStorageAt

Introduction

This API returns the value from a storage position at a given address. It consumes 15 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 47,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address.
Storage position	String	A hexadecimal code of the position in the storage.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns the value at this storage position.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getStorageAt","params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251",
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9", "0x65a8db"],"id":1}'
```

3.4.1.28 eth_accounts

Introduction

This API returns an array of addresses owned by the client. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

An array of addresses owned by the client in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_accounts","params":[],"id":1}'
```

3.4.1.29 eth_getCode

Introduction

This API returns the compiled byte code of a smart contract, if any, at a given address. It consumes 40 CUs. In the dedicated edition, the throughput is 8000 per second for 8 vCPUs and 32 GB memory and 15,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The compiled byte code of the smart contract at the given address.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getCode","params":
["0x06012c8cf97bead5deae237070f9587f8e7a266d", "0x65a8db"],"id":1}'
```

3.4.1.30 eth_getProof

Introduction

This API returns the account and storage values, including the Merkle proof, of the specified account. It consumes 40 CUs. In the dedicated edition, the throughput is 10,000 per second for 8 vCPUs and 32 GB memory and 17,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Storage keys	Array	An array of 32-byte storage keys to be proofed and included.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

- address: the address related to the account.
- accountProof: an array of RLP-serialized MerkleTree-Nodes, starting with the stateRoot-Node, following the path of the SHA3 (address) as key.
- balance: a hexadecimal value of the current balance in wei.
- codeHash: the 32-byte hash of the code of the account.
- nonce: the nonce of the account.
- storageHash: 32 bytes. The SHA3 of the StorageRoot. All storage will deliver a Merkle proof starting with this rootHash.
- storageProof: an array of storage-entries as requested. Each entry is an object with these properties:
  - key: the requested storage key.
  - value: the storage value.
  - proof: an array of RLP-serialized MerkleTree-Nodes, starting with the storageHash-Node, following the path of the SHA3 (key) as path.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc": "2.0","method": "eth_getProof","id": 1,"params":
["0x7F0d15C7FAae65896648C8273B6d7E43f58Fa842",
["0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421"], "latest"]}'
```

3.4.1.31 eth_getLogs

Introduction

This API returns an array of all the logs matching the given filter object. It consumes 75 CUs. In the dedicated edition, the throughput is 10,000 per second for 8 vCPUs and 32 GB memory and 14,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

An array of log objects, or an empty array if nothing has changed since last poll. Log objects contain the following keys and their values:

- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.

- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- blockHash: 32 bytes. The hash of the block where this log was in. It is null for a pending log.
- blockNumber: the block number where this log was in. It is null for a pending log.
- address: 20 bytes. The address from which this log originated.
- data: It contains one or more 32-byte non-indexed arguments of the log.
- topics: an array of 0 to 4 indexed log arguments, each 32 bytes. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getLogs","params":[{"blockHash":
"0x7c5a35e9cb3e8ae0e221ab470abae9d446c3a5626ce6689fc777dcffcab52c70", "topics":
["0x241ea03ca20251805084d27d4440371c34a0b85ff108f6bb5611248f73818b80"]}],"id":74}'
```

3.4.1.32 eth_getFilterChanges

Introduction

The polling method for a filter, which returns an array of logs which occurred since last poll. Call eth_newFilter, eth_newBlockFilter, or eth_newPendingTransactionFilter to create a filter. It consumes 12 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 58,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

- log object array: an array of log objects, or an empty array if nothing has changed since last poll.
- For filters created with eth_newBlockFilter, the return values are block hashes (32 bytes), for example, ["0x3454645634534..."].
- For filters created with eth_newFilter, the logs are objects with the following parameters:
  - address: the address from which this log originated.

- blockHash: the hash of the block where this log was in. It is null for a pending log.
- blockNumber: the number of the block where this log was in. It is null for a pending log.
- data: the non-indexed arguments of the log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":73}'
```

3.4.1.33 eth_getFilterLogs

Introduction

This API returns an array of all the logs matching the given filter ID. It consumes 500 CUs. In the dedicated edition, the throughput is 500 per second for 8 vCPUs and 32 GB memory and 1300 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.

Parameter	Type	Description
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

- Log object array: an array of log objects that match the filter. For an array of logs that occurred since the last poll, use eth_getFilterChanges. Log objects contain the following keys and their values:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
  - removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterLogs","params":["0x16"],"id":74}'
```

3.4.1.34 eth_newBlockFilter

Introduction

This API creates a filter in the node to notify when a new block arrives. It consumes 24 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 30,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newBlockFilter","params":[],"id":73}'
```

3.4.1.35 eth_newFilter

Introduction

This API creates a filter object based on the given filter options to notify when the state changes (logs). It consumes 17 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 41,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newFilter","params":[{"topics":
["0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef"]}],"id":73}'
```

3.4.1.36 eth_newPendingTransactionFilter

Introduction

This API creates a filter in the node to notify when new pending transactions arrive at Polygon. It consumes 24 CUs. In the dedicated edition, the throughput is 50 per second for 8 vCPUs and 32 GB memory and 80 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newPendingTransactionFilter","params":[],"id":73}'
```

3.4.1.37 eth_uninstallFilter

Introduction

This API uninstalls a filter with the given filter ID. It should always be called when watching is no longer needed. Additionally, filters time out when they are not requested with eth_getFilterChanges for a period of time. It consumes 13 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 55,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

true is returned if a filter was successfully uninstalled, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_uninstallFilter","params":["0xb"],"id":73}'
```

3.4.1.38 eth_chainId

Introduction

This API returns the currently configured chain ID. It consumes 1 CU. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 58,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current chain ID.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_chainId","params": [],"id":1}'
```

3.4.1.39 web3_sha3

Introduction

This API returns Keccak-256 (not the standardized SHA3-256) hash of the given data. It consumes 13 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Data	String	Data to be converted.

Return Value

The SHA3 hash of the given string.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"web3_sha3","params": ["0x68656c6c6f20776f726c64"],"id":64}'
```

3.4.1.40 web3_clientVersion

Introduction

This API returns the current version of the chain client. It consumes 13 CUs. In the dedicated edition, the throughput is 25,000 per second for 8 vCPUs and 32 GB memory and 56,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

The current client version.

3.4.2 Polygon JSON-RPC APIs

3.4.2.1 bor_getAuthor

Introduction

This API returns the author address. It consumes 13 CUs. In the dedicated edition, the throughput is 20,000 per second for 8 vCPUs and 32 GB memory and 55,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number or the string latest.

Return Value

The address of the author.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
-X POST \
-H "Content-Type: application/json" \
-d '{"jsonrpc":"2.0","method":"bor_getAuthor","params":["0x1000"], "id":1}'
```

3.4.2.2 bor_getCurrentProposer

Introduction

This API returns the address of the current proposer. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

The address of the current proposer.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
-X POST \
-H "Content-Type: application/json" \
-d '{"jsonrpc":"2.0","method":"bor_getCurrentProposer","params":[], "id":1}'
```

3.4.2.3 bor_getCurrentValidators

Introduction

This API returns the current list of validators. It consumes 19 CUs. In the dedicated edition, the throughput is 15,000 per second for 8 vCPUs and 32 GB memory and 38,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

This method does not accept any parameters.

Return Value

An array of validator objects with the following fields:

- ID: the validator ID.
- accum: the validator's proposer priority.
- power: the validator's voting power.
- signer: the validator address.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
-X POST \
-H "Content-Type: application/json" \
-d '{"jsonrpc":"2.0","method":"bor_getCurrentValidators","params":[], "id":1}'
```

3.4.2.4 bor_getRootHash

Introduction

This API returns the root hash of a specified block. It consumes 13 CUs. In the dedicated edition, the throughput is 30,000 per second for 8 vCPUs and 32 GB memory and 53,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Start block	int	The number of the start block.
End block	int	The number of the end block.

Return Value

The root hash of a specified block range.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
-X POST \
-H "Content-Type: application/json" \
-d '{"jsonrpc":"2.0","method":"bor_getRootHash","params":[1000, 1032], "id":1}'
```

3.4.2.5 bor_getSignersAtHash

Introduction

This API returns all the signers of the block which match the specified block hash. It consumes 16 CUs. In the dedicated edition, the throughput is 20,000 per second for 8 vCPUs and 32 GB memory and 44,000 per second for 16 vCPUs and 64 GB memory.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

An array of all the signers of the block which match the specified block hash.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
-X POST \
-H "Content-Type: application/json" \
-d '{"jsonrpc":"2.0","method":"bor_getSignersAtHash","params":
["0x29fa73e3da83ddac98f527254fe37002e052725a88904bac14f03e919e1e2876"], "id":1}'
```

4 Arbitrum

4.1 Arbitrum Introduction

Arbitrum is a layer 2 scaling solution for the Ethereum blockchain that aims to enhance scalability and reduce transaction fees while maintaining the compatibility and security of the EVM. It uses the Optimistic Rollup protocol to increase Ethereum's transaction throughput and lower transaction costs. Specifically, it packs transactions into a single block, submits them to the Ethereum main chain, and verifies them only when a dispute occurs.

Learn more about Arbitrum at their **Developer Hub** and from their **Whitepaper**.

NES can enhance the stability and privacy of your blockchain usage and development, while also boosting its overall performance. **Note that Huawei Cloud will never collect your blockchain addresses.**

NOTE

- Supported networks
  - HTTP and WebSocket
- **Arbitrum APIs**

4.2 HTTP Request Examples

4.2.1 Using cURL to Send HTTP API Requests

Request example

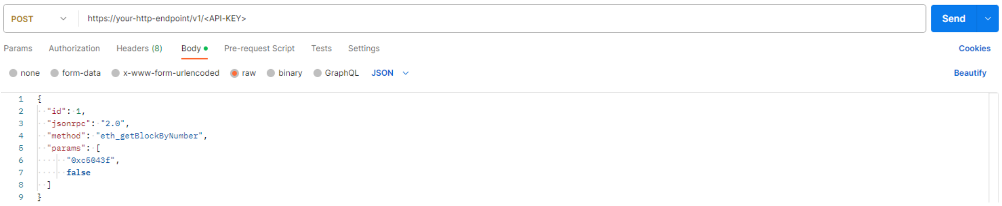
```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

Response example

```
{
   "jsonrpc": "2.0",
   "id": 1,
    "result": {
        "difficulty": "0x0",
        "extraData": "0x",
        "gasLimit": "0xbc87657",
        "gasUsed": "0x3daa0",
        "hash": "0x8f9ecad637559914862de6821bd352d6ac7744d130085d4c5b3d821786aab3ac",
        "l1BlockNumber": "0xe2733e",
        "logsBloom":
"0x0008000000000000008000000000001000000000000000000002000000000000000000000000
008000000000000000440000000000000400000000000080000000000000000000800000000000
000000000000000000000020000000000000000008000000000000000000000000000000000000
000000000004000000000008000000000000400800400000000000000008000000000000000000
000000000000000000000000000000000004000000000000000000400000000000000400000000
000000000000000000000000000000000000400000000000000000000000000800000000001000
0400000000020000000000000000000000000020000000",
        "miner": "0x0000000000000000000000000000000000000000",
        "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
        "nonce": "0x0000000000000000",
        "number": "0xc5043f",
        "parentHash":
"0xf1b5d6a7a45df869b2eef85541584c69bbeb7a58f77e557d4898de2b464d5715",
        "receiptsRoot":
"0x17f8ad0067aff1dbb35d5100b1f131838564b3e980e2b8a2674b8d5362d12e97",
        "sha3Uncles":
"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
        "size": "0x367",
        "stateRoot":
"0x0000000000000000000000000000000000000000000000000000000000000000",
        "timestamp": "0x628dde24",
        "totalDifficulty": "0x0",
        "transactions": [
            {
                "blockHash":
"0x8f9ecad637559914862de6821bd352d6ac7744d130085d4c5b3d821786aab3ac",
                "blockNumber": "0xc5043f",
                "from": "0x1eedcd7c2334463c51b4af75c96b983be7ed4d39",
                "gas": "0xd5d2f",
                "gasPrice": "0x14c30896",
                "hash":
"0x39229c9c973b7675086e4404c30fd23b4130bdb33cef8f799ac0450e2489ba08",
                "input":
"0x0a0e28d400000000000000000000000000000000000000000000000000000000000000200000
000000000000000000000000000000000000000000000000000000040000000000000000000000
000000000000000000000000000000000000000000046a00000000000000000000000000000000
0000000000000000000000000000046900000000000000000000000000000000000000000000000
000000000000000046b000000000000000000000000000000000000000000000000000000000000
5ea2",
                "nonce": "0x324",
                "to": "0x737eaf14061fe68f04ff4ca8205acf538555fcc8",
                "transactionIndex": "0x0",
                "value": "0x0",
                "type": "0x78",
                "v": "0x14985",
                "r": "0x64445cd16ea28f39f5b94d3730147f4f0fde0dd549adfc9c80c8948a3679e878",
                "s": "0x39851deff11a344c3fd67c022bcaf9711a5c8eae60f81d47a977253df9e95e39"
            }
        ],
        "transactionsRoot":
"0xc4b4888ec249430e95f700432d21a3042e5dd2572e68771dc61f73e1fbbd9900",
        "uncles": []
    }
}
```

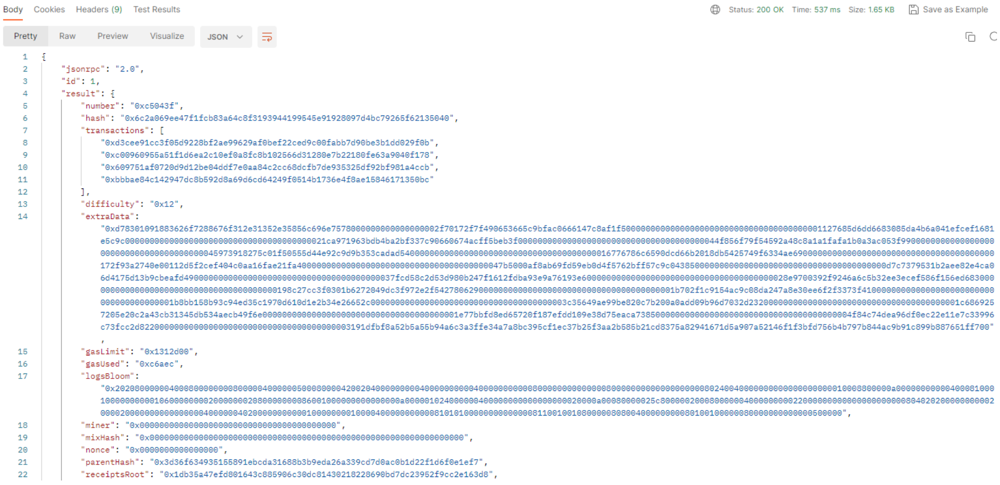
4.2.2 Using Postman to Send HTTP API Requests

Request example



```
{
  "id": 1,
  "jsonrpc": "2.0",
  "method": "eth_getBlockByNumber",
  "params": [
      "0xc5043f",
      false
  ]
}
```

Response example

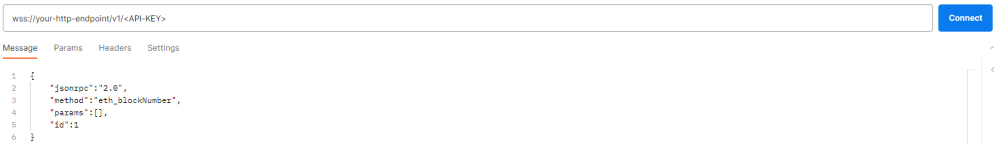


```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "number": "0xc5043f",
        "hash": "0x6c2a069ee47f1fcb83a64c8f3193944199545e91928097d4bc79265f62135040",
        "transactions": [
            "0xd3cee91cc3f05d9228bf2ae99629af0bef22ced9c00fabb7d90be3b1dd029f0b",
            "0xc00960955a51f1d6ea2c10ef0a8fc8b102566d31280e7b22180fe63a9040f178",
            "0x609751af0720d9d12be04ddf7e0aa84c2cc68dcfb7de935325df92bf981a4ccb",
            "0xbbbae84c142947dc8b592d8a69d6cd64249f0514b1736e4f8ae15846171350bc"
        ],
        "difficulty": "0x12",
        "extraData":
            "0xd78301091883626f7288676f312e31352e35856c696e7578000000000000000002f70172f7f490653665c9bfac0666147c8af1f50000000000000000000000000000000000000001127685d6dd6683085da4b6a041efcef1681
            e5c9c00000000000000000000000000000000000000021ca971963bdb4ba2bf337c90660674acff5beb3f00000000000000000000000000000000000000044f856f79f54592a48c8a1a1fafa1b0a3ac053f9900000000000000000
            000000000000000000000045973918275c01f50555d44e92c9d9b353cadad5400000000000000000000000000000000000000016776786c6590dcd66b2018db5425749f6334ae69000000000000000000000000000000000000000
            172f93a2740e00112d5f2cef404c0aa16fae21fa400000000000000000000000000000000000000047b5000af8ab69fd59eb0d4f5762bff57c9c04385000000000000000000000000000000000000000d7c7379531b2aee82e4ca0
            6d4175d13b9cbeafd4900000000000000000000000000000000000000037fcd58c2d53d980b247f1612fdba93e9a76193e600000000000000000000000000000000000000028e9700392f9246a6c5b32ee3ecef586f156ed683000
            000000000000000000000000000000000000198c27cc3f0301b6272049dc3f972e2f5427806290000000000000000000000000000000000000001b702f1c9154ac9c08da247a8e30ee6f2f3373f410000000000000000000000000
            000000000000001b8bb158b93c94ed35c1970d610d1e2b34e26652c0000000000000000000000000000000000000003c35649ae99be820c7b200a0add09b96d7032d2320000000000000000000000000000000000000001c686925
            7205e20c2a43cb31345db534aecb49f6e0000000000000000000000000000000000000001e77bbfd8ed65720f187efdd109e38d75eaca73850000000000000000000000000000000000000004f84c74dea96df0ec22e11e7c33996
            c73fcc2d8220000000000000000000000000000000000000003191dfbf8a52b5a55b94a6c3a3ffe34a7a8bc395cf1ec37b25f3aa2b585b21cd8375a82941671d5a907a52146f1f3bfd756b4b797b844ac9b91c899b887651ff700"
            ,
        "gasLimit": "0x1312d00",
        "gasUsed": "0xc6aec",
        "logsBloom":
            "0x20208000000400080000000080000004000000500080000420020400000000000040000000000004000000000000800000000000000080000000000000000802400400000000000000010008800000a000000000000400081000
            100000000001060000000002000000002080000000086001000000000000000000a000001024000000040000000000000020000a0080000025c800000020008000000040000000000022000000000000000008040202000000000002
            00002000000000000000400000004020000000000100000000010000400000000000008101010000000000000000811001001080000080800400000000801001000000800000000005000000",
        "miner": "0x0000000000000000000000000000000000000000",
        "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
        "nonce": "0x0000000000000000",
        "parentHash": "0x3d36f634935155891ebcda31688b3b9eda26a339cd7d0ac0b1d22f1d6f0e1ef7",
        "receiptsRoot": "0x1db35a47efd801643c885906c30dc81430218228690bd7dc23952f9cc2e163d8",
```

4.3 WebSocket Request Examples

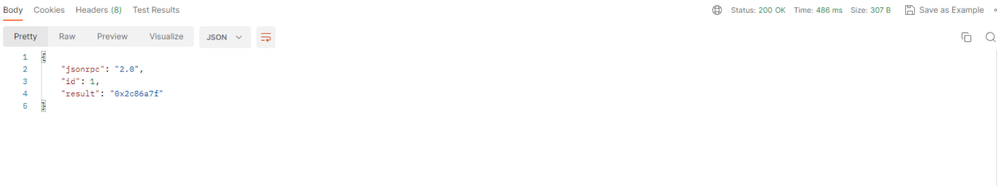
4.3.1 Using Postman to Send JSON-RPC API Requests

Request example



```
{
    "jsonrpc":"2.0",
    "method":"eth_blockNumber",
    "params":[],
    "id":1
}
```

Response example



```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": "0x2c86a7f"
}
```

4.4 Arbitrum APIs

4.4.1 Ethereum JSON-RPC APIs

4.4.1.1 eth_blocknumber

Introduction

This API returns the latest block number of the blockchain. It consumes 13 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An integer value of the latest block number encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_blockNumber","params":[],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.2 eth_getBlockByNumber

Introduction

This API returns information about the block by block number. It consumes 21 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.3 eth_getUncleByBlockNumberAndIndex

Introduction

This API returns information about an uncle of a block by number and uncle index position. It consumes 15 CUs.

Parameter Description

Parameter	Type	Description
Block number or tag	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
  - uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockNumberAndIndex","params":["latest","0x0"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.4 eth_getUncleByBlockHashAndIndex

Introduction

This API returns information about an uncle of a block by hash and uncle index position. It consumes 16 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Uncle index position	String	The uncle's index position in hexadecimal.

Return Value

Object: A block object with the following fields, or null when no block was found:

- number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
- hash: the hash of the block. It is null if the block is pending.
- parentHash: the hash of the parent block.
- nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
- sha3Uncles: SHA3 of the uncles data in the block.
- logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
- transactionsRoot: the root of the transaction trie of the block.
- stateRoot: the root of the final state trie of the block.
- receiptsRoot: the root of the receipts trie of the block.
- miner: the address of the miner receiving the reward.
- difficulty: the difficulty for this block.
- totalDifficulty: the total difficulty of the chain until this block.
- extraData: the "extra data" field of this block.
- size: the size of this block in bytes.
- gasLimit: the maximum gas allowed in this block.
- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleByBlockHashAndIndex","params":
[0xc6ef2fc5426d6ad6fd9e2a26abeab0aa2411b7ab17f30a99d3cb96aed1d1055b",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.5 eth_getUncleCountByBlockNumber

Introduction

This API returns the number of uncles for the block by block number. It consumes 16 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.6 eth_getUncleCountByBlockHash

Introduction

This API returns the number of uncles for the block by block hash. It consumes 16 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.7 eth_getBlockByHash

Introduction

This API returns information about the block by block hash. It consumes 21 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

Object: A block object with the following fields, or null when no block was found:

- number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
- hash: the hash of the block. It is null if the block is pending.
- parentHash: the hash of the parent block.
- nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
- sha3Uncles: SHA3 of the uncles data in the block.
- logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
- transactionsRoot: the root of the transaction trie of the block.
- stateRoot: the root of the final state trie of the block.
- receiptsRoot: the root of the receipts trie of the block.
- miner: the address of the miner receiving the reward.
- difficulty: the difficulty for this block.
- totalDifficulty: the total difficulty of the chain until this block.
- extraData: the "extra data" field of this block.
- size: the size of this block in bytes.
- gasLimit: the maximum gas allowed in this block.
- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.

- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec",false],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.8 eth_getTransactionByHash

Introduction

This API returns the information about a transaction by transaction hash. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.
- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.

- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByHash","params":
["0xb142342a7fd70602b7a0ba3688a41bfcbb4fbc3490c252ca48af2594619d220c"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.9 eth_getTransactionCount

Introduction

This API returns the number of transactions sent from an address. It consumes 26 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address from which the transaction count to be checked.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions sent from an address encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionCount","params":
["0x8D97689C9818892B700e27F316cc3E41e17fBeb9", "latest"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.10 eth_getTransactionByBlockHashAndIndex

Introduction

This API returns information about a transaction by a block hash and transaction index position. It consumes 16 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.
- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockHashAndIndex","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec","0x0"],"id":1,"jsonrpc":"2.0"}
'
```

4.4.1.11 eth_getTransactionByBlockNumberAndIndex

Introduction

This API returns information about a transaction by a block number and transaction index position. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.
- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.

- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockNumberAndIndex","params":["0xc5043f",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.12 eth_getBlockTransactionCountByHash

Introduction

This API returns the number of transactions for the block by block hash. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.13 eth_getBlockTransactionCountByNumber

Introduction

This API returns the number of transactions for the block by block number. It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
 -X POST \
 -H "Content-Type: application/json" \
 --data '{"method":"eth_getBlockTransactionCountByNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.14 eth_syncing

Introduction

This API returns the sync status. It consumes 1 CU.

Parameter Description

This method does not accept any parameters.

Return Value

An array of objects with the following fields:

Return value one:

Boolean: false is returned if synchronization is complete.

Return value two:

Object: the sync status is returned if the synchronization is in progress.

- startingBlock: the block at which the import started, encoded as hexadecimal.
- currentBlock: the current block, same as **eth_blockNumber**, encoded as hexadecimal.
- highestBlock: the estimated highest block, encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
   -X POST \
   -H "Content-Type: application/json" \
   -d '{"jsonrpc":"2.0","method":"eth_syncing","params": [],"id":1}'
```

4.4.1.15 eth_getTransactionReceipt

Introduction

This API returns the receipt of a transaction by transaction hash. It consumes 17 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

Object: A transaction receipt object with the following fields, or null when no transaction receipt was found:

- blockHash: the hash of the block where this transaction was in.
- blockNumber: the block number where this transaction was added encoded as a hexadecimal.
- contractAddress: the contract address created for contract creation. It is null for a transaction that is not for contract creation.
- cumulativeGasUsed: the total gas used when this transaction was executed in the block.
- effectiveGasPrice: the total base charge plus tip paid for each unit of gas.
- from: the address of the sender.
- gasUsed: the amount of gas used by this specific transaction alone.
- logs: an array of log objects that generated this transaction.
  - address: the address from which this log was generated.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - data: the 32-byte non-indexed argument of the log.
  - blockNumber: the number of the block where this log was in.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the transactions index position from which this log was created. It is null for a pending log.
  - blockHash: the hash of the block where this log was in.
  - logIndex: the integer of log index position in the block encoded as hexadecimal. It is null for a pending log.
  - removed: true if log was removed due to a chain reorganization and false if the log is valid.
- logsBloom: the bloom filter which is used to retrieve related logs.
- status: 1 (success) or 0 (failure) encoded as a hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionHash: the hash of the transaction.
- transactionIndex: the transaction index position encoded as a hexadecimal.
- type: the value type.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionReceipt","params":
["0x6d755989f51032147484162c4dc3d6550552dbd8d3b094fe3c221bfa3c5942b2"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.16 eth_sendRawTransaction

Introduction

This API creates a new message call transaction or creates a contract for signed transactions. It consumes 308 CUs.

Parameter Description

Parameter	Type	Description
Signed transaction data	String	The transaction generated using the private key.

Return Value

The transaction hash, or the zero hash if the transaction is not yet available.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"jsonrpc":"2.0","method":"eth_sendRawTransaction","params":["signed transaction"],"id":1}'
```

4.4.1.17 eth_call

Introduction

This API executes a new message call immediately without creating a transaction on the blockchain. It consumes 41 CUs.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	(Optional) The address from which the transaction is sent.
to	String	The address to which the transaction is directed.

Parameter	Type	Description
gas	Integer	(Optional) The integer of gas provided for the transaction execution.
gasPrice	Integer	(Optional) The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	(Optional) The integer of value sent with this transaction encoded as hexadecimal.
data	String	(Optional) The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The return value of the executed contract method.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_call","params":
[{"from":null,"to":"0x6b175474e89094c44da98b954eedeac495271d0f","data":"0x70a08231000000000000000
00000000006E0d01A76C3Cf4288372a29124A26D4353EE51BE"}, "latest"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.18 eth_createAccessList

Introduction

This API creates an EIP-2930 type accessList based on a given Transaction object. It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles). It consumes 44 CUs.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns a list of addresses and storage keys that are read and written by the transaction (except the sender account and precompiles), plus the estimated gas consumed when the access list is added.

accessList: a list of objects with the following fields:

- address: the addresses to be accessed by the transaction.
- storageKeys: the storage keys to be accessed by the transaction.
- gasUsed: a hexadecimal string representing the approximate gas cost for the transaction if the access list is included.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"method":"eth_createAccessList","params":[{"from":
"0xaeA8F8f781326bfE6A7683C2BD48Dd6AA4d3Ba63", "data": "0x608060806080608155"},
"pending"],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.19 eth_estimateGas

Introduction

This API returns an estimation of gas for a given transaction. It consumes 87 CUs.

Parameter Description

The parameters are the same as those of eth_call, but they are all optional. If no gas 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 execute the call/transaction when the amount of actual gas needed is higher than the pending block gas limit.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

An estimation of gas for a given transaction.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_estimateGas","params":
[{"from":"0x8D97689C9818892B700e27F316cc3E41e17fBeb9","to":"0xd3CdA913deB6f67967B99D67aCDFa1
712C293601","value":"0x186a0"}],"id":1,"jsonrpc":"2.0"}'
```

4.4.1.20 eth_feeHistory

Introduction

This API returns historical gas information. It consumes 32 CUs.

Parameter Description

Parameter	Type	Description
Number of blocks	String/Integer	Number of blocks in the requested range. 1 to 1024 blocks can be requested in a single query. Less than requested may be returned if not all blocks are available.
Newest block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Reward percentiles	Integer	(Optional) A monotonically increasing list of percentile values to sample from each block's effective priority fees per gas in ascending order, weighted by gas used.

Return Value

- oldestBlock: the lowest number block of the returned range encoded as hexadecimal.
- baseFeePerGas: an array of block base fees per gas, including an extra block value. The extra value is the next block after the newest block in the returned range. Zeroes are returned for blocks created before EIP-1559.
- gasUsedRatio: an array of block gas used ratios. These are calculated as the ratio of gasUsed and gasLimit.
- reward: an array of effective priority fees per gas data points from a single block. All zeroes are returned if the block is empty.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"id": 1, "jsonrpc": "2.0", "method": "eth_feeHistory", "params": ["0x5", "latest", [20,30]] }'
```

4.4.1.21 eth_maxPriorityFeePerGas

Introduction

This API returns a fee per gas that is an estimate of how much you can pay as a priority fee, or a tip, to get a transaction included in the current block. It consumes 13 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the priority fee needed to be included in a block.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
   -X POST \
   -H "Content-Type: application/json" \
   -d '{"jsonrpc":"2.0","method":"eth_maxPriorityFeePerGas","id":1}'
```

4.4.1.22 eth_gasPrice

Introduction

This API returns the current gas price in wei. It consumes 19 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current gas price in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_gasPrice","params": [],"id":1}'
```

4.4.1.23 eth_getBalance

Introduction

This API returns the balance of the given account address. It consumes 19 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address to check for balance.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

A hexadecimal value of the current balance in the account at the given address, in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0xc94770007dda54cF92009BFF0dE90c06F603a09f", "latest"],"id":1}'
```

4.4.1.24 eth_subscribe

Introduction

This API creates a new subscription for particular events. The node returns a subscription ID. For each event that matches the subscription, a notification with relevant data is sent together with the subscription ID. It consumes 10 CUs.

Parameter Description

Parameter	Type	Description
Event type	String	The type of event to listen to.
Optional parameters	String	Optional parameters to include to describe the type of event to listen to (e.g. newHeads, newPendingTransactions, logs).

Return Value

While the subscription is active, you will receive events formatted as an object described below:

Event Object:

- jsonrpc: always **2.0**.
- method: always **eth_subscription**.
- params: an object with the following fields:
  - subscription: the subscription ID returned by the API that creates this subscription. This ID will be attached to all received events and can also be used to cancel the subscription using eth_unsubscribe.
  - result: an object whose contents vary depending on the event type.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_subscribe",
"params": ["logs"]}'
```

4.4.1.25 eth_unsubscribe

Introduction

This API cancels subscriptions with the subscription ID. It returns a boolean indicating that the subscription was canceled successfully. It consumes 10 CUs.

Parameter Description

Parameter	Type	Description
Subscription ID	String	The ID of the subscription you want to unsubscribe.

Return Value

true is returned if a subscription was successfully canceled, or false is returned.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_unsubscribe",
"params": ["0x9cef478923ff08bf67fde6c64013158d"]}'
```

4.4.1.26 eth_getStorageAt

Introduction

This API returns the value from a storage position at a given address. It consumes 18 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address.
Storage position	String	A hexadecimal code of the position in the storage.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns the value at this storage position.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getStorageAt","params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251",
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9", "0x65a8db"],"id":1}'
```

4.4.1.27 eth_accounts

Introduction

This API returns an array of addresses owned by the client. It consumes 12 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An array of addresses owned by the client in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_accounts","params":[],"id":1}'
```

4.4.1.28 eth_getCode

Introduction

This API returns the compiled byte code of a smart contract, if any, at a given address. It consumes 19 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The compiled byte code of the smart contract at the given address.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getCode","params":
["0x06012c8cf97bead5deae237070f9587f8e7a266d", "0x65a8db"],"id":1}'
```

4.4.1.29 eth_getProof

Introduction

This API returns the account and storage values, including the Merkle proof, of the specified account. It consumes 34 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Storage keys	Array	An array of 32-byte storage keys to be proofed and included.

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

- address: the address related to the account.
- accountProof: an array of RLP-serialized MerkleTree-Nodes, starting with the stateRoot-Node, following the path of the SHA3 (address) as key.
- balance: a hexadecimal value of the current balance in wei.
- codeHash: the 32-byte hash of the code of the account.
- nonce: the nonce of the account.
- storageHash: 32 bytes. The SHA3 of the StorageRoot. All storage will deliver a Merkle proof starting with this rootHash.
- storageProof: an array of storage-entries as requested. Each entry is an object with these properties:
  - key: the requested storage key.
  - value: the storage value.
  - proof: an array of RLP-serialized MerkleTree-Nodes, starting with the storageHash-Node, following the path of the SHA3 (key) as path.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc": "2.0","method": "eth_getProof","id": 1,"params":
["0x7F0d15C7FAae65896648C8273B6d7E43f58Fa842",
["0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421"], "latest"]}'
```

4.4.1.30 eth_getLogs

Introduction

This API returns an array of all the logs matching the given filter object. It consumes 75 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.

Parameter	Type	Description
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

An array of log objects, or an empty array if nothing has changed since last poll. Log objects contain the following keys and their values:

- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- blockHash: 32 bytes. The hash of the block where this log was in. It is null for a pending log.
- blockNumber: the block number where this log was in. It is null for a pending log.

- address: 20 bytes. The address from which this log originated.
- data: It contains one or more 32-byte non-indexed arguments of the log.
- topics: an array of 0 to 4 indexed log arguments, each 32 bytes. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getLogs","params":[{"blockHash":
"0x7c5a35e9cb3e8ae0e221ab470abae9d446c3a5626ce6689fc777dcffcab52c70", "topics":
["0x241ea03ca20251805084d27d4440371c34a0b85ff108f6bb5611248f73818b80"]}],"id":74}'
```

4.4.1.31 eth_getFilterChanges

Introduction

The polling method for a filter, which returns an array of logs which occurred since last poll. Call eth_newFilter, eth_newBlockFilter, or eth_newPendingTransactionFilter to create a filter. It consumes 26 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

- log object array: an array of log objects, or an empty array if nothing has changed since last poll.
- For filters created with eth_newBlockFilter, the return values are block hashes (32 bytes), for example, ["0x3454645634534..."].
- For filters created with eth_newFilter, the logs are objects with the following parameters:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
  - removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g.

Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.

- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":73}'
```

4.4.1.32 eth_getFilterLogs

Introduction

This API returns an array of all the logs matching the given filter ID. It consumes 75 CUs.

Parameter Description

Paramete r	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

- Log object array: an array of log objects that match the filter. For an array of logs that occurred since the last poll, use eth_getFilterChanges. Log objects contain the following keys and their values:

- address: the address from which this log originated.
- blockHash: the hash of the block where this log was in. It is null for a pending log.
- blockNumber: the number of the block where this log was in. It is null for a pending log.
- data: the non-indexed arguments of the log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
- transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterLogs","params":["0x16"],"id":74}'
```

4.4.1.33 eth_newBlockFilter

Introduction

This API creates a filter in the node to notify when a new block arrives. It consumes 20 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newBlockFilter","params":[],"id":73}'
```

4.4.1.34 eth_newFilter

Introduction

This API creates a filter object based on the given filter options to notify when the state changes (logs). It consumes 20 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newFilter","params":[{"topics":
["0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef"]}],"id":73}'
```

4.4.1.35 eth_newPendingTransactionFilter

Introduction

This API creates a filter in the node to notify when new pending transactions arrive at Arbitrum. It consumes 20 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newPendingTransactionFilter","params":[],"id":73}'
```

4.4.1.36 eth_uninstallFilter

Introduction

This API uninstalls a filter with the given filter ID. It should always be called when watching is no longer needed. Additionally, filters time out when they are not requested with eth_getFilterChanges for a period of time. It consumes 12 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

true is returned if a filter was successfully uninstalled, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_uninstallFilter","params":["0xb"],"id":73}'
```

4.4.1.37 eth_chainId

Introduction

This API returns the currently configured chain ID. It consumes 1 CU.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current chain ID.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_chainId","params": [],"id":1}'
```

4.4.1.38 web3_sha3

Introduction

This API returns Keccak-256 (not the standardized SHA3-256) hash of the given data. It consumes 15 CUs.

Parameter Description

Parameter	Type	Description
Data	String	Data to be converted.

Return Value

The SHA3 hash of the given string.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"web3_sha3","params": ["0x68656c6c6f20776f726c64"],"id":64}'
```

4.4.1.39 web3_clientVersion

Introduction

This API returns the current version of the chain client. It consumes 15 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

The current client version.

5 BNB Smart Chain

5.1 BNB Smart Chain Introduction

BNB Smart Chain (BSC) is a blockchain that runs in parallel with the Binance Chain. It supports EVM-compatible smart contracts and protocols. BSC relies on a system of 55 validators with Proof of Staked Authority (PoSA) consensus that can support short block time and lower fees. The design goal here is to leave the high throughput of BNB Beacon Chain intact while introducing smart contracts into its ecosystem.

Learn more about BNB Smart Chain in their **Documentation**.

NES can enhance the stability and privacy of your blockchain usage and development, while also boosting its overall performance. **Note that Huawei Cloud will never collect your blockchain addresses.**

NOTE

- Supported networks
  - HTTP and WebSocket
- **BNB Smart Chain APIs**

5.2 HTTP Request Examples

5.2.1 Using cURL to Send HTTP API Requests

Request example

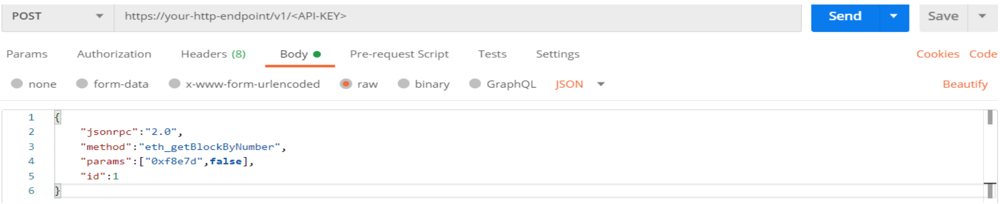
```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xf8e7d",false],"id":1,"jsonrpc":"2.0"}'
```

Response example

```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": {
        "baseFeePerGas": "0x0",
        "difficulty": "0x2",
        "extraData":
"0xd883010300846765746888676f312e32302e31856c696e7578000000394f7f55f8b003b860a1c4
3902a0f6cc97fc61995d28ddb7074edf5c69fe4c503b5194c152cf3b827de2e8b32f6a5d1e6aab282e
84e46ee978157c85676c3fb964e90a42ef49ebfc564ae6b863f41f95228784da1315d5f1ba50749b3
9dbdc0212db7e7920519b985ef84a830f8e7ba05fbae7c490271975e55112f31add5c69a35a46c2b5
3f1629f18b477f9a8ef6ee830f8e7ca08384763b6b4cab00a7a9dd0de27e8e319d10fd66ce0d16708
b70f4264c8f2c6180e801644243b324bb502cf301ca933d84f26895aa7607dbf425c9a12af0b48dad
1ee634fddcd28fa0ca33d6b007ef8da8b26f224c81d5876fce13c46bf8fa78b400",
        "gasLimit": "0x8583b00",
        "gasUsed": "0x0",
        "hash": "0x7f222d3f1a7c664fc8709361e0f0d0e60ae63bfcd9f770e7892f043c8885b167",
        "logsBloom":
"0x0000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000",
        "miner": "0x69cb38199d2c2419b384fa6e22f4667069843730",
        "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
        "nonce": "0x0000000000000000",
        "number": "0xf8e7d",
        "parentHash":
"0x8384763b6b4cab00a7a9dd0de27e8e319d10fd66ce0d16708b70f4264c8f2c61",
        "receiptsRoot":
"0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
        "sha3Uncles":
"0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
        "size": "0x316",
        "stateRoot":
"0xbb246175024b46f7c985032bba83d3d5aa42ea708e8f8b90834d8445aa3ab651",
        "timestamp": "0x655d6eca",
        "totalDifficulty": "0x1f1b52",
        "transactions": [],
        "transactionsRoot":
"0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
        "uncles": []
    }
}
```

5.2.2 Using Postman to Send HTTP API Requests

Request example



https://your-http-endpoint/v1/<API-KEY>

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

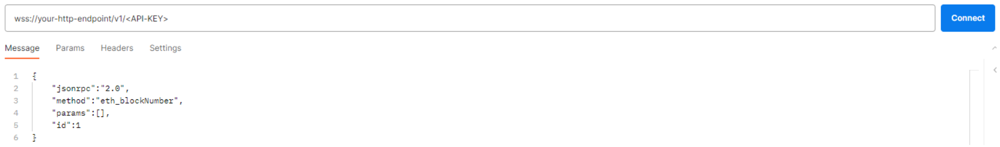
```
{
    "jsonrpc":"2.0",
    "method":"eth_getBlockByNumber",
    "params":["0xf8e7d",false],
    "id":1
}
```

Response example

5.3 WebSocket Request Examples

5.3.1 Using Postman to Send JSON-RPC API Requests

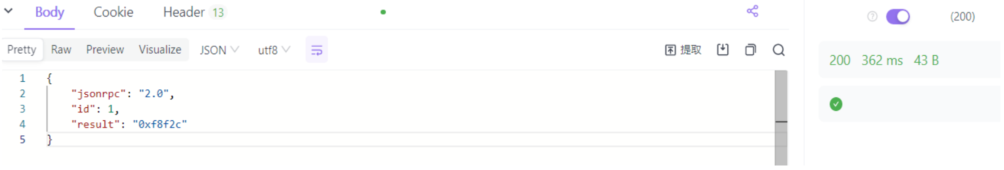
Request example



wss://your-http-endpoint/v1/<API-KEY>

```
{
    "jsonrpc":"2.0",
    "method":"eth_blockNumber",
    "params":[],
    "id":1
}
```

Response example



```
{
    "jsonrpc": "2.0",
    "id": 1,
    "result": "0xf8f2c"
}
```

200 362 ms 43 B

5.4 BNB Smart Chain APIs

5.4.1 Ethereum JSON-RPC APIs

5.4.1.1 eth_blocknumber

Introduction

This API returns the latest block number of the blockchain. It consumes 105 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An integer value of the latest block number encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_blockNumber","params":[],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.2 eth_getBlockByNumber

Introduction

This API returns information about the block by block number. It consumes 133 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

- Object: A block object with the following fields, or null when no block was found:
  - number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.
  - hash: the hash of the block. It is null if the block is pending.
  - parentHash: the hash of the parent block.
  - nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
  - sha3Uncles: SHA3 of the uncles data in the block.
  - logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
  - transactionsRoot: the root of the transaction trie of the block.
  - stateRoot: the root of the final state trie of the block.
  - receiptsRoot: the root of the receipts trie of the block.
  - miner: the address of the miner receiving the reward.
  - difficulty: the difficulty for this block.
  - totalDifficulty: the total difficulty of the chain until this block.
  - extraData: the "extra data" field of this block.
  - size: the size of this block in bytes.
  - gasLimit: the maximum gas allowed in this block.
  - gasUsed: the total used gas by all transactions in this block.
  - timestamp: the Unix timestamp for when the block was collated.
  - transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.

- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByNumber","params":["0xc5043f",false],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.3 eth_hashrate

Introduction

This API returns the number of hashes per second calculated by the node. It consumes 104 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

The number of hashes per second encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_hashrate","params":[],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.4 eth_getUncleCountByBlockNumber

Introduction

This API returns the number of uncles for the block by block number. It consumes 130 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.5 eth_getUncleCountByBlockHash

Introduction

This API returns the number of uncles for the block by block hash. It consumes 136 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of uncles in the block encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getUncleCountByBlockHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.6 eth_getBlockByHash

Introduction

This API returns information about the block by block hash. It consumes 145 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.
Transaction details	Bool	If true, it returns the detail of each transaction. If false, it returns only the hashes of the transactions.

Return Value

Object: A block object with the following fields, or null when no block was found:

- number: the block number of the requested block, encoded as hexadecimal. It is null if the block is pending.

- hash: the hash of the block. It is null if the block is pending.
- parentHash: the hash of the parent block.
- nonce: the hash used to demonstrate proof-of-work. It is null if the block is pending.
- sha3Uncles: SHA3 of the uncles data in the block.
- logsBloom: the bloom filter for the logs of the block. It is null if the block is pending.
- transactionsRoot: the root of the transaction trie of the block.
- stateRoot: the root of the final state trie of the block.
- receiptsRoot: the root of the receipts trie of the block.
- miner: the address of the miner receiving the reward.
- difficulty: the difficulty for this block.
- totalDifficulty: the total difficulty of the chain until this block.
- extraData: the "extra data" field of this block.
- size: the size of this block in bytes.
- gasLimit: the maximum gas allowed in this block.
- gasUsed: the total used gas by all transactions in this block.
- timestamp: the Unix timestamp for when the block was collated.
- transactions: an array of transaction objects, or 32-bytes transaction hashes, depending on the last given parameter.
- uncles: an array of uncle hashes.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec",false],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.7 eth_getTransactionByHash

Introduction

This API returns the information about a transaction by transaction hash. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.
- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByHash","params":
["0xb142342a7fd70602b7a0ba3688a41bfcbb4fbc3490c252ca48af2594619d220c"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.8 eth_getTransactionCount

Introduction

This API returns the number of transactions sent from an address. It consumes 148 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address from which the transaction count to be checked.

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions sent from an address encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionCount","params":
["0x8D97689C9818892B700e27F316cc3E41e17fBeb9", "latest"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.9 eth_getTransactionByBlockHashAndIndex

Introduction

This API returns information about a transaction by a block hash and transaction index position. It consumes 149 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.

- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockHashAndIndex","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec","0x0"],"id":1,"jsonrpc":"2.0"}
'
```

5.4.1.10 eth_getTransactionByBlockNumberAndIndex

Introduction

This API returns information about a transaction by a block number and transaction index position. It consumes 137 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Index	String	The transaction index position encoded as a hexadecimal.

Return Value

Object: A transaction object with the following fields, or null when no transaction was found:

- blockHash: the hash of the block where this transaction was in. It is null for a pending log.
- blockNumber: the number of the block where this transaction was in. It is null for a pending log.
- from: the address of the sender.
- gas: the gas provided by the sender in hexadecimal.
- gasPrice: the gas price provided by the sender in wei encoded as hexadecimal.
- maxFeePerGas: the maximum fee per gas set in the transaction.
- maxPriorityFeePerGas: the maximum priority gas fee set in the transaction.
- hash: the hash of the transaction.
- input: the data sent along with the transaction.
- nonce: the number of transactions made by the sender prior to this one encoded as hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionIndex: the integer of the transactions index position from which this log was created. It is null for a pending log.
- value: the value transferred in wei encoded as hexadecimal.
- type: the transaction type.
- accessList: a list of addresses and storage keys that the transaction plans to access.
- chainId: the transaction chain ID, if any.
- v: the standard V field of the signature.
- r: the R field of the signature.
- s: the S field of the signature.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionByBlockNumberAndIndex","params":["0xc5043f",
"0x0"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.11 eth_getBlockTransactionCountByHash

Introduction

This API returns the number of transactions for the block by block hash. It consumes 143 CUs.

Parameter Description

Parameter	Type	Description
Block hash	String	The hash of a block.

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByHash","params":
["0x81e807e7a6031d9f103eeee2a2edc5994c3432ee1e3227c66ff78eef30ea1dec"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.12 eth_getBlockTransactionCountByNumber

Introduction

This API returns the number of transactions for the block by block number. It consumes 128 CUs.

Parameter Description

Parameter	Type	Description
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The number of transactions associated with a specific block, in hexadecimal value.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getBlockTransactionCountByNumber","params":["0xc5043f"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.13 eth_syncing

Introduction

This API returns the sync status. It consumes 118 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An array of objects with the following fields:

Return value one:

Boolean: false is returned if synchronization is complete.

Return value two:

Object: the sync status is returned if the synchronization is in progress.

- startingBlock: the block at which the import started, encoded as hexadecimal.
- currentBlock: the current block, same as **eth_blocknumber**, encoded as hexadecimal.
- highestBlock: the estimated highest block, encoded as hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
   -X POST \
   -H "Content-Type: application/json" \
   -d '{"jsonrpc":"2.0","method":"eth_syncing","params": [],"id":1}'
```

5.4.1.14 eth_getTransactionReceipt

Introduction

This API returns the receipt of a transaction by transaction hash. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Transaction hash	String	The hash of a transaction.

Return Value

Object: A transaction receipt object with the following fields, or null when no transaction receipt was found:

- blockHash: the hash of the block where this transaction was in.
- blockNumber: the block number where this transaction was added encoded as a hexadecimal.
- contractAddress: the contract address created for contract creation. It is null for a transaction that is not for contract creation.
- cumulativeGasUsed: the total gas used when this transaction was executed in the block.
- effectiveGasPrice: the total base charge plus tip paid for each unit of gas.
- from: the address of the sender.
- gasUsed: the amount of gas used by this specific transaction alone.
- logs: an array of log objects that generated this transaction.
  - address: the address from which this log was generated.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g.

  Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - data: the 32-byte non-indexed argument of the log.
  - blockNumber: the number of the block where this log was in.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the transactions index position from which this log was created. It is null for a pending log.
  - blockHash: the hash of the block where this log was in.
  - logIndex: the integer of log index position in the block encoded as hexadecimal. It is null for a pending log.
  - removed: true if log was removed due to a chain reorganization and false if the log is valid.
- logsBloom: the bloom filter which is used to retrieve related logs.
- status: 1 (success) or 0 (failure) encoded as a hexadecimal.
- to: the address of the receiver. It is null for a contract creation transaction.
- transactionHash: the hash of the transaction.
- transactionIndex: the transaction index position encoded as a hexadecimal.
- type: the value type.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_getTransactionReceipt","params":
["0x6d755989f51032147484162c4dc3d6550552dbd8d3b094fe3c221bfa3c5942b2"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.15 eth_sendRawTransaction

Introduction

This API creates a new message call transaction or creates a contract for signed transactions. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Signed transaction data	String	The transaction generated using the private key.

Return Value

The transaction hash, or the zero hash if the transaction is not yet available.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"jsonrpc":"2.0","method":"eth_sendRawTransaction","params":["signed transaction"],"id":1}'
```

5.4.1.16 eth_call

Introduction

This API executes a new message call immediately without creating a transaction on the blockchain. It consumes 120 CUs.

Parameter Description

It consists of transaction-related fields and the block number.

Parameter	Type	Description
from	String	(Optional) The address from which the transaction is sent.
to	String	The address to which the transaction is directed.
gas	Integer	(Optional) The integer of gas provided for the transaction execution.
gasPrice	Integer	(Optional) The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	(Optional) The integer of value sent with this transaction encoded as hexadecimal.
data	String	(Optional) The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The return value of the executed contract method.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_call","params":
[{"from":null,"to":"0x6b175474e89094c44da98b954eedeac495271d0f","data":"0x70a0823100000000000000
00000000006E0d01A76C3Cf4288372a29124A26D4353EE51BE"}, "latest"],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.17 eth_mining

Introduction

This API returns true if the node is actively mining new blocks. It consumes 99 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

true is returned if the node is actively mining new blocks, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"method":"eth_mining","params":[],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.18 eth_estimateGas

Introduction

This API returns an estimation of gas for a given transaction. It consumes 120 CUs.

Parameter Description

The parameters are the same as those of eth_call, but they are all optional. If no gas 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 execute the call/ transaction when the amount of actual gas needed is higher than the pending block gas limit.

Parameter	Type	Description
from	String	The address from which the transaction is sent.
to	String	The address to which the transaction is directed.

Parameter	Type	Description
gas	Integer	The integer of gas provided for the transaction execution.
gasPrice	Integer	The integer of gasPrice used for each paid gas encoded as hexadecimal.
value	Integer	The integer of value sent with this transaction encoded as hexadecimal.
data	String	The hash of the method signature and encoded parameters. For more information, see the Contract ABI description in the Solidity documentation.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

An estimation of gas for a given transaction.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"eth_estimateGas","params":
[{"from":"0x8D97689C9818892B700e27F316cc3E41e17fBeb9","to":"0xd3CdA913deB6f67967B99D67aCDFa1
712C293601","value":"0x186a0"}],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.19 eth_feeHistory

Introduction

This API returns historical gas information. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Number of blocks	String/Integer	Number of blocks in the requested range. 1 to 1024 blocks can be requested in a single query. Less than requested may be returned if not all blocks are available.
Newest block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).
Reward percentiles	Integer	(Optional) A monotonically increasing list of percentile values to sample from each block's effective priority fees per gas in ascending order, weighted by gas used.

Return Value

- oldestBlock: the lowest number block of the returned range encoded as hexadecimal.
- baseFeePerGas: an array of block base fees per gas, including an extra block value. The extra value is the next block after the newest block in the returned range. Zeroes are returned for blocks created before EIP-1559.
- gasUsedRatio: an array of block gas used ratios. These are calculated as the ratio of gasUsed and gasLimit.
- reward: an array of effective priority fees per gas data points from a single block. All zeroes are returned if the block is empty.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"id": 1, "jsonrpc": "2.0", "method": "eth_feeHistory", "params": ["0x5", "latest", [20,30]] }'
```

5.4.1.20 eth_maxPriorityFeePerGas

Introduction

This API returns a fee per gas that is an estimate of how much you can pay as a priority fee, or a tip, to get a transaction included in the current block. It consumes 109 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the priority fee needed to be included in a block.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_maxPriorityFeePerGas","id":1}'
```

5.4.1.21 eth_gasPrice

Introduction

This API returns the current gas price in wei. It consumes 101 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current gas price in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_gasPrice","params": [],"id":1}'
```

5.4.1.22 eth_getBalance

Introduction

This API returns the balance of the given account address. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Address	String	The address to check for balance.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

A hexadecimal value of the current balance in the account at the given address, in wei.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getBalance","params":
["0xc94770007dda54cF92009BFF0dE90c06F603a09f", "latest"],"id":1}'
```

5.4.1.23 eth_subscribe

Introduction

This API creates a new subscription for particular events. The node returns a subscription ID. For each event that matches the subscription, a notification with relevant data is sent together with the subscription ID. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Event type	String	The type of event to listen to.
Optional parameters	String	Optional parameters to include to describe the type of event to listen to (e.g. newHeads, newPendingTransactions, logs).

Return Value

While the subscription is active, you will receive events formatted as an object described below:

Event Object:

- jsonrpc: always **2.0**.
- method: always **eth_subscription**.
- params: an object with the following fields:
  - subscription: the subscription ID returned by the API that creates this subscription. This ID will be attached to all received events and can also be used to cancel the subscription using eth_unsubscribe.
  - result: an object whose contents vary depending on the event type.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_subscribe",
"params": ["logs"]}'
```

5.4.1.24 eth_unsubscribe

Introduction

This API cancels subscriptions with the subscription ID. It returns a boolean indicating that the subscription was canceled successfully. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Subscription ID	String	The ID of the subscription you want to unsubscribe.

Return Value

true is returned if a subscription was successfully canceled, or false is returned.

Request

```
wscat -c wss://your-http-endpoint/v1/<API-KEY> -x '{"jsonrpc":"2.0", "id": 1, "method": "eth_unsubscribe",
"params": ["0x9cef478923ff08bf67fde6c64013158d"]}'
```

5.4.1.25 eth_getStorageAt

Introduction

This API returns the value from a storage position at a given address. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address.
Storage position	String	A hexadecimal code of the position in the storage.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

It returns the value at this storage position.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getStorageAt","params":
["0x295a70b2de5e3953354a6a8344e616ed314d7251",
"0x6661e9d6d8b923d5bbaab1b96e1dd51ff6ea2a93520fdc9eb75d059238b8c5e9", "0x65a8db"],"id":1}'
```

5.4.1.26 eth_accounts

Introduction

This API returns an array of addresses owned by the client. It consumes 103 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

An array of addresses owned by the client in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_accounts","params":[],"id":1}'
```

5.4.1.27 eth_getCode

Introduction

This API returns the compiled byte code of a smart contract, if any, at a given address. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

The compiled byte code of the smart contract at the given address.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getCode","params":
["0x06012c8cf97bead5deae237070f9587f8e7a266d", "0x65a8db"],"id":1}'
```

5.4.1.28 eth_getProof

Introduction

This API returns the account and storage values, including the Merkle proof, of the specified account. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
Address	String	A 20-byte string of the storage address from which the bytecode will be obtained.
Storage keys	Array	An array of 32-byte storage keys to be proofed and included.
Block number	String	A hexadecimal block number, or the string (earliest, latest, or pending).

Return Value

- address: the address related to the account.
- accountProof: an array of RLP-serialized MerkleTree-Nodes, starting with the stateRoot-Node, following the path of the SHA3 (address) as key.
- balance: a hexadecimal value of the current balance in wei.
- codeHash: the 32-byte hash of the code of the account.
- nonce: the nonce of the account.
- storageHash: 32 bytes. The SHA3 of the StorageRoot. All storage will deliver a Merkle proof starting with this rootHash.
- storageProof: an array of storage-entries as requested. Each entry is an object with these properties:
  - key: the requested storage key.
  - value: the storage value.
  - proof: an array of RLP-serialized MerkleTree-Nodes, starting with the storageHash-Node, following the path of the SHA3 (key) as path.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc": "2.0","method": "eth_getProof","id": 1,"params":
["0x7F0d15C7FAae65896648C8273B6d7E43f58Fa842",
["0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421"], "latest"]}'
```

5.4.1.29 eth_getLogs

Introduction

This API returns an array of all the logs matching the given filter object. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

An array of log objects, or an empty array if nothing has changed since last poll. Log objects contain the following keys and their values:

- removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
- logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
- transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.
- transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
- blockHash: 32 bytes. The hash of the block where this log was in. It is null for a pending log.
- blockNumber: the block number where this log was in. It is null for a pending log.
- address: 20 bytes. The address from which this log originated.
- data: It contains one or more 32-byte non-indexed arguments of the log.
- topics: an array of 0 to 4 indexed log arguments, each 32 bytes. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"jsonrpc":"2.0","method":"eth_getLogs","params":[{"blockHash":
"0x7c5a35e9cb3e8ae0e221ab470abae9d446c3a5626ce6689fc777dcffcab52c70", "topics":
["0x241ea03ca20251805084d27d4440371c34a0b85ff108f6bb5611248f73818b80"]}],"id":74}'
```

5.4.1.30 eth_getFilterChanges

Introduction

The polling method for a filter, which returns an array of logs which occurred since last poll. Call eth_newFilter, eth_newBlockFilter, or eth_newPendingTransactionFilter to create a filter. It consumes 108 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

- log object array: an array of log objects, or an empty array if nothing has changed since last poll.

- For filters created with eth_newBlockFilter, the return values are block hashes (32 bytes), for example, ["0x3454645634534..."].
- For filters created with eth_newFilter, the logs are objects with the following parameters:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
  - removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - transactionHash: 32 bytes. The hash of the transactions from which this log was created. It is null for a pending log.
  - transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":73}'
```

5.4.1.31 eth_getFilterLogs

Introduction

This API returns an array of all the logs matching the given filter ID. It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.

Parameter	Type	Description
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.
blockhash	String	(Optional) It restricts the logs returned to the single block referenced in the 32-byte hash blockHash. Using blockHash is equivalent to setting fromBlock and toBlock to the block number referenced in the blockHash. If blockHash is present in in the filter criteria, then neither fromBlock nor toBlock are allowed.

Return Value

- Log object array: an array of log objects that match the filter. For an array of logs that occurred since the last poll, use eth_getFilterChanges. Log objects contain the following keys and their values:
  - address: the address from which this log originated.
  - blockHash: the hash of the block where this log was in. It is null for a pending log.
  - blockNumber: the number of the block where this log was in. It is null for a pending log.
  - data: the non-indexed arguments of the log.
  - logIndex: the hexadecimal of the log index position in the block. It is null for a pending log.
  - removed: true when the log was removed due to a chain reorganization. false if it is a valid log.
  - topics: an array of zero to four 32-byte data of the index log arguments. In Solidity, the first topic is the hash of the signature of the event (e.g. Deposit(address,bytes32,uint256)), except you declare the event with the anonymous specifier.
  - transactionHash: the hash of the transaction from which this log was created. It is null for a pending log.
  - transactionIndex: the hexadecimal of the transactions index position from which the log created. It is null for a pending log.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterLogs","params":["0x16"],"id":74}'
```

5.4.1.32 eth_newBlockFilter

Introduction

This API creates a filter in the node to notify when a new block arrives. It consumes 144 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newBlockFilter","params":[],"id":73}'
```

5.4.1.33 eth_newFilter

Introduction

This API creates a filter object based on the given filter options to notify when the state changes (logs). It consumes 120 CUs.

Parameter Description

Parameter	Type	Description
address	String	(Optional) A 20-byte contract address or a list of addresses from which logs should originate.
fromBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
toBlock	String	(Optional) A hexadecimal block number, or the string (earliest, latest, or pending). Latest is set by default.
topics	String	(Optional) An array of 32-byte data topics. Topics are order-dependent.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newFilter","params":[{"topics":
["0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef"]}],"id":73}'
```

5.4.1.34 eth_newPendingTransactionFilter

Introduction

This API creates a filter in the node to notify when new pending transactions arrive at BSC. It consumes 252 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

It returns the ID of the new filter in hexadecimal.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_newPendingTransactionFilter","params":[],"id":73}'
```

5.4.1.35 eth_uninstallFilter

Introduction

This API uninstalls a filter with the given filter ID. It should always be called when watching is no longer needed. Additionally, filters time out when they are not requested with eth_getFilterChanges for a period of time. It consumes 127 CUs.

Parameter Description

Parameter	Type	Description
Filter ID	String	The string of the filter ID.

Return Value

true is returned if a filter was successfully uninstalled, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_uninstallFilter","params":["0xb"],"id":73}'
```

5.4.1.36 eth_chainId

Introduction

This API returns the currently configured chain ID. It consumes 103 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

A hexadecimal value of the current chain ID.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_chainId","params": [],"id":1}'
```

5.4.1.37 web3_sha3

Introduction

This API returns Keccak-256 (not the standardized SHA3-256) hash of the given data. It consumes 131 CUs.

Parameter Description

Parameter	Type	Description
Data	String	Data to be converted.

Return Value

The SHA3 hash of the given string.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"web3_sha3","params": ["0x68656c6c6f20776f726c64"],"id":64}'
```

5.4.1.38 web3_clientVersion

Introduction

This API returns the current version of the chain client. It consumes 137 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

The current client version.

5.4.1.39 txpool_status

Introduction

This API returns the number of transactions in pending and queued states. It consumes 120 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

Object: A transaction object with the following fields:

- pending: the total number of pending transactions in the transaction pool (txpool), represented in hexadecimal format.
- queued: the total number of queued transactions in the txpool, represented in hexadecimal format.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"txpool_status","params":[],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.40 net_listening

Introduction

This API returns true if the client is actively listening for network connections. It consumes 130 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

true is returned if the client is actively listening for network connections, or false is returned.

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"net_listening","params":[],"id":1,"jsonrpc":"2.0"}'
```

5.4.1.41 net_version

Introduction

This API returns the current network ID. It consumes 107 CUs.

Parameter Description

This method does not accept any parameters.

Return Value

Object: The string value of current network ID. Typical values are as follows:

- 1 - ethereum mainnet
- 2 - morden testnet (deprecated)
- 3 - ropsten testnet
- 4 - rinkeby testnet
- 5 - goerli testnet
- 11155111 - sepolia testnet
- 10 - optimism mainnet
- 69 - optimism kovan testnet
- 42 - kovan testnet
- 137 - matic/polygon mainnet
- 80001 - matic/polygon mumbai testnet
- 250 - fantom mainnet
- 100 - xdai mainnet
- 56 - bsc mainnet

Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
  -X POST \
  -H "Content-Type: application/json" \
  --data '{"method":"net_version","params":[],"id":1,"jsonrpc":"2.0"}'
```

6 Batch Requests

6.1 Introduction

A batch request is a single HTTP request that contains multiple API calls nested within it. A client can send several request objects together at the same time, filled within an array, and will get a corresponding array of response objects from the server.

The server processes all requests of this batch RPC calls concurrently within the CU limit. RPC calls that exceed the CU limit continue to be processed after the previous calls are processed.

Batch requests contain different RPC calls can be complicated. They can be less reliable compared to individual API calls. Therefore, batch requests are not recommended.

6.2 Scope

All HTTP JSON-RPC APIs opened by public blockchains of NES can be called in batches. WebSocket APIs are not supported currently.

6.3 Examples

The eth_getFilterChange API is used as an example. Different from a single request, a batch request's request body encapsulates multiple sub-requests in arrays. The server returns the response of each sub-request in an array.

Example of a Single Request

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":1}'
```

Example of Batch Requests

```
curl https://your-http-endpoint/v1/<API-KEY> \
    -X POST \
    -H "Content-Type: application/json" \
    -d '[{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":1},
{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":2},
{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":3},
{"jsonrpc":"2.0","method":"eth_getFilterChanges","params":["0x16"],"id":4}]'
```

Return Value

Response arrays corresponding to each RPC call.

```
[{"jsonrpc":"2.0","id":1,"result":null},{"jsonrpc":"2.0","id":2,"result":null},{"jsonrpc":"2.0","id":3,"result":null},
{"jsonrpc":"2.0","id":4,"result":null}]
```

7 Enhanced APIs

7.1 Introduction

NES provides enhanced APIs to optimize your gas fees.

7.2 Enhanced APIs

7.2.1 Optimizing Gas Fees

These APIs save your gas fees without affecting the transaction success rate and confirmation rate.

NOTE

These APIs are supported only in the dedicated edition.

Procedure

You can send a group of raw transactions with different gas fees to the node and call APIs. These APIs will:

Step 1 Sort transactions in an ascending order based on their gas fees.

Step 2 Submit the transaction with the lowest gas fees.

Step 3 Check whether the transaction has been confirmed by the block.

Step 4 Submit another transaction with higher gas fees if the previous one is not confirmed in four seconds. This operation repeats until the submitted transaction is confirmed.

----End

Precautions

1. These APIs can save gas fees for EIP-1559 and typical transactions.

- EIP-1559 transactions: the parameters of a group of raw transactions should be the same except **maxPriorityFeePerGas**.
- Typical transactions: the parameters of a group of raw transactions should be the same except **gasPrice**.

2. EIP-1559 and typical transactions cannot be in a same group.
3. These APIs can be used by blockchains that support the **eth_sendRawTransactionAPI** API, including Ethereum, Arbitrum, Polygon, and BSC.

7.2.1.1 nes_sendGasOptimizedTransaction

This API receives a string array containing raw transactions, and returns an ID. This ID can be used to query the gas optimization status. The array length is 1 to 10.

Request

```
curl https://your-http-endpoint/your-credential \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"nes_sendGasOptimizedTransation","params":[
"0x02f87583064aba048405f5e10085012a05f20082520894958a15271aa13f6b7feb029b6114a69e6de8b69387
2386f26fc1000080c001a0df9b7a8ab18f081930b6c6e85fd75fdbaa1de8e0027f21bff1aeeb9d6ff6e477a062bec
3ad68abc739c3bc96305a97d7952627cd95190f86e4f0cb3ffbcc002623",
"0x02f87483064aba048405f5e10084ee6b280082520894958a15271aa13f6b7feb029b6114a69e6de8b693872
386f26fc1000080c001a024b016079f1aa5b437f5f8c7aee25a3accb0873eab7ac86f39af10068ee725efa05896d1
f7562f21f185b23d4d6c0cb735518d3e5c9f35d5fa5f900b6b9b0f80d9",
"0x02f87483064aba048405f5e10084b2d05e0082520894958a15271aa13f6b7feb029b6114a69e6de8b693872
386f26fc1000080c080a00f3767650d09f1330953abd4498fd8bca3ca0c1444cf56ca67e4ebda003d5ba4a042d96
1648b42b33846c4e1812b5727c1fba4f9cad8602b708251c912aa3f647e"
],"id":1}'
```

Return Value

An ID that can be an input for **nes_getGasOptimizedTransactionStatus** to query the gas optimization status.

```
{"jsonrpc":"2.0","id":1,"result":"0x4201"}
```

7.2.1.2 nes_getGasOptimizedTransactionStatus

This API returns the gas optimization status by an ID.

Request

```
curl https://your-http-endpoint/your-credential \
    -X POST \
    -H "Content-Type: application/json" \
    -d '{"jsonrpc":"2.0","method":"nes_getGasOptimizedTransactionStatus","params":["0x4201"],"id":1}'
```

Return Value

The return value can be **pending**, **success**, or **failed**.

1. If the API is called, **pending** is returned.
2. If the transaction is not confirmed, **pending** is returned.
3. If the transaction is confirmed, **success** is returned.

4. If an error occurs, **failed** is returned.

```
{"jsonrpc":"2.0","id":1,"result":"pending"}
```