星云链Nebulas——2.发送交易

Nebulas提供了三种方式去发送我们的交易：

1.签名 & 发送
2.密码 & 发送
3.解锁 & 发送

下面我们分别介绍如何通过以上三种方式在Nebulas中发送一笔交易，并验证交易是否成功。

1、准备账户

在星云链上，每个地址表示一个唯一的账户，一一对应。

在发送交易前，我们需要准备两个账户：一个账户用来发送代币 (称为"from") 和另一个账户来接收代币 (称为"to")。

- 发送者账户

  在这里，我们将会使用配置文件`conf/default/genesis.conf`中预分配过代币的账户中选择一个作为发送者账户，默认选择21个初始矿工中的第1个矿工地址，也就是`n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE`。

genesis.conf

```
# Neb genesis text file. Scheme is defined in core/pb/genesis.proto.
#

meta {
  # 每条链的唯一标识
  # 每个区块和交易只会属于一条唯一的链，保证安全性
  chain_id: 100
}

consensus {
  # 在贡献度证明(PoD)被充分验证前，星云链采用DPoS共识算法
  # DPoS共识中，21个人组成一个朝代
  # 每隔一段时间都会切换朝代，每个朝代内，21个矿工轮流出块
  # 由于DPoS只是过渡方案，所以暂时不开放给公众挖矿，即当前版本朝代不会发生变更
  dpos {
    # 初始朝代，包含21个初始矿工地址
    dynasty: [
      "n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE",
      "n1GmkKH6nBMw4rrjt16RrJ9WcgvKUtAZP1s",
      "n1H4MYms9F55ehcvygwWE71J8tJC4CRr2so",
      "n1JAy4X6KKLCNiTd7MWMRsVBjgdVq5WCCpf",
      "n1LkDi2gGMqPrjYcczUiweyP4RxTB6Go1qS",
      "n1LmP9K8pFF33fgdgHZonFEMsqZinJ4EUqk",
      "n1MNXBKm6uJ5d76nJTdRvkPNVq85n6CnXAi",
      "n1NrMKTYESZRCwPFDLFKiKREzZKaN1nhQvz",
      "n1NwoSCDFwFL2981k6j9DPooigW33hjAgTa",
      "n1PfACnkcfJoNm1Pbuz55pQCwueW1BYs83m",
      "n1Q8mxXp4PtHaXtebhY12BnHEwu4mryEkXH",
      "n1RYagU8n3JSuV4R7q4Qs5gQJ3pEmrZd6cJ",
      "n1SAQy3ix1pZj8MPzNeVqpAmu1nCVqb5w8c",
      "n1SHufJdxt2vRWGKAxwPETYfEq3MCQXnEXE",
      "n1SSda41zGr9FKF5DJNE2ryY1ToNrndMauN",
      "n1TmQtaCn3PNpk4f4ycwrBxCZFSVKvwBtzc",
      "n1UM7z6MqnGyKEPvUpwrfxZpM1eB7UpzmLJ",
      "n1UnCsJZjQiKyQiPBr7qG27exqCLuWUf1d7",
      "n1XkoVVjswb5Gek3rRufqjKNpwrDdsnQ7Hq",
      "n1cYKNHTeVW9v1NQRWuhZZn9ETbqAYozckh",
      "n1dYu2BXgV3xgUh8LhZu8QDDNr15tz4hVDv"
    ]
  }
}
```

- 接收者账户

  我们使用如下命令创建一个全新的账户来做接收者，请记住输入该账号的密码。

  ```
  ./neb account new
  ```

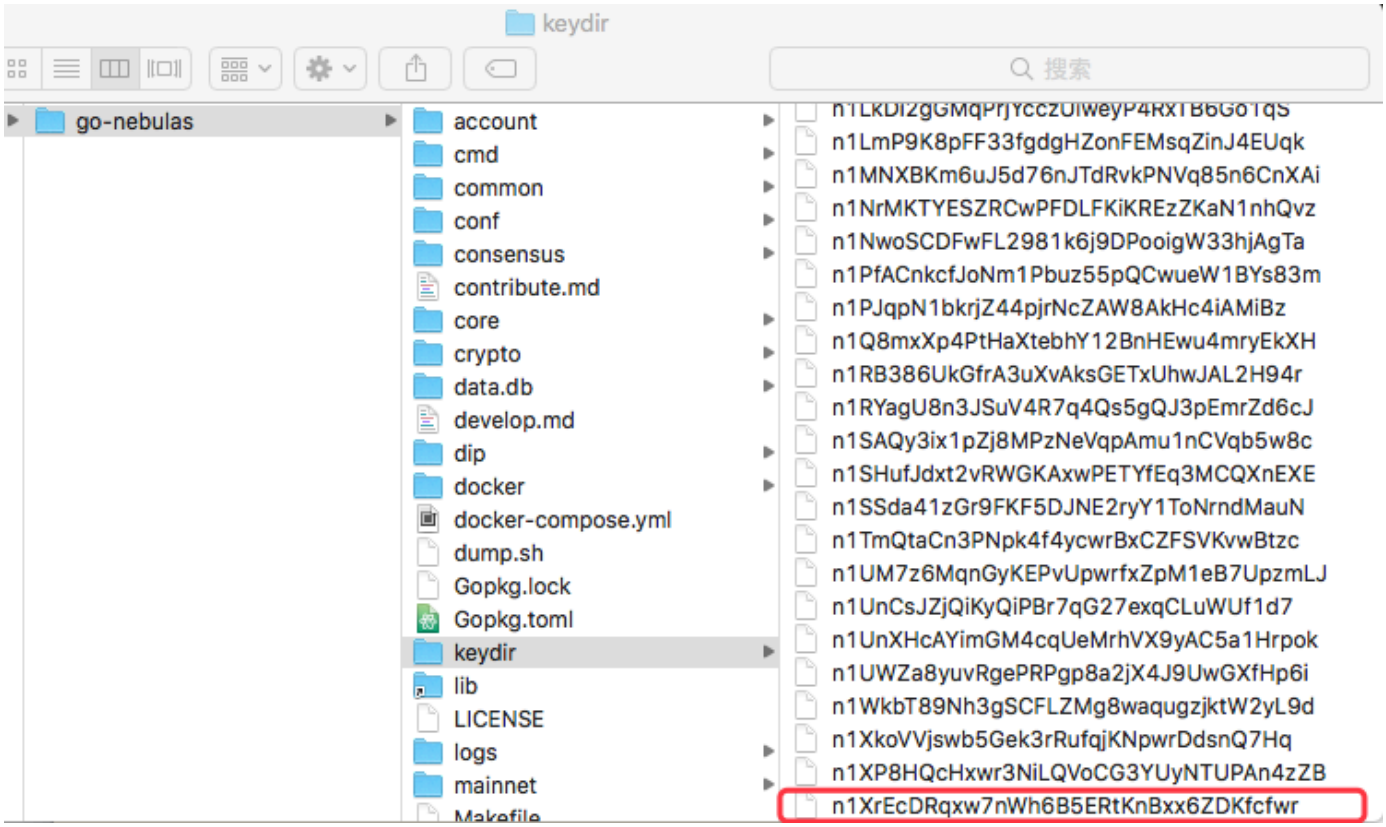
  终端操作如下：

  ```
  wenzildeiMac:~ wenzil$ cd $GOPATH/src/github.com/nebulasio/go-nebulas
  wenzildeiMac:go-nebulas wenzil$ ./neb account new
  Your new account is locked with a passphrase. Please give a passphrase.
  Do not forget this passphrase.
  ```

```
Passphrase:
Repeat passphrase:
Address: n1XrEcDRqxw7nWh6B5ERtKnBxx6ZDKfcfwr
```

> 提示：你创建的新账户和上面可能不一样，请以你创建的账户做为接收者继续接下来的实验。

新账将会被放置在$GOPATH/src/github.com/nebulasio/go-nebulas/keydir/内。



2、启动私有链

我们将在本地搭建一个私有链来作为本教程的测试环境。

2.1 启动种子节点

首先，我们启动本地私有链的第一个节点，它可以作为其他节点的种子节点。

```
./neb -c conf/default/config.conf
```

2.2 启动矿工节点

接着，我们启动一个矿工节点接入本地私有链，这个节点之后将会产生新的区块。

```
./neb -c conf/example/miner.conf
```

> 多久会生成一个新的区块?
>
> 在星云链上, 在贡献度证明（Proof-of-Devotion， 技术白皮书中有详细描述）被充分验证前，DPoS被选择作为一个过渡方案。在我们采用DPoS共识算法中，总共有21个矿工，每个矿工会轮流每15s出一个新区块。
>
> 在我们目前的测试环境中，由于我们只启动了21个矿工中的一个，所以需要等待15*21s才会出一个新区块。

一旦一个新区块被挖出，挖块奖励将会被自动发送到当前矿工配置的Coinbase账户中，在`conf/example/miner.conf`里，该账户就是`n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE`。

3、星云链交互

星云链提供给开发者HTTP API, RPC API和CLI来和运行中的星云节点交互。在教程中，我们将会基于HTTP API（API Module | Admin Module）来介绍三种发送交易的方法。

> 提示：星云链的HTTP服务默认端口号为8685。

首先，在发送新交易前，我们检查下发送者账户的状态。

3.1 检查账户状态

每个交易如果需要上链，都需要给矿工缴纳一部分手续费，所以发送者账户中需要有一部分钱才能成功发送交易。一般一个普通转账交易，手续费在0.000000002NAS左右，非常少。

我们可以通过API Module中的`/v1/user/accountstate`接口来获取发送者账户`n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE`的账户信息，检查下是否有足够的钱支付上链手续费。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X POST h
ttp://localhost:8685/v1/user/accountstate -d '{"address":"n1FF1nz6tarkDVwWQk
MnnwFPuPKUaQTdptE"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:37:06 GMT
Content-Length: 72

{"result":{"balance":"50000000000000000000000000","nonce":"0","type":87}}
```

> 提示：

> `Type`用于标记账户类型。88表示该账户为智能合约账户，部署一个合约之后，就可以得到一个合约账户。87表示非合约账户，我们通过`./neb account new`创建的账户就是非合约账户，用户存储链上资产。
>
> `Nonce`用于标记账户发起的所有交易的顺序。同一个账户，每发起一个新的交易，`Nonce`就加一，初始为0，第一个交易的`Nonce`为1。

如我们所见，发送者账户在预分配后拥有500000000000000000000000(5 * 10^{24})个代币，1个NAS是1000000000000000000（10^{18}）个代币，用于支付交易上链的手续费绰绰有余。

然后我们检查接收者账户的状态。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X POST h
ttp://localhost:8685/v1/user/accountstate -d '{"address":"n1XrEcDRqxw7nWh6B5
ERtKnBxx6ZDKfcfwr"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:37:12 GMT
Content-Length: 48

{"result":{"balance":"0","nonce":"0","type":87}}
```

如我们期望的那样，新账户没有任何代币。

3.2 发送交易

接下来，我们将介绍星云链上三种发送交易的方式。

3.2.1 签名 & 发送

使用这种方式，我们可以在离线环境下先使用私钥签名好交易，然后把签好名的交易在联网的机器上发出。这是最安全的发送交易的方式，私钥可以完全离线保存，不联网。Web-Wallet正是基于Neb.js采用这种方法发送的交易。

首先，我们使用Admin Module中的`v1/admin/sign`接口给准备发的交易签名，得到交易的二进制数据。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/admin/sign -d '{"transaction":{"from":"n1FF
1nz6tarkDVwWQkMnnwFPuPKUaQTdptE","to":"n1XrEcDRqxw7nWh6B5ERtKnBxx6ZDKfcfwr",
 "value":"1000000000000000000","nonce":1,"gasPrice":"1000000","gasLimit":"20
00000"}, "passphrase":"passphrase"}'
```

```
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:37:25 GMT
Content-Length: 334

{"result":{"data":"CiAs+5R18Z6Xuohy8woPAx66ObFvUzTssT0n80inTCahRxIaGVcH+WT/S
VMkY18ix7SG4F1+Z8evXJoA35caGhlXviSpngoaH22bJcW6rhVG2t/5okgDErrdIhAAAAAAAAAAA
A3gtrOnZAAAKAEwpILe2AU6CAoGYmluYXJ5QGRKEAAAAAAAAAAAAAAAAAAAPQkBSEAAAAAAAAAAA
AAAAAAehIBYAWJBob/XDfybbJ19ZCp8AqLuDla/FKC08nZL/qClq7BKjZs0o7PXYJ1Q8UVdhRKdr
zlng7KjOG4sRkNOMp1R2OKsvQA="}}
```

> 提示：
> 你创建的交易返回的内容跟这里可能不一样，以你返回的内容继续接下来的实验。
>
> 在发送交易时，对于同一个账户，只有当他`Nonce`为N的交易上链后，`Nonce`为N+1的交易才能上链，有严格的顺序，`Nonce`必须严格加1。可以通过GetAccountState接口查看最新的Nonce。

然后，我们将签好名的交易原始数据提交到本地私有链里的星云节点，data字段的值替换为上面返回的内容。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/user/rawtransaction -d '{"data":"CiAs+5R18Z
6Xuohy8woPAx66ObFvUzTssT0n80inTCahRxIaGVcH+WT/SVMkY18ix7SG4F1+Z8evXJoA35caGh
lXviSpngoaH22bJcW6rhVG2t/5okgDErrdIhAAAAAAAAAAA3gtrOnZAAAKAEwpILe2AU6CAoGYm
luYXJ5QGRKEAAAAAAAAAAAAAAAAAAAPQkBSEAAAAAAAAAAAAAAAAAAAehIBYAWJBob/XDfybbJ19ZC
p8AqLuDla/FKC08nZL/qClq7BKjZs0o7PXYJ1Q8UVdhRKdrzlng7KjOG4sRkNOMp1R2OKsvQA="}
'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:41:30 GMT
Content-Length: 110

{"result":{"txhash":"2cfb9475f19e97ba8872f30a0f031eba39b16f5334ecb13d27f348a
74c26a147","contract_address":""}}
```

3.2.2 密码 & 发送

如果你信任一个星云节点帮你保存keystore文件，你可以使用第二种方法发送交易。

首先，上传你的keystore文件到你信任的星云节点的keydir文件夹下。如果在节点在本地，可以使用如下指令。

> 提示：意思是说把某个账号对应的私钥文件 `keystore.json` 复制到克隆下载的星云链 `go-nebulas` 的 `keydir` 目录下，本文有对 `keydir` 目录截图。

```
cp /path/to/keystore.json /path/to/keydir/
```

然后，我们发送交易的同时，带上我们keystore的密码，在被信任的节点使用SendTransactionWithPassphrase接口上一次性完成签名和发送过程。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/admin/transactionWithPassphrase -d '{"trans
action":{"from":"n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE","to":"n1XrEcDRqxw7nWh6
B5ERtKnBxx6ZDKfcfwr", "value":"1000000000000000000","nonce":2,"gasPrice":"10
00000","gasLimit":"2000000"},"passphrase":"passphrase"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:43:04 GMT
Content-Length: 110

{"result":{"txhash":"6795d88382586ac0d5ea55039f078eb1f133388a25932f6bcebfc9e
2993d7781","contract_address":""}}
```

> 提示：因为我们在之前使用 `n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE` 发送了一个 `Nonce` 为1的交易，所以这里新的交易的 `Nonce` 应该增加1，变成2再提交。

3.2.3 解锁 & 发送

这是最危险的发送交易的方法。除非你完全信任一个星云节点，否则不要使用这种方法来发送交易。

首先，上传你的keystore文件到你信任的星云节点的keydir文件夹下。如果在节点在本地，可以使用如下指令。

> 提示：意思是说把某个账号对应的私钥文件 `keystore.json` 复制到克隆下载的星云链 `go-nebulas` 的 `keydir` 目录下，本文有对 `keydir` 目录截图。

```
cp /path/to/keystore.json /path/to/keydir/
```

然后，使用你的keystore文件的密码，在指定的时间范围来在被信任的节点上使用Unlock接口解锁账户。时间单位为纳秒，300000000000为300s。

```
wenzildeiMac:go-nebulas wenzil$ wenzil$ curl -i -H 'Content-Type: applicatio
n/json' -X POST http://localhost:8685/v1/admin/account/unlock -d '{"address"
:"n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE","passphrase":"passphrase","duration":
"300000000000"}'
-bash: wenzil$: command not found
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/admin/account/unlock -d '{"address":"n1FF1n
z6tarkDVwWQkMnnwFPuPKUaQTdptE","passphrase":"passphrase","duration":"3000000
00000"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:43:55 GMT
Content-Length: 26

{"result":{"result":true}}
```

一旦一个账户在节点上被解锁，任何可以访问该机器SendTransaction接口的人，都可以直接使用该账户的身份发送交易。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: application/json'
-X POST http://localhost:8685/v1/admin/transaction -d '{"from":"n1FF1nz6tark
DVwWQkMnnwFPuPKUaQTdptE","to":"n1XrEcDRqxw7nWh6B5ERtKnBxx6ZDKfcfwr", "value"
:"1000000000000000000","nonce":3,"gasPrice":"1000000","gasLimit":"2000000"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:44:28 GMT
Content-Length: 110

{"result":{"txhash":"af755d4076bb721a2c51caf270a198eee1cbef7d220563da25f64de
2a54d698a","contract_address":""}}
```

4、交易数据

不论使用的哪一种方法发送交易，我们都会得到两个返回值，`txhash`和`contract_address`。其中`txhash`为交易hash，是一个交易的唯一标识。如果当前交易是一个部署合约的交易，`contract_address`将会是合约地址，调用合约时都会使用这个地址，是合约的唯一标识。我们将在后续推出的《星云链Nebulas——3.编译和部署智能合约》一文中介绍如何发送部署智能合约的交易。

使用`txhash`我们可以查看交易数据，知道当前交易的状态。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X POST h
ttp://localhost:8685/v1/user/getTransactionReceipt -d '{"hash":"af755d4076bb
721a2c51caf270a198eee1cbef7d220563da25f64de2a54d698a"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Wed, 06 Jun 2018 10:45:21 GMT
Content-Length: 413

###交易待定的返回内容，交易还没被打包，交易还未上链###
{"result":{"hash":"af755d4076bb721a2c51caf270a198eee1cbef7d220563da25f64de2a
54d698a","chainId":100,"from":"n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE","to":"n1
XrEcDRqxw7nWh6B5ERtKnBxx6ZDKfcfwr","value":"1000000000000000000","nonce":"3"
,"timestamp":"1528267468","type":"binary","data":null,"gas_price":"1000000",
"gas_limit":"2000000","contract_address":"","status":2,"gas_used":"","execut
e_error":"","execute_result":""}}

###交易成功的返回内容，交易已经被打包，交易已上链###
{"result":{"hash":"af755d4076bb721a2c51caf270a198eee1cbef7d220563da25f64de2a
54d698a","chainId":100,"from":"n1FF1nz6tarkDVwWQkMnnwFPuPKUaQTdptE","to":"n1
XrEcDRqxw7nWh6B5ERtKnBxx6ZDKfcfwr","value":"1000000000000000000","nonce":"3"
,"timestamp":"1528267468","type":"binary","data":null,"gas_price":"1000000",
"gas_limit":"2000000","contract_address":"","status":1,"gas_used":"20000"}}

###实验时，交易成功后过段时间再查看交易数据时返回的内容###
{"error":"transaction not found"}
```

这里的`status`可能有三种状态值，0，1和2。

- **0: 交易失败.** 表示当前交易已经上链，但是执行失败了。可能是因为部署合约或者调用合约参数错误。
- **1: 交易成功.** 表示当前交易已经上链，而且执行成功了。
- **2: 交易待定.** 表示当前交易还没有上链。可能是因为当前交易还没有被打包；如果长时间处于当前状态，可能是因为当前交易的发送者账户的余额不够支付上链手续费。

4.1 复查接收者账户余额

我们复查一下接收者账户上的钱是否已经到账了。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X POST h
ttp://localhost:8685/v1/user/accountstate -d '{"address":"n1XrEcDRqxw7nWh6B5
ERtKnBxx6ZDKfcfwr"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
```

```
Date: Wed, 06 Jun 2018 10:47:01 GMT
Content-Length: 66

{"result":{"balance":"3000000000000000000","nonce":"0","type":87}}
```

我们用三种方式分别发送了一笔转账，每笔转一个NAS，所以这里看到接收者账户中已经有了3个NAS，即3000000000000000000个代币。

本文参考：星云链Nebulas官方Github

下一篇预告

星云链Nebulas——3.编译和部署智能合约