星云链Nebulas——5.通过RPC API和星云链交互

星云链节点启动后可以通过RPC远程控制访问。星云链提供了一系列API来获取节点的信息，账号余额，发送交易和部署调用智能合约。

星云链的远程访问是GRPC实现的，通过代理(GRPC Gateway)也可以通过HTTP访问。HTTP访问是RESTful实现的接口，参数与GRPC的调用接口参数相同。

API

我们已经在每个星云节点中实现了RPC服务器和HTTP服务器，提供给用户丰富的接口来与星云节点交互。

1、接口模块

现在，星云节点的所有的接口被分为两个模块：API和Admin。

- API：提供所有和用户私钥无关的接口
- Admin：提供所有和用户私钥相关的接口

建议星云节点对外提供服务时，可以把API接口开放给公众，而将Admin接口开放给授权用户。（注：星云官方的testnet及mainnet的admin相关api不支持远程访问）

2、配置文件

星云节点中的RPC服务器和HTTP服务器都可以在节点的配置中配置对应的端口，以及开放的模块。

```
# 用户与节点交互的服务配置，同一台机器启动多个时注意修改端口防止占用
rpc {
    # gRPC API服务端口
    rpc_listen: ["127.0.0.1:8684"]
    # HTTP API服务端口
    http_listen: ["127.0.0.1:8685"]
    # 开放可对外提供http服务的模块
    http_module: ["api","admin"]
}
```

3、使用实例

3.1 HTTP

通过**HTTP**接口和星云节点交互。

3.1.1 GetNebState（获取当前节点状态）

我们可以调用API模块中的`GetNebState`接口来获取节点当前状态，包括所在链ID，最新区块，协议版本等等。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H Accept:application/json -X GET ht
tp://localhost:8685/v1/user/nebstate
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Fri, 08 Jun 2018 11:42:25 GMT
Content-Length: 259

{"result":{"chain_id":100,"tail":"4c987dbdd2aae2cd8b0c6529e99740e12f7ac10048
c9ef46e40106e79d261742","lib":"0000000000000000000000000000000000000000000000
0000000000000000","height":"28","protocol_version":"/neb/1.0.0","synchron
ized":false,"version":"1.0.1"}}
```

3.1.2 UnlockAccount（解锁账户）

我们可以调用Admin模块中的`UnlockAccount`接口来在节点内存中解锁一个账户。所有解锁的账户都可以被用来直接发送交易，而不需要密码。

```
wenzildeiMac:go-nebulas wenzil$ curl -i -H 'Content-Type: a POST http://loca
lhost:8685/v1/admin/account/unlock -d '{"address":"n1NrMKTYESZRCwPFDLFKiKREz
ZKaN1nhQvz", "passphrase": "passphrase"}'
HTTP/1.1 200 OK
Content-Type: application/json
Vary: Origin
Date: Fri, 08 Jun 2018 11:45:08 GMT
Content-Length: 26

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

3.2 RPC

RPC服务器基于GRPC实现. GRPC的基于Protocol Buffers来做序列化，你可以在星云RPC Protobuf文件夹下找到所有的RPC相关的Proto文件定义。

这里有一些使用golang调用RPC接口的实例。

3.2.1 GetNebState

我们可以使用API模块中的`GetNebState`接口来获取节点当前状态。

```
import(
    "github.com/nebulasio/go-nebulas/rpc"
    "github.com/nebulasio/go-nebulas/rpc/pb"
)

// GRPC server connection address configuration
addr := fmt.Sprintf("127.0.0.1:%d",uint32(8684))
conn, err := grpc.Dial(addr, grpc.WithInsecure())
if err != nil {
    log.Warn("rpc.Dial() failed:", err)
}
defer conn.Close()

// API interface to access node status information
api := rpcpb.NewAPIServiceClient(conn)
resp, err := ac.GetNebState(context.Background(), & rpcpb.GetNebStateRequest
 {})
if err != nil {
    log.Println("GetNebState", "failed", err)
} else {
    log.Println("GetNebState tail", resp)
}
```

3.2.2 LockAccount

我们已经在之前使用HTTP接口把账户`n1NrMKTYESZRCwPFDLFKiKREzZKaN1nhQvz`解锁了。 我们可以调用Admin模块中的`LockAccount`再次锁定它。

```
import(
    "github.com/nebulasio/go-nebulas/rpc"
    "github.com/nebulasio/go-nebulas/rpc/pb"
)

// GRPC server connection address configuration
addr := fmt.Sprintf("127.0.0.1:%d",uint32(8684))
conn, err := grpc.Dial(addr, grpc.WithInsecure())
if err != nil {
    log.Warn("rpc.Dial() failed:", err)
}
defer conn.Close()

// Admin interface to access, lock account address
```

```
admin := rpcpb.NewAdminServiceClient(conn)
from := "n1NrMKTYESZRCwPFDLFKiKREzZKaN1nhQvz"
resp, err = management.LockAccount(context.Background(), & rpcpb.LockAccount
Request {Address: from})
if err != nil {
    log.Println("LockAccount", from, "failed", err)
} else {
    log.Println("LockAccount", from, "result", resp)
}
```

接口列表

更多的接口列表请参考官方文档。

- API Module
- Admin Module.

完成

恭喜你，成功走完了整个教程! 欢迎阅读下列指南来加入官方的测试网和主网，探索更加广阔的区块链世界。

Join to Testnet
Join to Mainnet

后续，打算再推出一篇《星云链Nebulas——6.开发和部署星云链DApp》

本文参考：星云链Nebulas官方Github