**1.7.truechain\_ide\_remote\_breakpoint\_debug**

This tutorials provides an introduction how to debug the go program on remote host use ide or dlv command line tool

**System requirements**

* dlv

Delve is a debugger for the Go programming language. The goal of the project is to provide a simple, full featured debugging tool for Go

**Step 1**

**Note: this article operates under the OSX system**

install the requirements

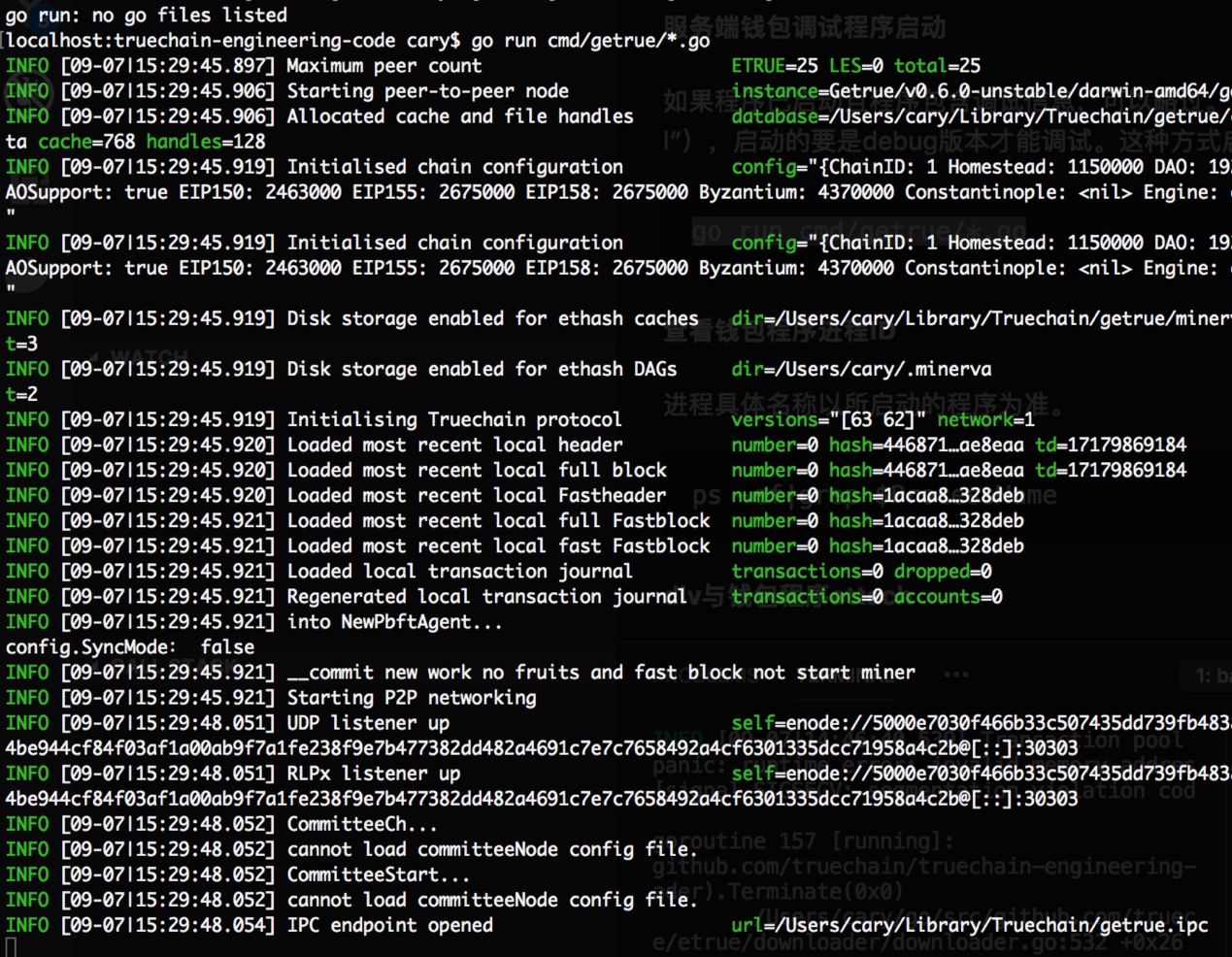
go get -u github.com/derekparker/delve/cmd/dlv

**Step 2**

Launch the wallet debug program on the server

**Note:If the program is started and the program contains debugging information, you can skip it. Note the compile options (go build -gcflags "-N -l"), if you start the debug version. This way of starting is not recommended to follow.**

go run cmd/getrue/\*.go



**Step 3**

use dlv commandline tool attach the wallet program and set stdout to network endpoint

dlv attach $(ps -ef|grep $ProcessName)--headless --api-version=2 --log --listen=:8181

Note: parameter \*\*--listen=:8181 \*\*is used to set port on the server e.g.: 8181

**debug with dlv**

* b ${function} : Set a breakpoint on the main function
* c : Continue execution after the breakpoint
* b : ${go*file:line*number} Set breakpoints by line number
* bp : List the breakpoint
* clear ${breakpoint} : Del the breakpoint
* bt : Print the stack infomation
* p : print the variable
* frame ${n} ${cmd} : Execute the corresponding instruction on the nth layer call stack
* goroutines : Print goroutines's information

**debug with vscode**

configure the launch.json file

{

"name": "Attach",

"type": "go",

"request": "launch",

"mode": "remote",

"remotePath": "${workspaceRoot}",

"port": 8181,

"host": "192.168.16.126",

"program": "${workspaceRoot}",

"env": {},

"args":["--nodiscover", "--mine", "--etherbase", "8a45d70f096d3581866ed27a5017a4eeec0db2a1"],

}

check the arguments host and port was set to the remote host endpoint address.

then start the vscode debugger, do what you want to do.

