區塊鏈中的節點

Bill Hsu @ SCU

Bill Hsu 徐子修

- 經歷

  資工系 @ 台灣大學

  技術長 @ 台大區塊鏈研究社

  核心成員 @ 台灣密碼龐克

  區塊鏈工程師 @ 密碼貨幣交易所

  資訊奧林匹亞保送生

  .....

CONTENT

目錄 >>

01 ／

什麼是節點

02 ／

節點類型

03 ／

節點同步

04 ／

來自前線的報導

01

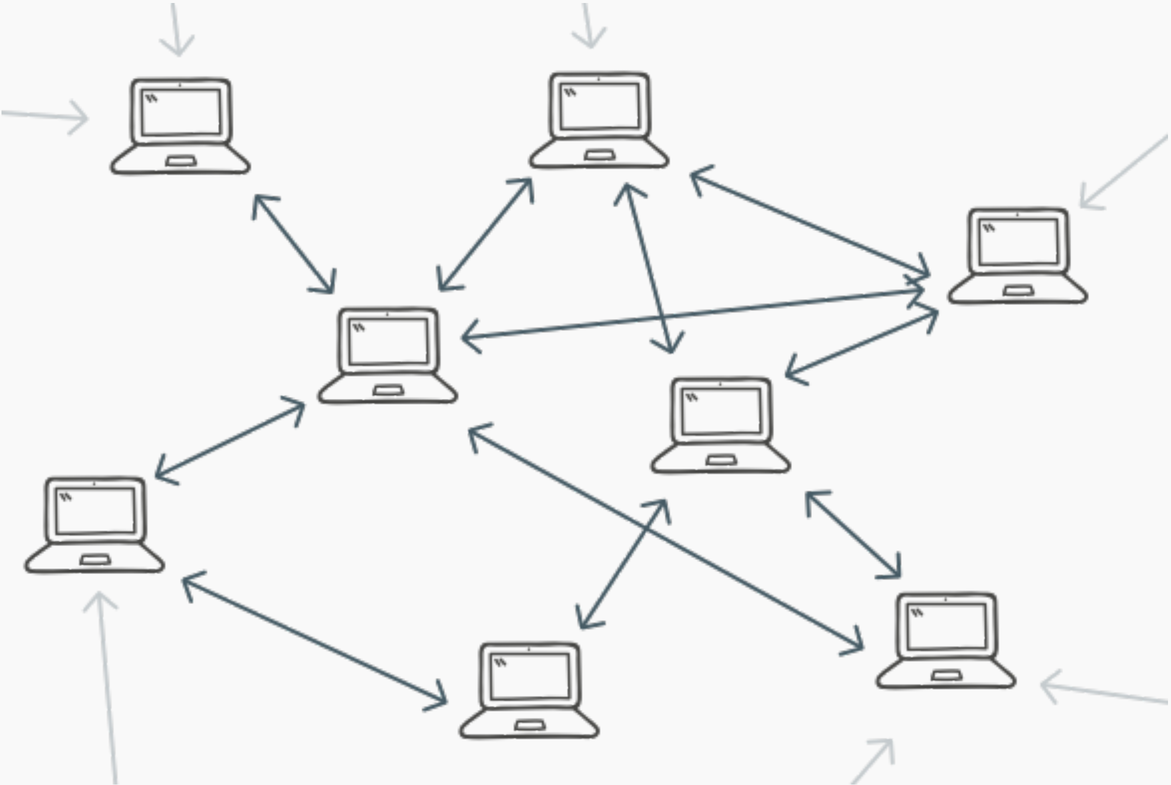
什麼是節點？

節點的定義

The definition of node

如果用一句話來描述區塊鏈中的節點，那就是訊息能夠被產生、接收或傳送的點，這可以是電腦、手機甚至是衛星。

節點間互相傳遞區塊、交易等訊息，來共同維護區塊鏈網路。



節點的定義

The definition of node

區塊鏈中的這些節點，也成為了比特幣等這些密碼貨幣 P2P 去中心化網路的本質。

這些節點根據不同區塊鏈系統中的協議，在分散式的網路中通訊，不論他們在真實世界中距離多遠。

無需中間人

抗審查性

02

節點類型

全節點

Full Node

全節點在區塊鏈網路中提供了安全性與可靠性。通常一個全節點會保存所有的區塊與交易，因此可以驗證網路中訊息的正確性。

運行一個以太坊全節點對設備的要求：

✦ 300GB 的可用儲存空間

✦ 高速以及不限流量的網路

✦ 8GB Ram

輕節點
Light Node

在輕節點上不保存所有區塊鏈上的資料，只保存區塊頭 (Block Header) 的數據。這種節點無法驗證全部的交易，只能驗證支付 (確認某筆交易存在區塊中，以及確認多少次)。

運行一個以太坊輕節點對設備的要求：

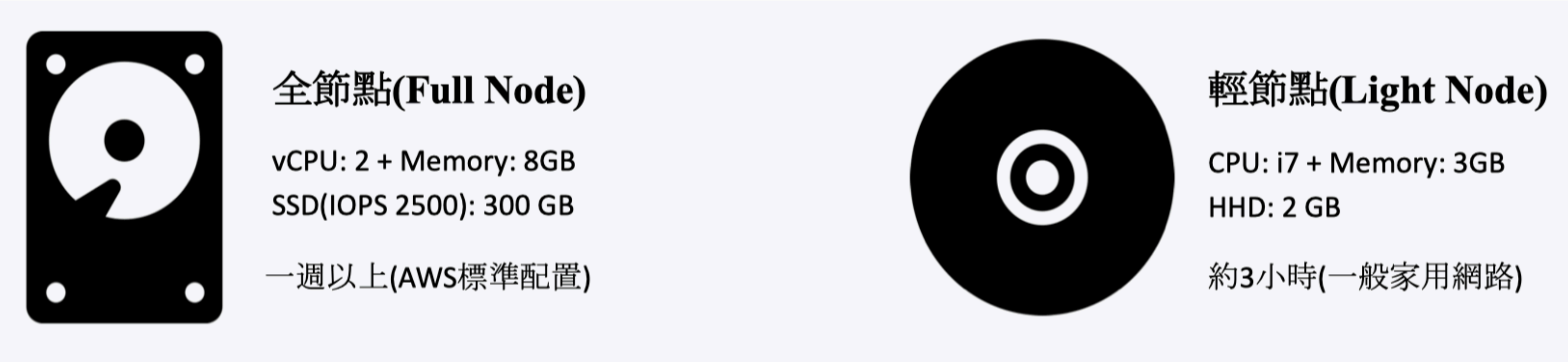
✦ 2 GB 儲存空間

03

節點同步

同步所需時間

Time needed for syncing



全節點(Full Node)

vCPU: 2 + Memory: 8GB

SSD(IOPS 2500): 300 GB

一週以上(AWS標準配置)

輕節點(Light Node)

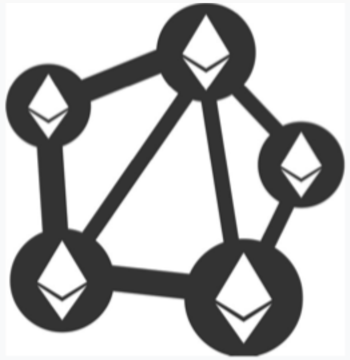
CPU: i7 + Memory: 3GB

HHD: 2 GB

約3小時(一般家用網路)

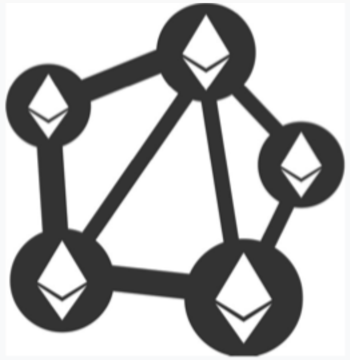
主鏈與測試鏈

Maine and Testnet



主鏈 Mainnet

- 正式網路 (Production)
- Network ID = 1



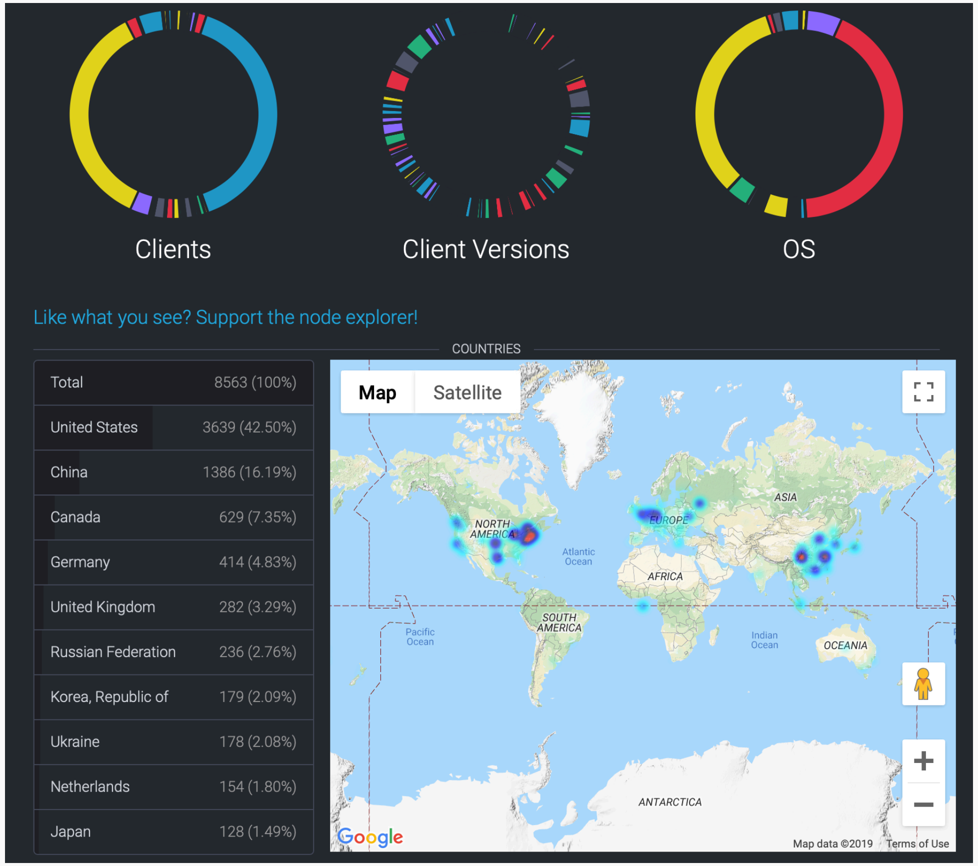
測試鏈 Testnet

- 測試網路
- Network ID != 1

Network ID	Code	Usage
1	Metropolis	Ethereum public main network
3	Ropsten	Ethereum cross-client public test network
4	Rinkeby	The public Geth client test network
42	Kovan	The public Parity client test network

網路節點狀態

Network Node Visualisation



Like what you see? Support the node explorer!

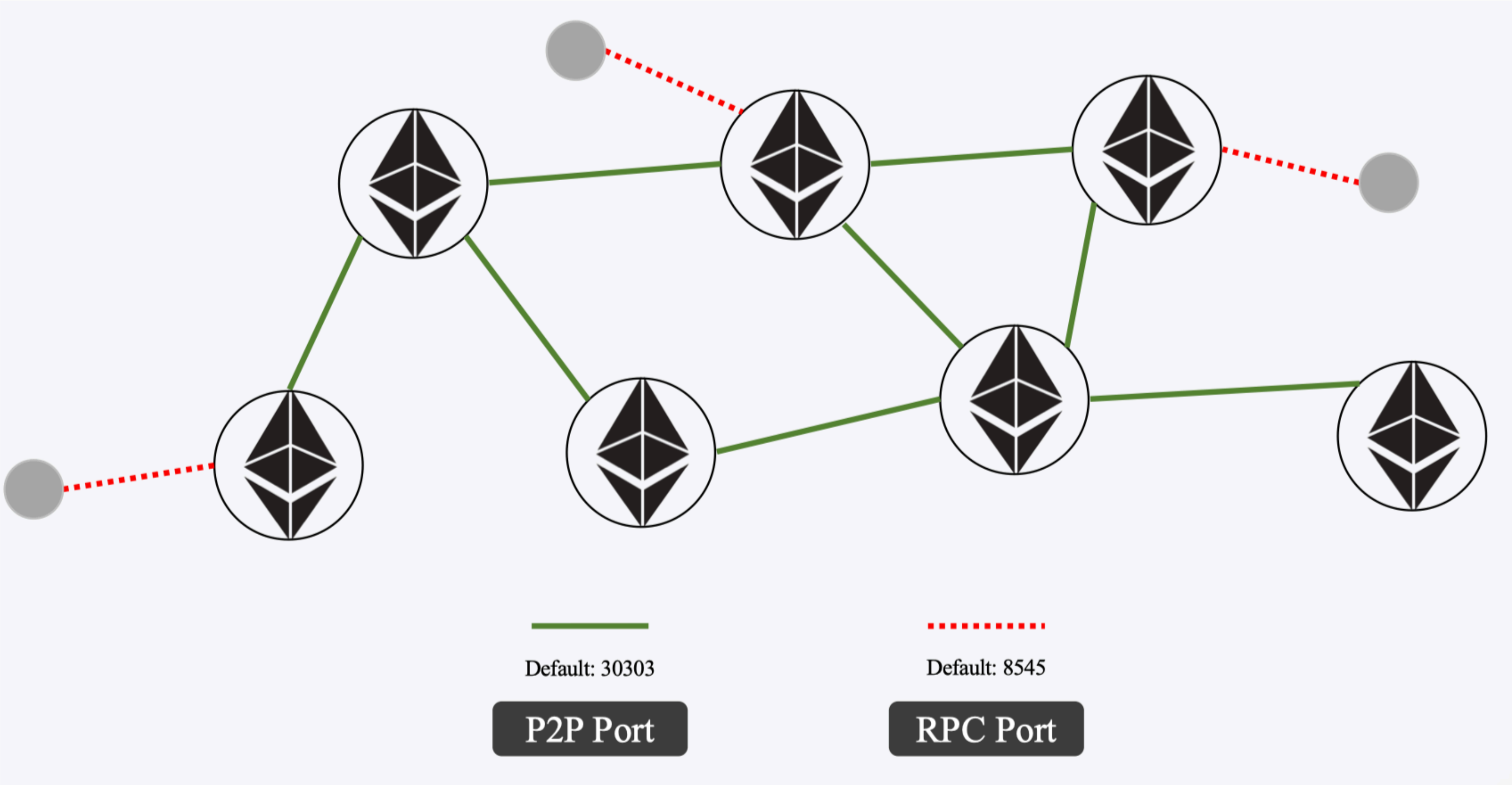
COUNTRIES

Total	8563 (100%)
United States	3639 (42.50%)
China	1386 (16.19%)
Canada	629 (7.35%)
Germany	414 (4.83%)
United Kingdom	282 (3.29%)
Russian Federation	236 (2.76%)
Korea, Republic of	179 (2.09%)
Ukraine	178 (2.08%)
Netherlands	154 (1.80%)
Japan	128 (1.49%)

https://www.ethernodes.org/network/1

節點的 Port

Node Port

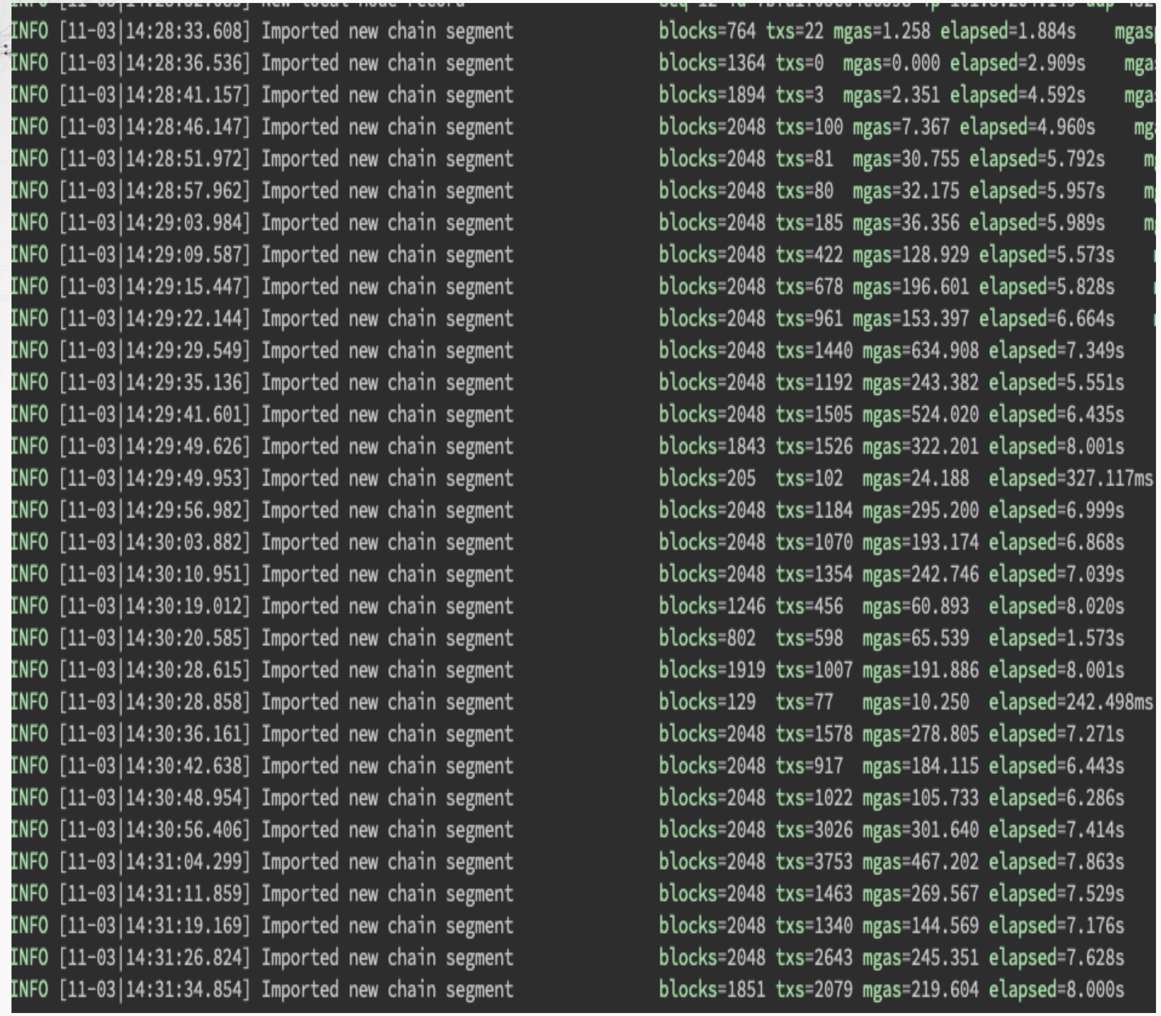


Default: 30303

P2P Port

Default: 8545

RPC Port



```
INFO [11-03|14:28:33.608] Imported new chain segment          blocks=764 txs=22 mgas=1.258 elapsed=1.884s    mgas
INFO [11-03|14:28:36.536] Imported new chain segment          blocks=1364 txs=0  mgas=0.000 elapsed=2.909s    mga
INFO [11-03|14:28:41.157] Imported new chain segment          blocks=1894 txs=3  mgas=2.351 elapsed=4.592s    mga
INFO [11-03|14:28:46.147] Imported new chain segment          blocks=2048 txs=100 mgas=7.367 elapsed=4.960s    mg
INFO [11-03|14:28:51.972] Imported new chain segment          blocks=2048 txs=81  mgas=30.755 elapsed=5.792s    m
INFO [11-03|14:28:57.962] Imported new chain segment          blocks=2048 txs=80  mgas=32.175 elapsed=5.957s    m
INFO [11-03|14:29:03.984] Imported new chain segment          blocks=2048 txs=185 mgas=36.356 elapsed=5.989s    m
INFO [11-03|14:29:09.587] Imported new chain segment          blocks=2048 txs=422 mgas=128.929 elapsed=5.573s
INFO [11-03|14:29:15.447] Imported new chain segment          blocks=2048 txs=678 mgas=196.601 elapsed=5.828s
INFO [11-03|14:29:22.144] Imported new chain segment          blocks=2048 txs=961 mgas=153.397 elapsed=6.664s
INFO [11-03|14:29:29.549] Imported new chain segment          blocks=2048 txs=1440 mgas=634.908 elapsed=7.349s
INFO [11-03|14:29:35.136] Imported new chain segment          blocks=2048 txs=1192 mgas=243.382 elapsed=5.551s
INFO [11-03|14:29:41.601] Imported new chain segment          blocks=2048 txs=1505 mgas=524.020 elapsed=6.435s
INFO [11-03|14:29:49.626] Imported new chain segment          blocks=1843 txs=1526 mgas=322.201 elapsed=8.001s
INFO [11-03|14:29:49.953] Imported new chain segment          blocks=205  txs=102  mgas=24.188  elapsed=327.117ms
INFO [11-03|14:29:56.982] Imported new chain segment          blocks=2048 txs=1184 mgas=295.200 elapsed=6.999s
INFO [11-03|14:30:03.882] Imported new chain segment          blocks=2048 txs=1070 mgas=193.174 elapsed=6.868s
INFO [11-03|14:30:10.951] Imported new chain segment          blocks=2048 txs=1354 mgas=242.746 elapsed=7.039s
INFO [11-03|14:30:19.012] Imported new chain segment          blocks=1246 txs=456  mgas=60.893  elapsed=8.020s
INFO [11-03|14:30:20.585] Imported new chain segment          blocks=802  txs=598  mgas=65.539  elapsed=1.573s
INFO [11-03|14:30:28.615] Imported new chain segment          blocks=1919 txs=1007 mgas=191.886 elapsed=8.001s
INFO [11-03|14:30:28.858] Imported new chain segment          blocks=129  txs=77   mgas=10.250  elapsed=242.498ms
INFO [11-03|14:30:36.161] Imported new chain segment          blocks=2048 txs=1578 mgas=278.805 elapsed=7.271s
INFO [11-03|14:30:42.638] Imported new chain segment          blocks=2048 txs=917  mgas=184.115 elapsed=6.443s
INFO [11-03|14:30:48.954] Imported new chain segment          blocks=2048 txs=1022 mgas=105.733 elapsed=6.286s
INFO [11-03|14:30:56.406] Imported new chain segment          blocks=2048 txs=3026 mgas=301.640 elapsed=7.414s
INFO [11-03|14:31:04.299] Imported new chain segment          blocks=2048 txs=3753 mgas=467.202 elapsed=7.863s
INFO [11-03|14:31:11.859] Imported new chain segment          blocks=2048 txs=1463 mgas=269.567 elapsed=7.529s
INFO [11-03|14:31:19.169] Imported new chain segment          blocks=2048 txs=1340 mgas=144.569 elapsed=7.176s
INFO [11-03|14:31:26.824] Imported new chain segment          blocks=2048 txs=2643 mgas=245.351 elapsed=7.628s
INFO [11-03|14:31:34.854] Imported new chain segment          blocks=1851 txs=2079 mgas=219.604 elapsed=8.000s
```

全節點同步

Sync full node

Command

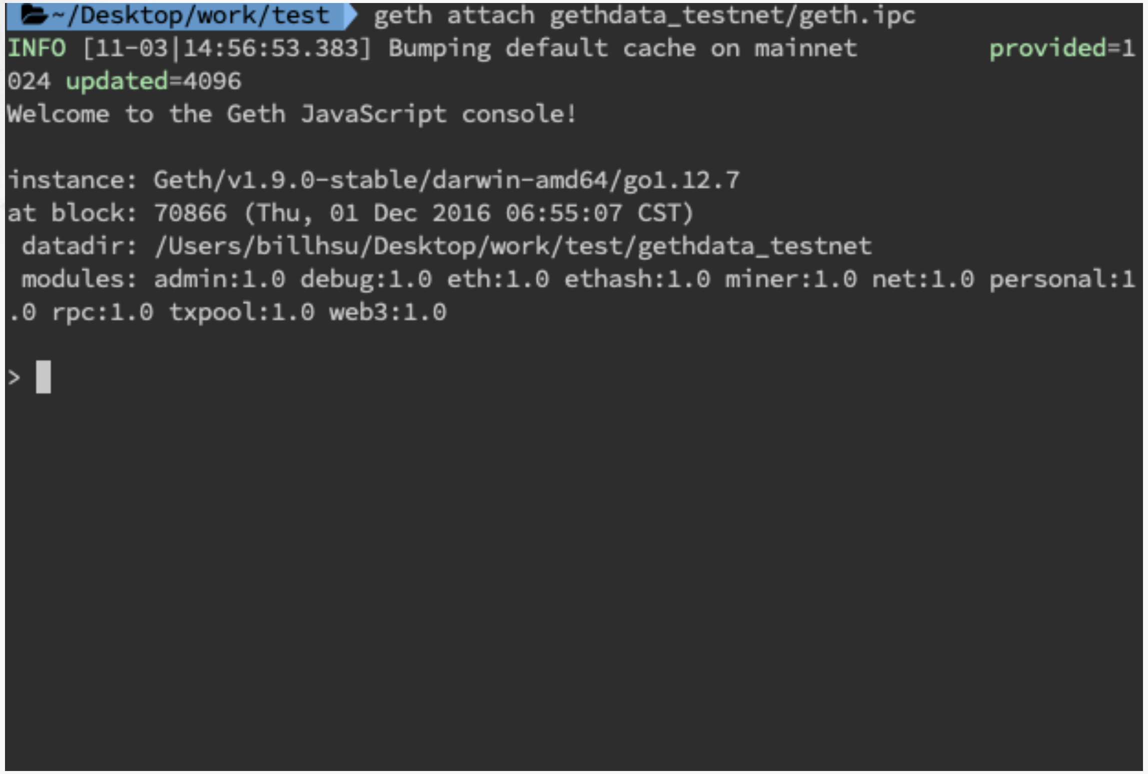
```
geth \
--syncmode full \
--datadir gethdata_testnet \
--rpcport 8545 \
--rpc --rpccorsdomain "*" --rpcaddr 0.0.0.0 \
--rpcapi "db,eth,net,web3,personal" \
—testnet --networkid=3
```

使用 geth 與節點互動

Interact with node

Command

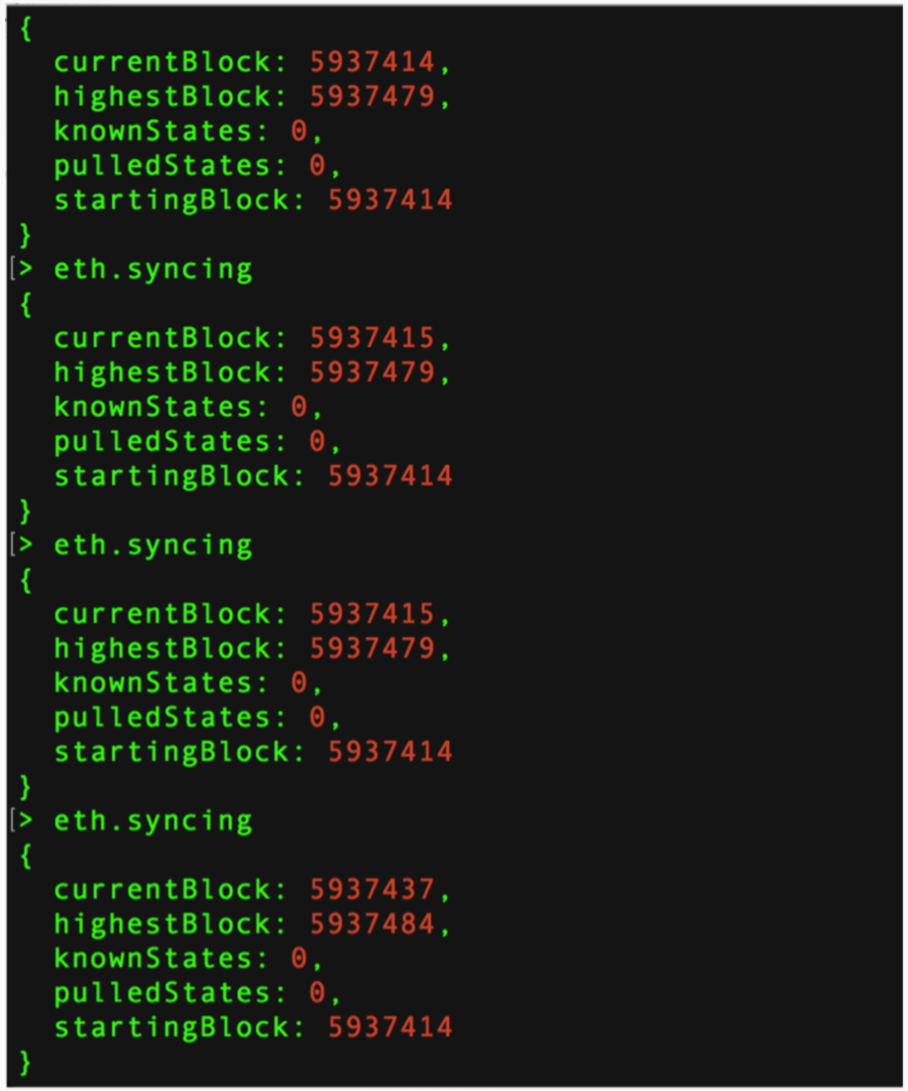
```
geth attach gethdata_testnet/geth.ipc
```



```
~/Desktop/work/test  geth attach gethdata_testnet/geth.ipc
INFO [11-03|14:56:53.383] Bumping default cache on mainnet         provided=1
024 updated=4096
Welcome to the Geth JavaScript console!

instance: Geth/v1.9.0-stable/darwin-amd64/go1.12.7
at block: 70866 (Thu, 01 Dec 2016 06:55:07 CST)
 datadir: /Users/billhsu/Desktop/work/test/gethdata_testnet
 modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1
.0 rpc:1.0 txpool:1.0 web3:1.0

>
```



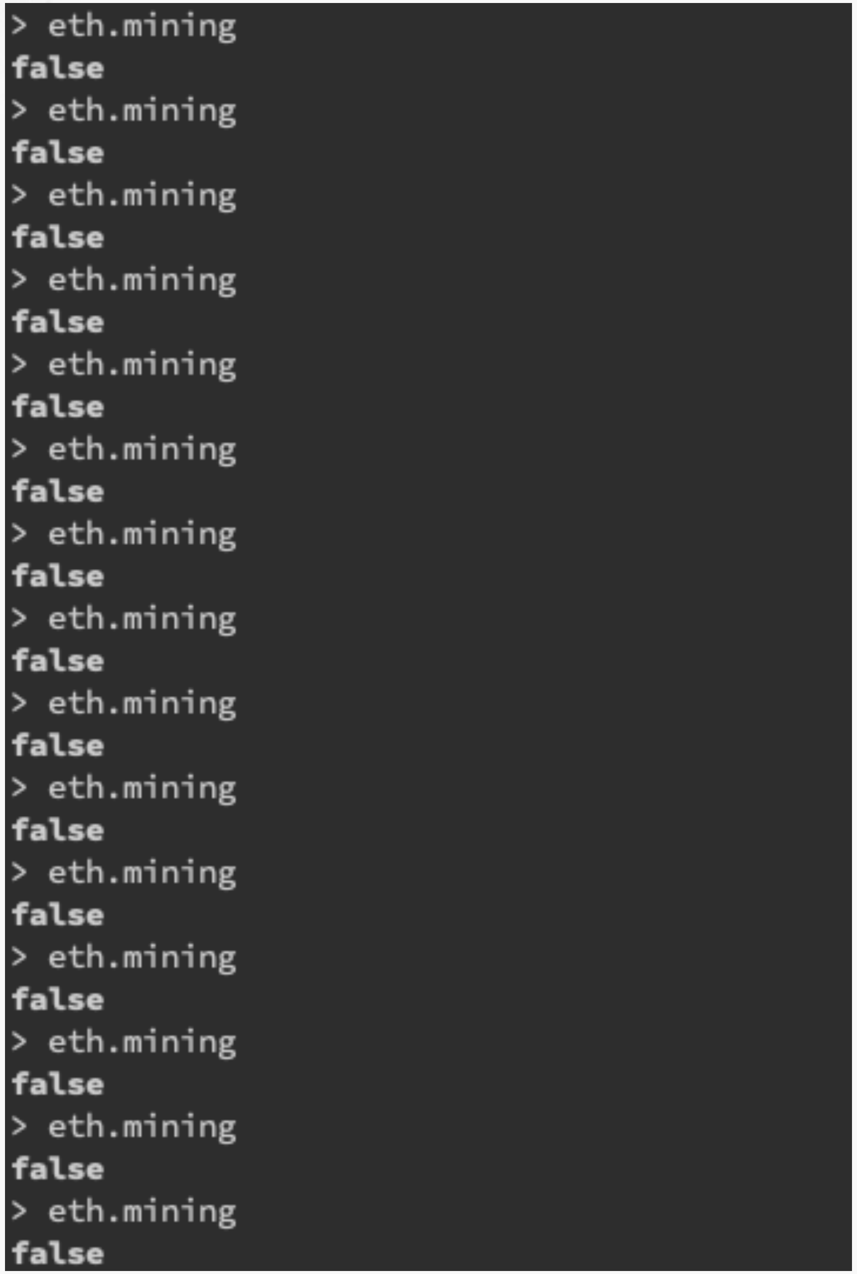
```
{
  currentBlock: 5937414,
  highestBlock: 5937479,
  knownStates: 0,
  pulledStates: 0,
  startingBlock: 5937414
}
[> eth.syncing
{
  currentBlock: 5937415,
  highestBlock: 5937479,
  knownStates: 0,
  pulledStates: 0,
  startingBlock: 5937414
}
[> eth.syncing
{
  currentBlock: 5937415,
  highestBlock: 5937479,
  knownStates: 0,
  pulledStates: 0,
  startingBlock: 5937414
}
[> eth.syncing
{
  currentBlock: 5937437,
  highestBlock: 5937484,
  knownStates: 0,
  pulledStates: 0,
  startingBlock: 5937414
}
```

節點同步狀態

Synchronous status

於 geth 控制台中輸入

```
eth.syncing
```



```
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
> eth.mining
false
```

檢查是否在挖礦

Check mining status

於 geth 控制台中輸入

```
eth.mining
```

異常狀況

Troubleshooting

Fatal: Error starting protocol stack:
listen unix /xxx.ipc: filaname too long

* 減少執行geth時的路徑層數

Fatal: Error starting protocol stack:
listen unix /xxx.ipc: bind: operation not permitted

* 以sudo執行
* 是否執行了另一個geth

執行後,始終沒有印出連線或sync訊息

* 檢查P2P port是否有開放

Synchronisation failed, dropping peer
err="action from bad peer ignored"

* 等待一段時間

Error, IO Wait

* 系統資源不足: SSD + Enough Memory

Unknown

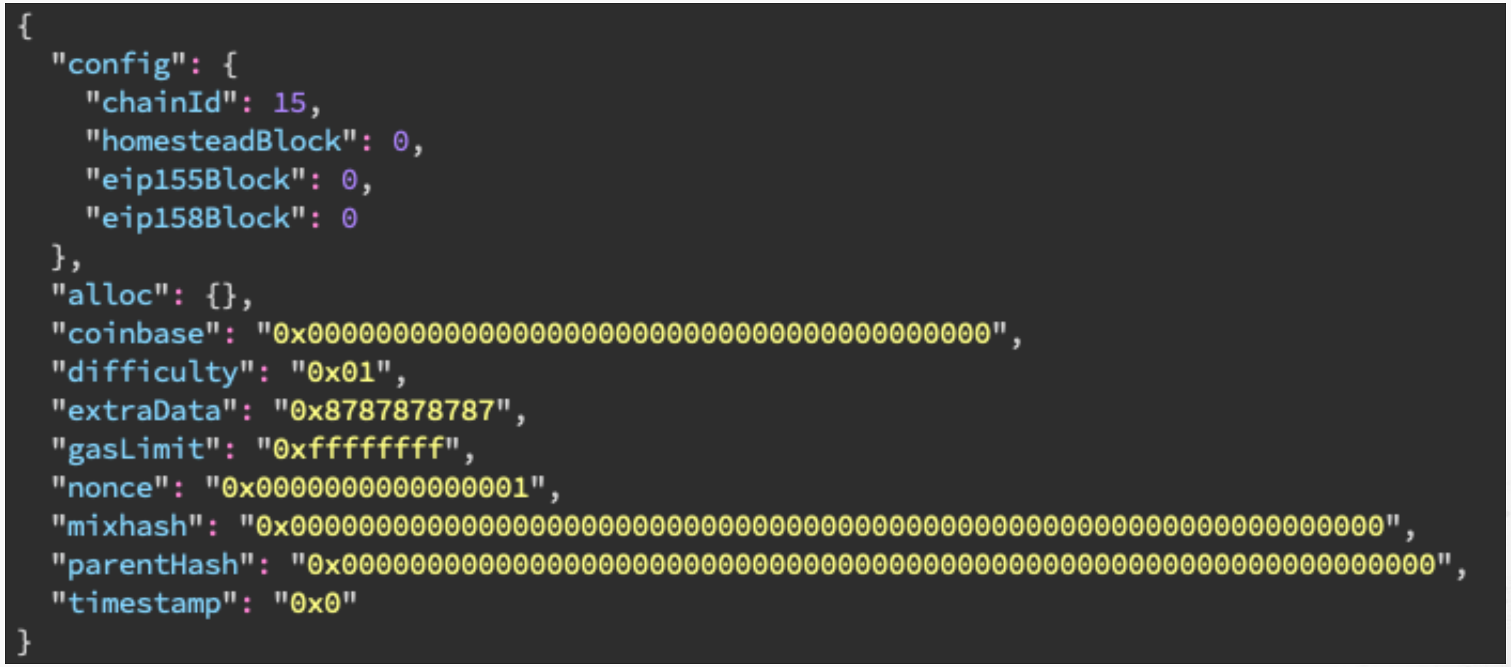
* Upgrade geth to latest version

架設私有鏈

Create your own private chain

1. 新增一個 geth 工作目錄

```
$ mkdir geth
$ cd geth
$ touch gensis.json
```



```
{
  "config": {
    "chainId": 15,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "alloc": {},
  "coinbase": "0x0000000000000000000000000000000000000000",
  "difficulty": "0x01",
  "extraData": "0x8787878787",
  "gasLimit": "0xffffffff",
  "nonce": "0x0000000000000001",
  "mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x0"
}
```

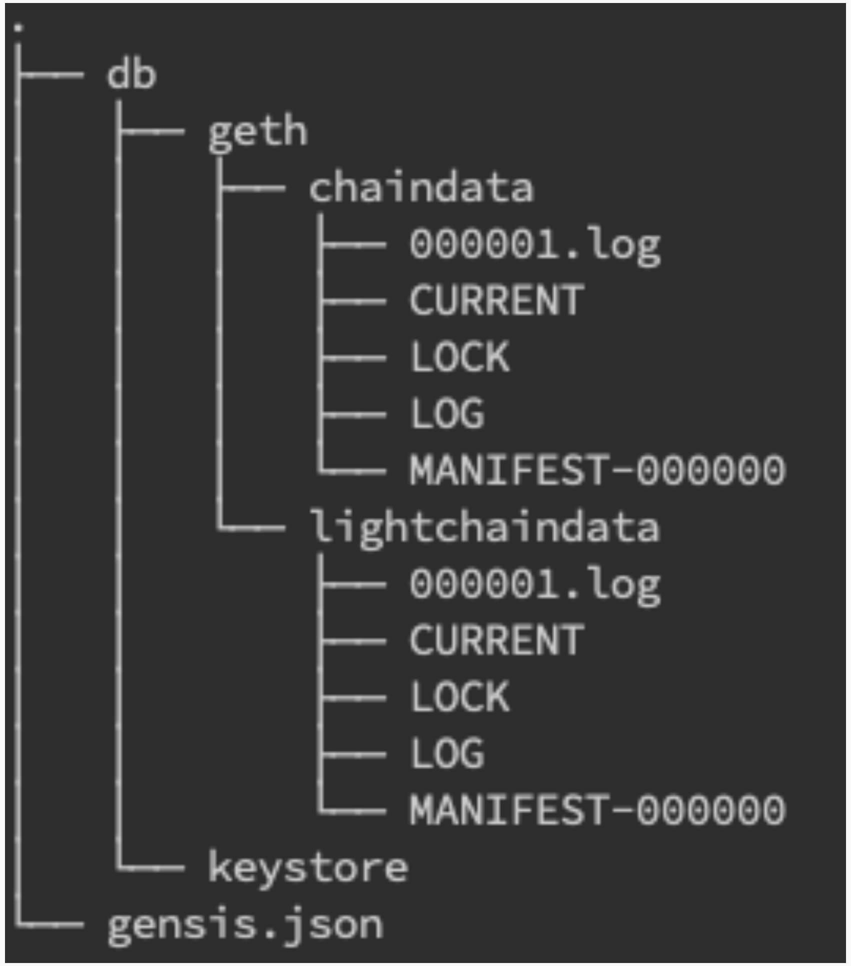
創世區塊 gensis.json

架設私有鏈

Create your own private chain

2. 初始化區塊鏈

```
$ mkdir db
$ geth --datadir db init gensis.json
```



```
.
├── db
│   ├── geth
│   │   ├── chaindata
│   │   │   ├── 000001.log
│   │   │   ├── CURRENT
│   │   │   ├── LOCK
│   │   │   ├── LOG
│   │   │   └── MANIFEST-000000
│   │   └── lightchaindata
│   │       ├── 000001.log
│   │       ├── CURRENT
│   │       ├── LOCK
│   │       ├── LOG
│   │       └── MANIFEST-000000
│   └── keystore
└── gensis.json
```

完成後的路徑圖

架設私有鏈

Create your own private chain

3. 啟動節點

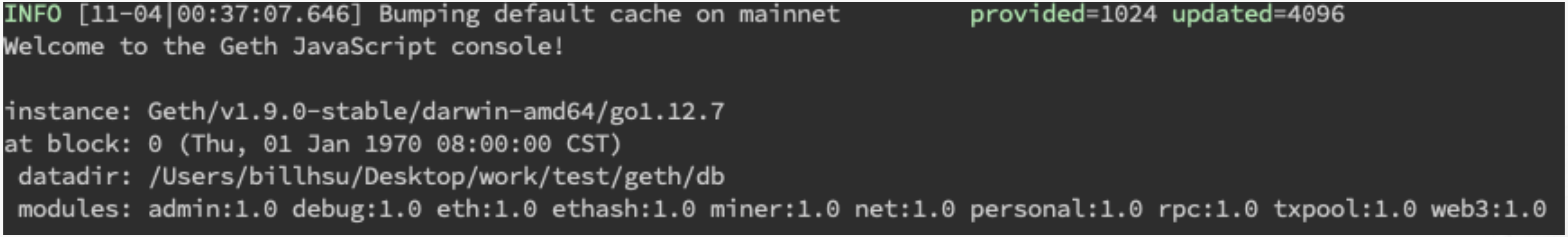
```
$ geth \
--datadir "./db" \
--rpc  --rpcaddr=0.0.0.0 --rpcport 8545 --rpccorsdomain "*" \
--rpcapi "eth,net,web3,personal,admin,shh,txpool,debug,miner" \
--nodiscover \
--networkid=1234 \
--port 30303 \
--allow-insecure-unlock
```

架設私有鏈

Create your own private chain

4. 開啟另一個視窗，連結啟動的節點

```
$ geth  attach  db/geth.ipc
```



```
INFO [11-04|00:37:07.646] Bumping default cache on mainnet         provided=1024 updated=4096
Welcome to the Geth JavaScript console!

instance: Geth/v1.9.0-stable/darwin-amd64/go1.12.7
at block: 0 (Thu, 01 Jan 1970 08:00:00 CST)
 datadir: /Users/billhsu/Desktop/work/test/geth/db
 modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0
```

執行後的結果

架設私有鏈

Create your own private chain

5. 新增帳戶

```
> personal.newAccount("password")
"0xa56aeaa043a10dc693bd6ae1fd3743161c2bcfeb"
```

為了稍後的轉帳練習，先產生兩個帳戶，接著查看所有帳戶：

```
> eth.accounts
["0xa56aeaa043a10dc693bd6ae1fd3743161c2bcfeb",
"0x5f5185b64ceac5345016b831ef0d9843a311f869"]
```

架設私有鏈

Create your own private chain

6. 轉帳的前提是要有餘額，我們先檢視剛剛產生的帳戶餘額

```
> web3.fromWei(eth.getBalance(eth.accounts[0]), "ether")
0
> web3.fromWei(eth.getBalance(eth.accounts[1]), "ether")
0
```

可以發現這兩個帳戶裡面都沒有錢，如果有錢，那麼恭喜你運氣很好。
如果沒有錢，可以透過兩種方式獲得，透過別人轉給你，或自己挖礦。

架設私有鏈

Create your own private chain

7. 為了取得以太幣，我們需要開啟挖礦，但在開始前需要先設定挖礦獎勵的地址

```
> miner.setEtherbase(eth.accounts[0])
true
```

設定完成後，檢查是否設定成功

```
> eth.coinbase
"0xa56aeaa043a10dc693bd6ae1fd3743161c2bcfeb"
```

eth.coinbase 傳回挖礦獎勵地址，如果與你的地址相符那麼就設定成功了。接下來就要開挖了！

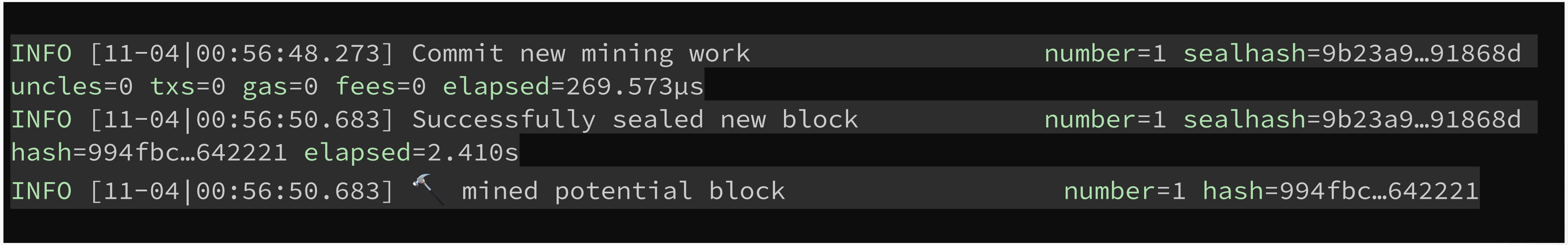
架設私有鏈

Create your own private chain

8. 開始挖礦

```
> miner.start(1)
```

回到節點的視窗



```
INFO [11-04|00:56:48.273] Commit new mining work                    number=1 sealhash=9b23a9…91868d
uncles=0 txs=0 gas=0 fees=0 elapsed=269.573µs
INFO [11-04|00:56:50.683] Successfully sealed new block             number=1 sealhash=9b23a9…91868d
hash=994fbc…642221 elapsed=2.410s
INFO [11-04|00:56:50.683] 🔨 mined potential block                   number=1 hash=994fbc…642221
```

如果可以看到小錘子，那就代表成功挖到區塊了！

架設私有鏈

Create your own private chain

9. 查看挖礦獎勵

```
> miner.stop()
null
> web3.fromWei(eth.getBalance(eth.accounts[0]), "ether")
70
```

10. 具備轉帳的前置條件後，在發送交易前我們需要先解鎖帳戶

```
> personal.unlockAccount(eth.accounts[0], "password")
true
```

架設私有鏈

Create your own private chain

11. 一切準備就緒，可以開始轉帳了

```
> eth.sendTransaction({ from: eth.accounts[0], to: eth.accounts[1], value: web3.toWei(1, "ether") })
"0x488470423854c9fb3b96d968c6d32dff15b308f1eac6575b6c9f4a70b743d4be"
```

檢視交易池中等待被打包的交易

```
> txpool.status
{
    pending: 1,
    queued: 0
}
```

架設私有鏈

Create your own private chain

12. 要使交易被打包到區塊中，需要透過進行挖礦

```
> miner.start(1);  admin.sleepBlocks(1);  miner.stop();
```

現在交易已經成功被打包，並加入區塊鏈中，我們可以檢視一下帳戶的餘額變化

```
> web3.fromWei(eth.getBalance(eth.accounts[0]), "ether")
74
> web3.fromWei(eth.getBalance(eth.accounts[1]), "ether")
1
```

04

來自前線的報導

實務經驗談

Practical Experience

節點的維護很辛苦

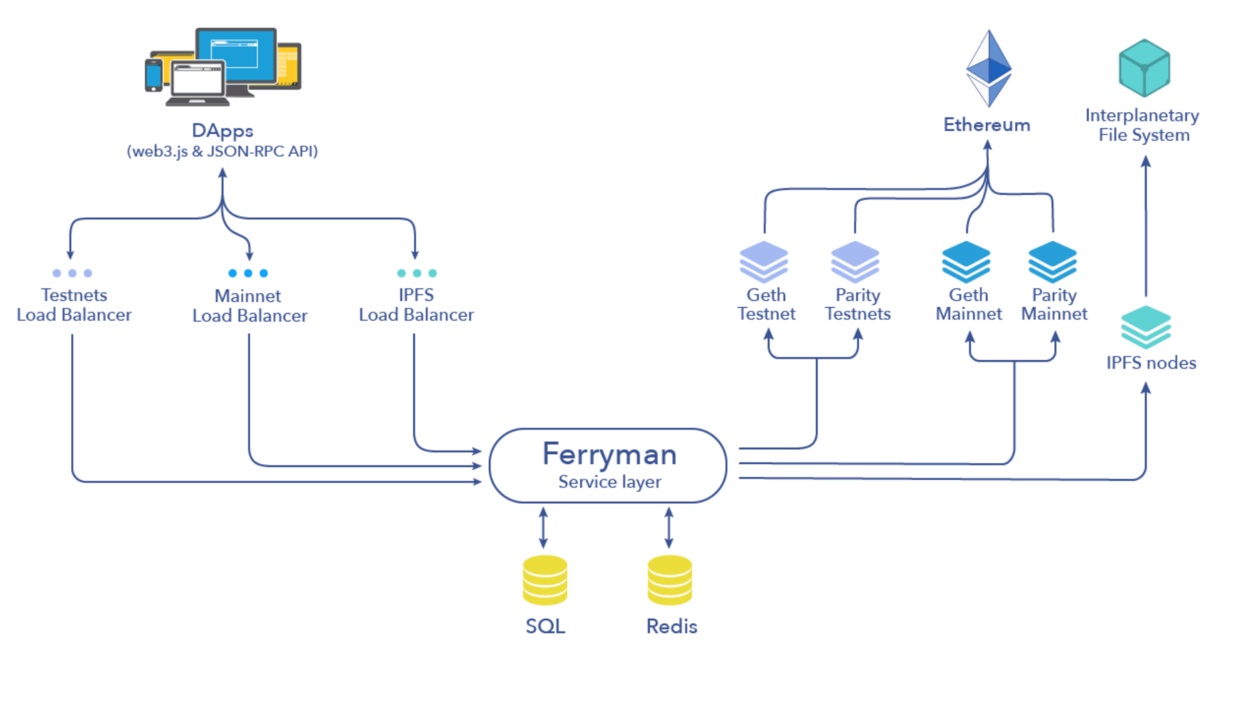
- 養節點
- 附載平衡

一個節點好貴

One full node = 2000 ~ 4000 NTD/Month

第三方服務的選擇

- BlockCypher for Bitcoin
- Infura for Ethereum



實務經驗談 安全篇

Practical Experience. — Security

- RPC Port 管理

- 日蝕攻擊 (Eclipse Attack)

谢谢

THANK YOU