部署thegraph私有节点

因为要搞的链是thegraph托管服务不支持的链（托管服务不支持的链或私有链），所以需要部署一个thegraph私有节点去同步链的信息，给subgraph查询提供服务。
必要信息：

- 你的链的rpc节点

一、部署thegraph私有节点

首先参考了剖析DeFi借贷产品之Compound：Subgraph篇

> 若要自己搭建私有节点，可按照 Github 上的 graph-node 项目的说明进行部署。其 Github 地址为：
> https://github.com/graphprotocol/graph-node
> 部署 graph-node 也有两种方式，一是 README 上所描述的步骤，二是用 Docker 进行部署。两种方式我都尝试过，但第一种方式以失败告终，多次尝试解决问题依然无果，第二种方式很快就成功了，所以我强烈推荐用 Docker 方式进行部署。
> 首先，在需要部署的服务器安装好 docker 和 docker-compose。
> 其次，打开 graph-node/docker/docker-compose.yml 文件，修改其中一行：
> ethereum: 'mainnet:http://host.docker.internal:8545'
> 该行指定了使用的网络和节点，比如，我部署接入 kovan 网络，节点使用 infura 的，那设置的值为：
> ethereum: 'kovan:https://kovan.infura.io/v3/<PROJECT_ID>'
> 其中，<PROJECT_ID> 是在 infura 注册项目时所分配的项目ID。

果断决定使用docker-compose 的方法进行部署。
然后找到thegraph Github 地址,按着说明进行部署，因为我是mac所以决定先用mac的方式进行部署

Running Graph Node on an Macbook M1

We do not currently build native images for Macbook M1, which can lead to processes being killed due to out-of-memory errors (code 137). Based on the example `docker-compose.yml` is possible to rebuild the image for your M1 by running the following, then running `docker-compose up` as normal:

> **Important** Increase memory limits for the docker engine running on your machine. Otherwise docker build command will fail due to out of memory error. To do that, open docker-desktop and go to Resources/advanced/memory.

```
# Remove the original image
docker rmi graphprotocol/graph-node:latest

# Build the image
./docker/build.sh

# Tag the newly created image
docker tag graph-node graphprotocol/graph-node:latest
```

新建一个文件夹，

```
git clone https://github.com/graphprotocol/graph-node
```

然后进入到docker 目录，按一下步骤进行。

```
# Remove the original image
docker rmi graphprotocol/graph-node:latest

# Build the image
./docker/build.sh

# Tag the newly created image
docker tag graph-node graphprotocol/graph-node:latest
```

在进行./docker/build.sh 可能会出现错误，这个时候我是编译出来了两个镜像：没有graph-node

```
hanpeng@hanpeng docker % docker images
REPOSITORY         TAG                             IMAGE ID       CREATED
graph-node-debug   latest                          21cbe2b289ad   30 minutes ago
graph-node-build   latest                          c90de47737e2   32 minutes ago
```

但是多试几次，没有报错的时候是三个
然后将graph-node tag一下（与docker-compose 文件中的services:
graph-node:
image: graphprotocol/graph-node
对应）

```
hanpeng@hanpeng docker % docker images
REPOSITORY                  TAG                             IMAGE ID       CREATED
graph-node-debug            latest                          02367ba98882   2 days
graph-node                  latest                          fcb4f4e89bf1   2 days
graphprotocol/graph-node    latest                          fcb4f4e89bf1   2 days
graph-node-build            latest                          bb5c56a1ae88   2 days
```

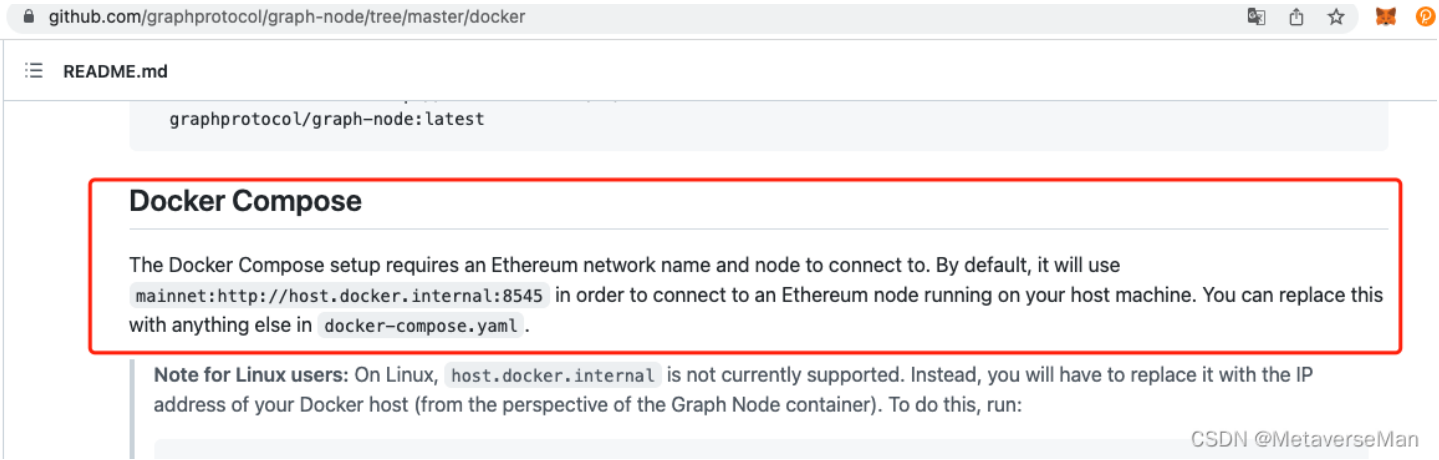
再然后修改docker-compose文件中

```
ethereum: 'mainnet:http://host.docker.internal:8545'
```

改为你要监控的链的rpc节点信息：

```
ethereum: 'rangersprotocl:https://robin.rangersprotocol.com/api/jsonrpc'
```

在这里是有疑惑的，因为这个配置开头是ethereum，虽然这里说可以任意repalce但是具体怎么replace，哪些是关键字，有没有校验，并没有说明。



而且后边还有一个rpc节点的名称，不知道怎么填。
索性就还按ethereum这样填了,然后后边的rpc节点名称就用我使用的公链名填，还填错了，（我们用的是rangersprotocol，我填的是rangersprotocl,少了一个o，由此可见，这些并没有规定和校验，只是一个用作标识符的名字）
但是你这里怎么填，后边子图的subgraph.yaml要对应上，不然subgraph会找不到thegraph节点。
修改结束后，就可以docker-compose up -d 启动了。
docker-compose ps 看一下：

```
hanpeng@hanpeng docker % docker-compose ps
        Name                       Command               State
-------------------------------------------------------------------------------
docker_graph-node_1   /bin/sh -c start                 Up      0.0.0.0:8000->800
                                                               0.0.0.0:8040->804
docker_ipfs_1         /sbin/tini -- /usr/local/b ...   Up      4001/tcp, 4001/ud
docker_postgres_1     docker-entrypoint.sh postg ...   Up      0.0.0.0:5432->543
```

服务都起来了。
接下来就是将子图部署到graph私有节点了。

二、准备合约

- 准备一个合约，将合约编译获得ABI
- 部署合约，获得合约地址

我这里还是用的之前写的game那个合约，参考：
https://github.com/MetaverseMan/contractSample
将合约部署到你的rpc节点对应的的链上，不要粗心搞错了。

三、部署子图到私有节点

1：准备子图
这里要参考一篇官方文章：https://thegraph.academy/developers/local-development/

我们参考文章里的3和5
graph-cli我已经安装过了，我从3.B开始。

B) Creating an example subgraph

Run the following command inside the folder on your machine you want to create a subgraph in:

```
$ graph init --from-example <GITHUB_USERNAME>/<SUBGRAPH_NAME> [<DIRECTORY>]
```

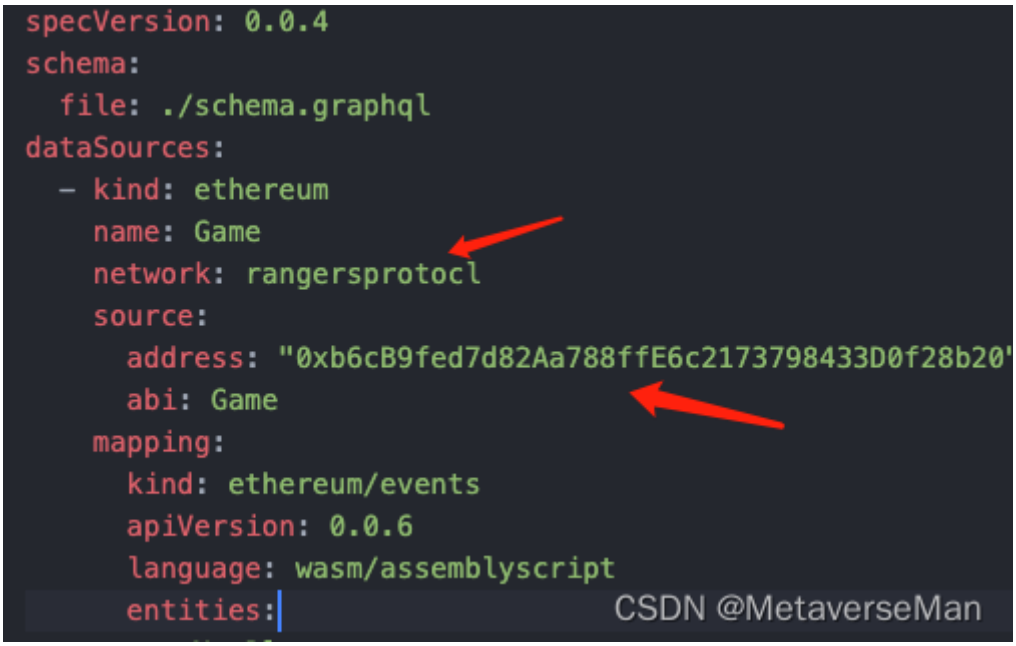
Replace the placeholders with the following credentials:

- `<GITHUB_USERNAME>` is required. It's your GitHub username
- `<SUBGRAPH_NAME>` is required. It's the name of your subgraph project
- `<DIRECTORY>` is optional. It's the name of the directory that your subgraph is created in. Defaults to `<SUBGRAPH_NAME>`.

By running the command you create an example subgraph. You can use the subgraph to build your own subgraph on top of it. In the next tutorial, we will show you how to define a subgraph.

```
graph init --from-example MetaverseMan/gameonrangers
```

然后修改subgraph.yaml，schema.graphql，package.json这三个文件。（还需要将ABI文件和Game.ts文件复制粘贴到对应目录）
我这里是用之前已经修改好的文件直接替换了subgraph.yaml和schema.graphql（将原来的重命名为别的），其中subgraph.yaml这里要注意：



network要和你启动thegraph私有节点的docker-compose文件里网络分类一致，我这里就将错就错，填rangersprotocl。

一定记得要看一下package.json

我把这种

```
1 --ipfs https://api.thegraph.com/ipfs/ --node https://api.thegraph.com/deploy/
```

都替换成了对应的

```
1 --ipfs http://127.0.0.1:5001 --node http://127.0.0.1:8020
```

但其实init项目的时候create-local和deploy-local是配置好的，你只要检查下是否是本地的路由和端口。如果不是，修改create-local和deploy-local就行了。
我的：

```
1 {
2   "name": "gameonrangers",
3   "version": "0.1.0",
4   "scripts": {
5     "build-contract": "solc contracts/Gravity.sol --abi -o abis --overwrite && s
6     "create": "graph create MetaverseMan/gameonrangers --node http://127.0.0.1:8
7     "create-local": "graph create MetaverseMan/gameonrangers --node http://127.0
8     "codegen": "graph codegen",
```

```
    "build": "graph build",
    "deploy": "graph deploy MetaverseMan/gameonrangers  --ipfs http://127.0.0.1:
    "deploy-local": "graph deploy MetaverseMan/gameonrangers  --node http://127.
  },
  "devDependencies": {
    "@graphprotocol/graph-cli": "^0.30.2",
    "@graphprotocol/graph-ts": "^0.27.0"
  },
  "dependencies": {
    "babel-polyfill": "^6.26.0",
    "babel-register": "^6.26.0",
    "truffle": "^5.0.4",
    "truffle-contract": "^4.0.5",
    "truffle-hdwallet-provider": "^1.0.4"
  }
}
```

修改完成后，就可以部署子图了。

2：部署子图

在你的子图项目的根目录参考https://thegraph.academy/developers/local-development/的5，运行命令行。主要有以下命令

```
1: sed -i -e 's/0x2E645469f354BB4F5c8a05B3b30A929361cf77eC/0xb6cB9fed7d82Aa788ff
2: yarn codegen
3: yarn create-local
4: yarn deploy-local
```

也可以在codegen之后加上yarn build

```
hanpeng@hanpeng gameonrangers % sed -i -e 's/0x2E645469f354BB4F5c8a05B3b30A92936
hanpeng@hanpeng gameonrangers % yarn codegen
yarn run v1.22.18
warning ../../../../package.json: No license field
$ graph codegen
  Skip migration: Bump mapping apiVersion from 0.0.1 to 0.0.2
  Skip migration: Bump mapping apiVersion from 0.0.2 to 0.0.3
  Skip migration: Bump mapping apiVersion from 0.0.3 to 0.0.4
  Skip migration: Bump mapping apiVersion from 0.0.4 to 0.0.5
  Skip migration: Bump mapping apiVersion from 0.0.5 to 0.0.6
  Skip migration: Bump manifest specVersion from 0.0.1 to 0.0.2
  Skip migration: Bump manifest specVersion from 0.0.2 to 0.0.4
✔ Apply migrations
```

```
✔ Load subgraph from subgraph.yaml
  Load contract ABI from abis/Game.json
✔ Load contract ABIs
  Generate types for contract ABI: Game (abis/Game.json)
  Write types to generated/Game/Game.ts
✔ Generate types for contract ABIs
✔ Generate types for data source templates
✔ Load data source template ABIs
✔ Generate types for data source template ABIs
✔ Load GraphQL schema from schema.graphql
  Write types to generated/schema.ts
✔ Generate types for GraphQL schema

Types generated successfully

✨  Done in 3.01s.
hanpeng@hanpeng gameonrangers % yarn create-local
yarn run v1.22.18
warning ../../../../package.json: No license field
$ graph create MetaverseMan/gameonrangers --node http://127.0.0.1:8020
Created subgraph: MetaverseMan/gameonrangers
✨  Done in 1.68s.
hanpeng@hanpeng gameonrangers % yarn deploy-local
yarn run v1.22.18
warning ../../../../package.json: No license field
$ graph deploy MetaverseMan/gameonrangers  --node http://127.0.0.1:8020 --ipfs h
? Version Label (e.g. v0.0.1) › (node:98385) ExperimentalWarning: The Fetch API
(Use `node --trace-warnings ...` to show where the warning was created)
✔ Version Label (e.g. v0.0.1) ·
  Skip migration: Bump mapping apiVersion from 0.0.1 to 0.0.2
  Skip migration: Bump mapping apiVersion from 0.0.2 to 0.0.3
  Skip migration: Bump mapping apiVersion from 0.0.3 to 0.0.4
  Skip migration: Bump mapping apiVersion from 0.0.4 to 0.0.5
  Skip migration: Bump mapping apiVersion from 0.0.5 to 0.0.6
  Skip migration: Bump manifest specVersion from 0.0.1 to 0.0.2
  Skip migration: Bump manifest specVersion from 0.0.2 to 0.0.4
✔ Apply migrations
✔ Load subgraph from subgraph.yaml
  Compile data source: Game => build/Game/Game.wasm
✔ Compile subgraph
  Copy schema file build/schema.graphql
  Write subgraph file build/Game/abis/Game.json
  Write subgraph manifest build/subgraph.yaml
✔ Write compiled subgraph to build/
  Add file to IPFS build/schema.graphql
                .. Qmc8mBqzRArjzreqnyBF4PEargATRKuaLesmGm7vbXALQ8
  Add file to IPFS build/Game/abis/Game.json
```

```
                .. QmapPy7RyHEGVX7Fpp8ZgjdeeDXGN3JA2xnvJzbV78R2rP
  Add file to IPFS build/Game/Game.wasm
                .. QmdW8vg7vvPJBXnPxx3dQURBdi3Peq3yiKCJSdbYsjdddG
✔ Upload subgraph to IPFS

Build completed: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263

Deployed to http://127.0.0.1:8000/subgraphs/name/MetaverseMan/gameonrangers/grap

Subgraph endpoints:
Queries (HTTP):     http://127.0.0.1:8000/subgraphs/name/MetaverseMan/gameonrang

✨  Done in 121.49s.
```

这里有一个小坑：

在执行yarn deploy-local后，是一个问号开头的log这里需要按一下enter键！！！

丫的，我以为跟上边一样呢，直接打印执行结果，或者就进入了一个进程，占用了一个窗口。但其实是需要按enter键确认，我每次都是等结果，没反应，然后就ctrl+c结束掉了。
另外 两个网络的名字一定要一致不然是会有以下错误：

最后终于成功！

```
hanpeng@hanpeng gameonrangers % yarn deploy-local
yarn run v1.22.18
warning ../../../../package.json: No license field
$ graph deploy MetaverseMan/gameonrangers  --node http://127.0.0.1:8020 --ipfs h
```

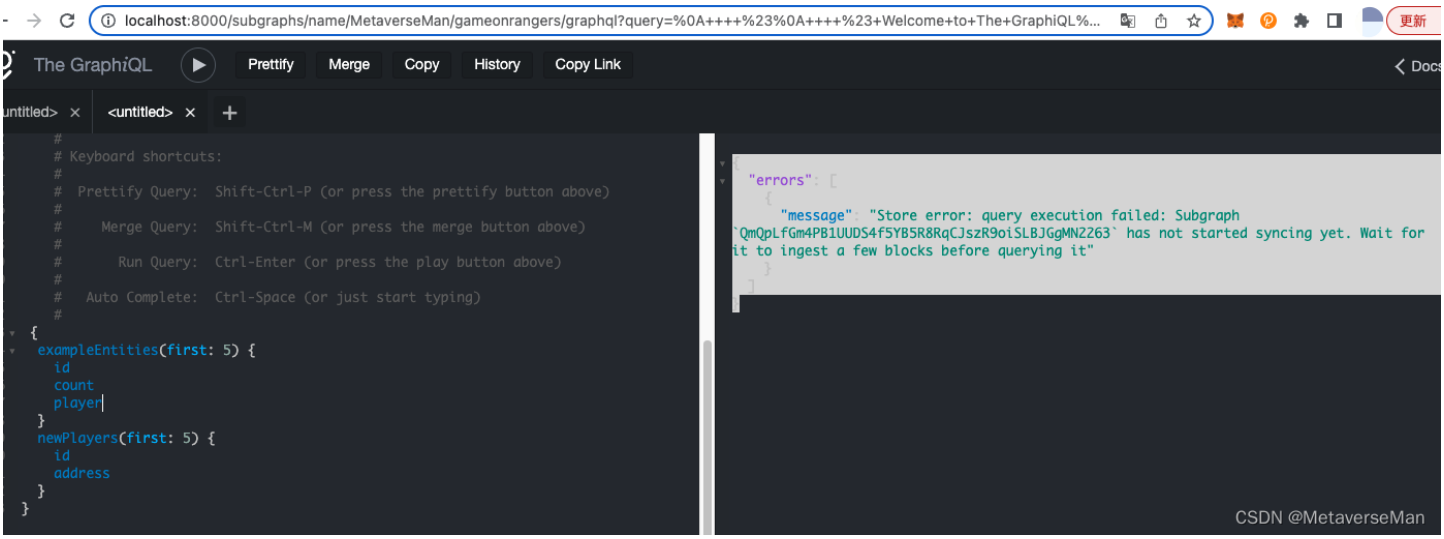
```
? Version Label (e.g. v0.0.1) › (node:98385) ExperimentalWarning: The Fetch API
(Use `node --trace-warnings ...` to show where the warning was created)
✔ Version Label (e.g. v0.0.1) ·
  Skip migration: Bump mapping apiVersion from 0.0.1 to 0.0.2
  Skip migration: Bump mapping apiVersion from 0.0.2 to 0.0.3
  Skip migration: Bump mapping apiVersion from 0.0.3 to 0.0.4
  Skip migration: Bump mapping apiVersion from 0.0.4 to 0.0.5
  Skip migration: Bump mapping apiVersion from 0.0.5 to 0.0.6
  Skip migration: Bump manifest specVersion from 0.0.1 to 0.0.2
  Skip migration: Bump manifest specVersion from 0.0.2 to 0.0.4
✔ Apply migrations
✔ Load subgraph from subgraph.yaml
  Compile data source: Game => build/Game/Game.wasm
✔ Compile subgraph
  Copy schema file build/schema.graphql
  Write subgraph file build/Game/abis/Game.json
  Write subgraph manifest build/subgraph.yaml
✔ Write compiled subgraph to build/
  Add file to IPFS build/schema.graphql
                .. Qmc8mBqzRArjzreqnyBF4PEargATRKuaLesmGm7vbXALQ8
  Add file to IPFS build/Game/abis/Game.json
                .. QmapPy7RyHEGVX7Fpp8ZgjdeeDXGN3JA2xnvJzbV78R2rP
  Add file to IPFS build/Game/Game.wasm
                .. QmdW8vg7vvPJBXnPxx3dQURBdi3Peq3yiKCJSdbYsjdddG
✔ Upload subgraph to IPFS

Build completed: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263

Deployed to http://127.0.0.1:8000/subgraphs/name/MetaverseMan/gameonrangers/grap

Subgraph endpoints:
Queries (HTTP):     http://127.0.0.1:8000/subgraphs/name/MetaverseMan/gameonrang

✨  Done in 121.49s.
```

docker-compose logs -f 日志如下：

```
graph-node_1  | Sep 21 08:13:10.888 INFO Syncing 2 blocks from Ethereum, code: B
graph-node_1  | Sep 21 08:13:12.879 INFO Syncing 2 blocks from Ethereum, code: B
graph-node_1  | Sep 21 08:13:14.007 INFO Resolve schema, link: /ipfs/Qmc8mBqzRAr
graph-node_1  | Sep 21 08:13:14.008 INFO Resolve data source, source_start_block
graph-node_1  | Sep 21 08:13:14.008 INFO Resolve mapping, link: /ipfs/QmdW8vg7vv
graph-node_1  | Sep 21 08:13:14.008 INFO Resolve ABI, link: /ipfs/QmapPy7RyHEGVX
graph-node_1  | Sep 21 08:13:14.013 INFO Set subgraph start block, block: None,
graph-node_1  | Sep 21 08:13:14.013 INFO Graft base, block: None, base: None, sg
```

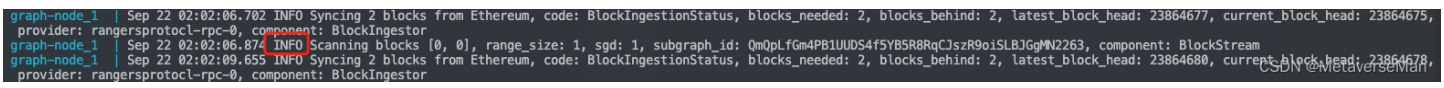
```
graph-node_1  | Sep 21 08:13:14.714 INFO Starting subgraph writer, queue_size: 5
graph-node_1  | Sep 21 08:13:14.732 INFO Resolve subgraph files using IPFS, sgd:
graph-node_1  | Sep 21 08:13:14.732 INFO Resolve schema, link: /ipfs/Qmc8mBqzRAr
graph-node_1  | Sep 21 08:13:14.733 INFO Resolve data source, source_start_block
graph-node_1  | Sep 21 08:13:14.733 INFO Resolve mapping, link: /ipfs/QmdW8vg7vv
graph-node_1  | Sep 21 08:13:14.733 INFO Resolve ABI, link: /ipfs/QmapPy7RyHEGVX
graph-node_1  | Sep 21 08:13:14.734 INFO Successfully resolved subgraph files us
graph-node_1  | Sep 21 08:13:14.734 INFO Data source count at start: 1, sgd: 1,
graph-node_1  | Sep 21 08:13:14.870 INFO Syncing 2 blocks from Ethereum, code: B
graph-node_1  | Sep 21 08:13:15.857 INFO Scanning blocks [0, 0], range_size: 1,
graph-node_1  | Sep 21 08:13:17.478 INFO Syncing 2 blocks from Ethereum, code: B
```

在这Queries (HTTP): http://127.0.0.1:8000/subgraphs/name/MetaverseMan/gameonrangers
去查显示还未同步



等了两个小时还是未同步，看来还是有毛病。。。
另外：你的合约部署的块的高度一定要大于你私有节点同步的起始高度。也就是必须要先部署私有节点然后再部署合约，因为私有节点不会同步你部署之前的区块，可以看这个github issue:subgraph can't fetch history block #3793
其实可以通过docker-compose logs 很容易看到私有节点同步的起始节点和正在同步的进度。

日志里已经开始块同步了 但是查不到



```
{
  newPlayers(first: 5) {
    id
    address
  }
  playGames(first: 5) {
    id
    address
    value
    haswon
  }
}
```

```
{
  "errors": [
    {
      "message": "Store error: query execution failed: Subgraph `QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2Z63` has not started syncing yet. Wait for it to ingest a few blocks before querying it"
    }
  ]
}
```


本来感觉[0,0]不对
但是找到了这个说是对的 https://docs.skale.network/develop/using-graph

12. Verify graph-node terminal

    Expected output in graph-node container:

```
graph-node_1   | Apr 23 20:20:58.952 INFO Scanning blocks [0, 0], range_size:
```

13. Execute graph test

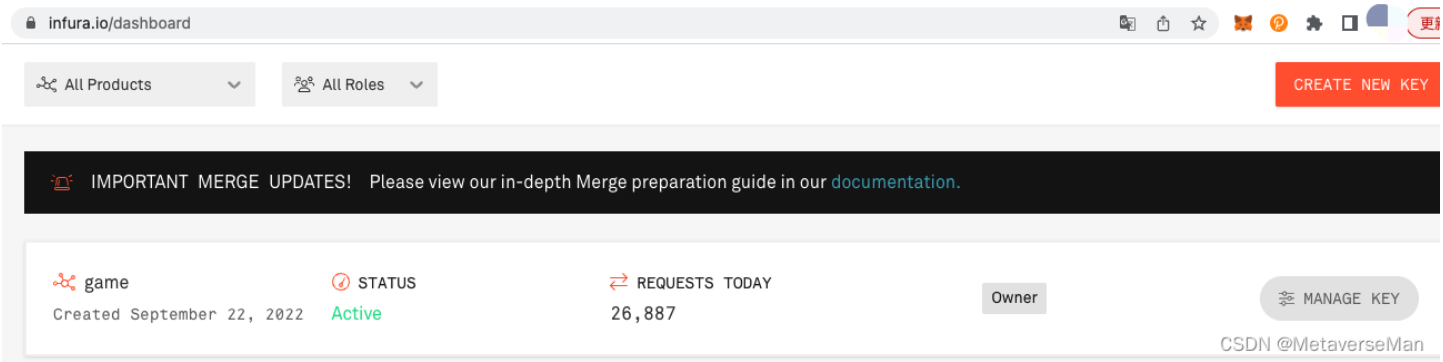
docker-compose logs -f | grep Scanning
查看一下同步的日志

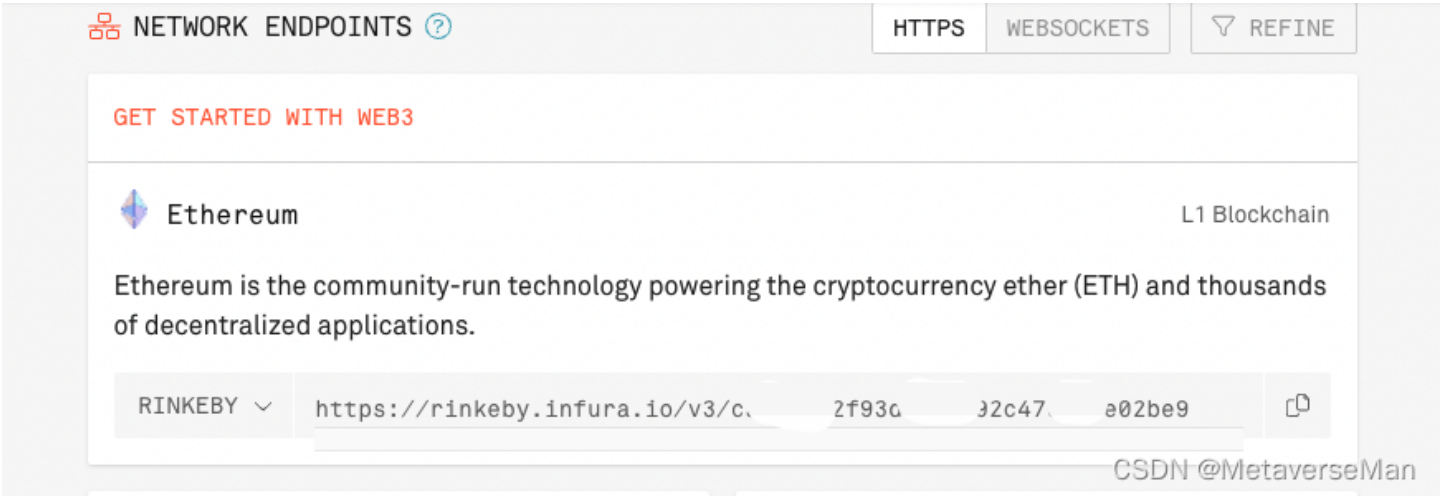
```
graph-node_1   | Sep 21 11:25:23.972 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 11:30:44.966 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 11:36:10.969 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 11:41:42.374 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 11:47:18.373 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 12:08:19.926 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 14:35:19.266 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 14:41:09.477 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 14:47:04.725 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 15:37:28.786 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 15:43:34.028 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 15:49:44.275 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 15:55:59.539 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 16:02:19.750 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 21 16:08:45.013 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 01:02:00.135 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 01:19:56.035 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 01:55:20.529 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 02:02:06.874 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 02:08:58.152 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 02:15:54.400 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 02:22:55.449 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
graph-node_1   | Sep 22 02:30:01.548 INFO Scanning blocks [0, 0], range_size: 1, sgd: 1, subgraph_id: QmQpLfGm4PB1UUDS4f5YB5R8RqCJszR9oiSLBJGgMN2263, component: BlockStream
```


明明有同步，却还是查不到数据

----------分割线----------

成功了, 事实证明不是部署上的错误，是链的RPC节点可能有点问题！而且[0,0]是不对的！应该是增长的。
经过几次略微调整我认为可能不对的点，重新部署之前的rangers链后，还是不对！
于是我决定换条链试试，我把合约部署在了 rinkeby测试网，节点选用infura的，






```
https://rinkeby.infura.io/v3/<your infura api key>
```

然后还是按着以上步骤进行了部署，但是thegraph节点，不用重新部署了，直接

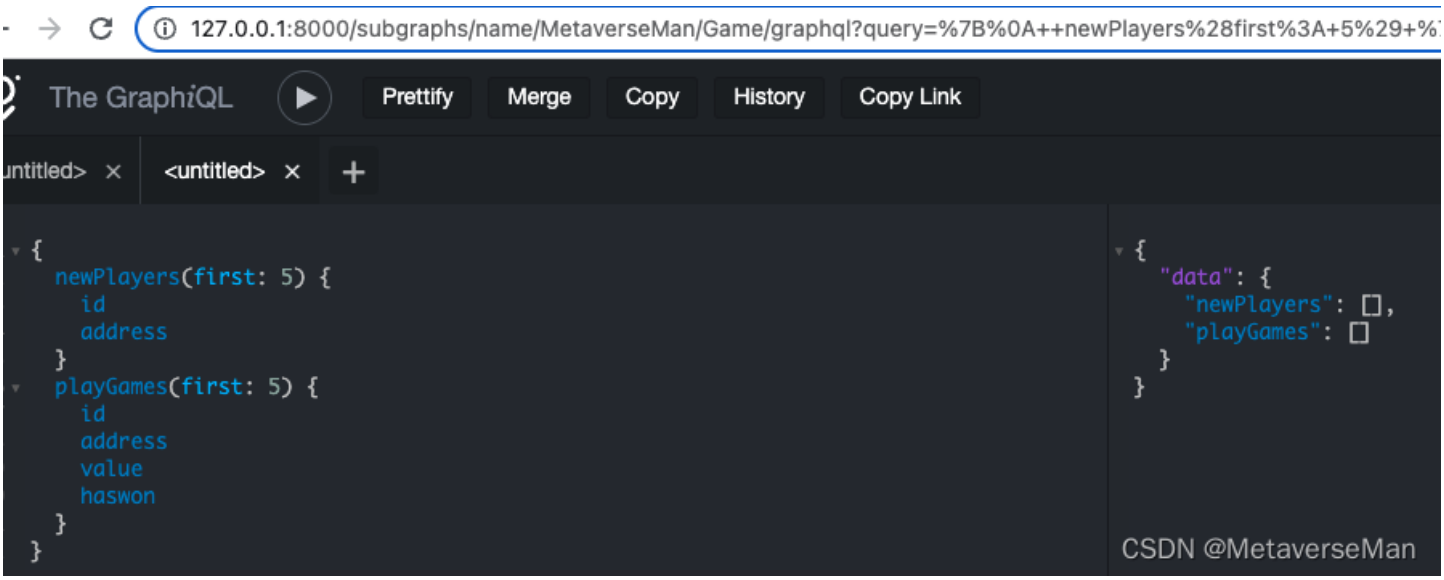
```
docker-compose down
vim docker-compose.yml
(修改这行：ethereum: 'rinkeby:https://rinkeby.infura.io/v3/<your infura api key>'
docker-compose up -d
```

然后参考以上将子图部署到私有节点上。

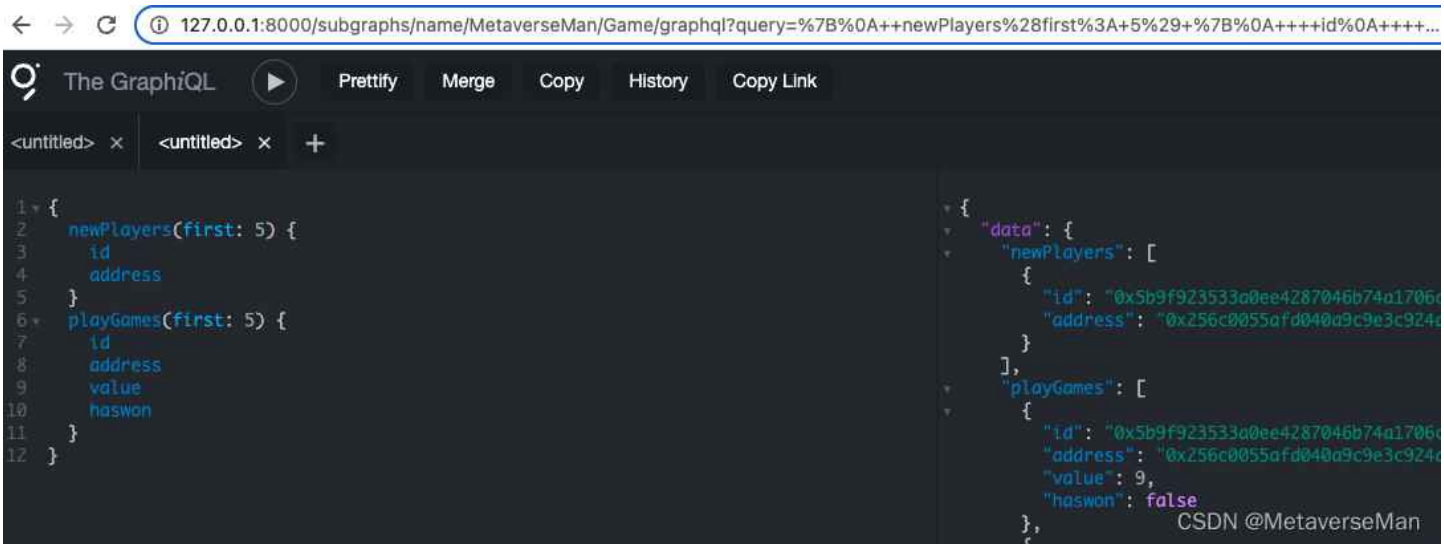
```
graph-node_1  | Sep 22 08:19:00.563 INFO Received subgraph_deploy request, param
graph-node_1  | Sep 22 08:19:05.581 INFO Resolve schema, link: /ipfs/QmdKLtNAeVD
graph-node_1  | Sep 22 08:19:05.582 INFO Resolve data source, source_start_block
graph-node_1  | Sep 22 08:19:05.583 INFO Resolve mapping, link: /ipfs/QmcmqtAfWd
graph-node_1  | Sep 22 08:19:05.583 INFO Resolve ABI, link: /ipfs/QmapPy7RyHEGVX
graph-node_1  | Sep 22 08:19:05.594 INFO Set subgraph start block, block: None,
graph-node_1  | Sep 22 08:19:05.594 INFO Graft base, block: None, base: None, sg
graph-node_1  | Sep 22 08:19:06.255 INFO Starting subgraph writer, queue_size: 5
```

```
 9 graph-node_1  | Sep 22 08:19:06.266 INFO Resolve subgraph files using IPFS, sgd:
10 graph-node_1  | Sep 22 08:19:06.266 INFO Resolve schema, link: /ipfs/QmdKLtNAeVD
11 graph-node_1  | Sep 22 08:19:06.266 INFO Resolve data source, source_start_block
12 graph-node_1  | Sep 22 08:19:06.267 INFO Resolve mapping, link: /ipfs/QmcmqtAfWd
13 graph-node_1  | Sep 22 08:19:06.267 INFO Resolve ABI, link: /ipfs/QmapPy7RyHEGVX
14 graph-node_1  | Sep 22 08:19:06.268 INFO Successfully resolved subgraph files us
15 graph-node_1  | Sep 22 08:19:06.268 INFO Data source count at start: 1, sgd: 1,
16 graph-node_1  | Sep 22 08:19:06.520 INFO Scanning blocks [0, 0], range_size: 1,
17 graph-node_1  | Sep 22 08:19:07.216 INFO Scanning blocks [1, 10], range_size: 10
18 graph-node_1  | Sep 22 08:19:07.915 INFO Scanning blocks [11, 110], range_size:
19 graph-node_1  | Sep 22 08:19:08.605 INFO Scanning blocks [111, 1110], range_size
20 graph-node_1  | Sep 22 08:19:09.310 INFO Scanning blocks [1111, 3110], range_siz
21 graph-node_1  | Sep 22 08:19:10.000 INFO Scanning blocks [3111, 5110], range_siz
22 graph-node_1  | Sep 22 08:19:10.763 INFO Scanning blocks [5111, 7110], range_siz
23 graph-node_1  | Sep 22 08:19:11.463 INFO Scanning blocks [7111, 9110], range_siz
24 graph-node_1  | Sep 22 08:19:12.190 INFO Scanning blocks [9111, 11110], range_si
25 graph-node_1  | Sep 22 08:19:12.900 INFO Scanning blocks [11111, 13110], range_s
26 graph-node_1  | Sep 22 08:19:13.603 INFO Scanning blocks [13111, 15110], range_s
27 graph-node_1  | Sep 22 08:19:14.296 INFO Scanning blocks [15111, 17110], range_s
```

看到扫块的个数是增长的，不是[0,0].....这里也被那篇文章误导了，以为[0,0]是正常状态，其实想想也应该是有增长的。
这个时候再去面板查询就已经不是query execution failed: Subgraph has not started syncing yet了，而是正常的展示，没报错。
现在还没同步完，我的合约在11423609块上，目前看还得同步个个把小时。一会好了查查看。对了，这个scanning是从0块开始扫的，syncing是从你部署的时候的高度开始同步的。



----------分割线----------



同步完了。查询功能正常使用。