## 一、环境配置

#### （1）操作系统

### 1）[下载 VMware Workstation Pro | CN](https://www.vmware.com/cn/products/workstation-pro/workstation-pro-evaluation.html)

### Workstation 16 Pro for Windows

**2）**[下载Ubuntu桌面系统 | Ubuntu](https://cn.ubuntu.com/download/desktop)

Ubuntu LTS 20.04

**3）**[【Ubuntu】Ubuntu 18.04 LTS 更换国内源——解决终端下载速度慢的问题 - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/61228593#:~:text=Ubuntu%E9%85%8D%E7%BD%AE%E7%9A%84%E9%BB%98%E8%AE%A4%E6%BA%90%E5%B9%B6%E4%B8%8D%E6%98%AF%E5%9B%BD%E5%86%85%E7%9A%84%E6%9C%8D%E5%8A%A1%E5%99%A8%EF%BC%8C%E4%B8%8B%E8%BD%BD%E6%9B%B4%E6%96%B0%E8%BD%AF%E4%BB%B6%E9%83%BD%E6%AF%94%E8%BE%83%E6%85%A2%E3%80%82 %E9%A6%96%E5%85%88%E5%A4%87%E4%BB%BD%E6%BA%90%E5%88%97%E8%A1%A8%E6%96%87%E4%BB%B6 sources.list %EF%BC%9A,2. %E6%89%93%E5%BC%80sources.list%E6%96%87%E4%BB%B6%E4%BF%AE%E6%94%B9 %E9%80%89%E6%8B%A9%E5%90%88%E9%80%82%E7%9A%84%E6%BA%90%EF%BC%8C%E6%9B%BF%E6%8D%A2%E5%8E%9F%E6%96%87%E4%BB%B6%E7%9A%84%E5%86%85%E5%AE%B9%EF%BC%8C%E4%BF%9D%E5%AD%98%E7%BC%96%E8%BE%91%E5%A5%BD%E7%9A%84%E6%96%87%E4%BB%B6, %E4%BB%A5%E9%98%BF%E9%87%8C%E4%BA%91%E6%9B%B4%E6%96%B0%E6%9C%8D%E5%8A%A1%E5%99%A8%E4%B8%BA%E4%BE%8B%EF%BC%88%E5%8F%AF%E4%BB%A5%E5%88%86%E5%88%AB%E6%B5%8B%E8%AF%95%E9%98%BF%E9%87%8C%E4%BA%91%E3%80%81%E6%B8%85%E5%8D%8E%E3%80%81%E4%B8%AD%E7%A7%91%E5%A4%A7%E3%80%81163%E6%BA%90%E7%9A%84%E9%80%9F%E5%BA%A6%EF%BC%8C%E9%80%89%E6%8B%A9%E6%9C%80%E5%BF%AB%E7%9A%84%EF%BC%89%EF%BC%9A)

①备份源列表文件sources.list

|  |
| --- |
| sudo cp /etc/apt/sources.list /etc/apt/sources.list\_backup |

②打开sources.list文件，在最前边添加阿里云镜像源

|  |
| --- |
| # 阿里源  deb http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse  deb http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse  deb http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse  deb http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse  deb http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse  deb-src http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse  deb-src http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse  deb-src http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse  deb-src http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse  deb-src http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse |

③刷新列表

|  |
| --- |
| sudo apt-get update  sudo apt-get upgrade  sudo apt-get install build-essential |

#### （2）安装依赖工具

1）安装git

|  |
| --- |
| sudo apt install git |

1. 安装cURL

|  |
| --- |
| sudo apt install curl |

1. 安装Docker

|  |
| --- |
| sudo apt install docker.io |

查看docker安装成功

|  |
| --- |
| docker --version |

设置成非 root 用户也能执行 docker，需要将普通用户加入 docker 组：

|  |
| --- |
| docker --version |



|  |
| --- |
| sudo usermod -aG docker 你的用户名 （重启生效） |

4）安装docker-compose

|  |
| --- |
| sudo apt install docker-compose |

查看docker-compose安装成功

|  |
| --- |
| docker-compose --version |



允许其他用户执行 compose 相关命令：

|  |
| --- |
| sudo chmod +x /usr/share/doc/docker-compose |

#### （3）安装Goland

1）下载Goland。运行前需配置go环境，建议使用go1.15.2版本，(与IPFS兼容)

**方法一：**wget下载命令

|  |
| --- |
| wget https://dl.google.com/go/go1.15.2.linux-amd64.tar.gz |

**方法二：**如果网络不行，上述命令执行失败，可以直接从 <https://studygolang.com/dl> 下载相应的 Golang 版本压缩包，拷贝到Ubuntu的/usr/local目录下。

2）解压文件

|  |
| --- |
| sudo tar -zxvf go1.15.2.linux-amd64.tar.gz -C /usr/local/ |

1. 配置环境变量。(本文采用了方案二)

**方案一：**如果想让系统所有用户使用 Golang，则编辑 /etc/profile 文件；如果只

|  |
| --- |
| sudo gedit /etc/profile |

在 profile 文件最后添加如下内容：

|  |
| --- |
| export GOROOT=/usr/local/go  export GOPATH=$HOME/go  export PATH=$PATH:$GOROOT/bin:$GOPATH/bin |

使用 source 命令，使刚刚添加的配置信息生效：

|  |
| --- |
| source /etc/profile |

go version 命令验证是否安装成功

|  |
| --- |
| go version |



**方案二：**是想让当前用户使用 Golang，则编辑当前用户 $HOME 目录下 .bashrc 或 .profile 文件。

|  |
| --- |
| gedit ~/.bashrc |

编辑 ~/.bashrc文件，在文件末尾添加以下代码

|  |
| --- |
| export GOROOT=/usr/local/go  export GOPATH=&HOME/go  export GO111MODULE=on #这两句为go get加速  export GOPROXY=https://goproxy.cn #这两句为go get加速  export PATH=$PATH:$GOROOT/bin:$GOPATH/bin |

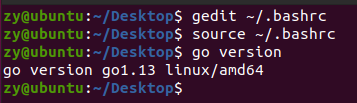
完成后关闭并保存，在命令行输入

|  |
| --- |
| source ~/.bashrc |

使配置文件生效。

go version 命令验证是否安装成功

|  |
| --- |
| go version |



4） 卸载旧版本 Golang 的命令。若 Ubuntu 中已经有 Golang，则用如下命令卸载旧版本：

|  |
| --- |
| $ su -  # apt-get remove golang-go --purge && apt-get autoremove --purge && apt-get clean |

#### 安装Fabric

1）创建一个空目录并进入该目录：

|  |
| --- |
| $ mkdir -p ~/go/src/github.com/hyperledger  $ cd ~/go/src/github.com/hyperledger |

1. 拉取 fabric 的源码，通过以下命令拉取：

|  |
| --- |
| $ git clone "https://github.com/hyperledger/fabric.git" |

1. 查看并切换当前分支

|  |
| --- |
| $ cd ./fabric  $ git checkout release-1.4  $ cd scripts/  $ sudo ./bootstrap.sh -b #不加-b可能受到网络限制 |

如果上一步操作下载二进制文件太慢或者没速度，可以直接对源码进行编译,执行以下命令(前提是以上相关路径配置没有错误)：

|  |
| --- |
| #首先进入fabric文件夹  $ cd ~/go/src/github.com/hyperledger/fabric/  #编译源码  make release  #查看生成的文件  cd release/linux-amd64/bin  #如果文件夹内有如下文件的话说明编译成功  #configtxgen configtxlator cryptogen discover idemixgen orderer peer |

回到fabric目录下对fabric进行编译

|  |
| --- |
| cd ..  make release |

将fabric-samples复制到hyperledger目录下

|  |
| --- |
| cd ..  cp -rf ./fabric-samples ../../ |

回到 fabric 目录下 并把 linux-amd64 下面的 bin 目录复制到 fabric 目录下和 fabric-samples 目录下

|  |
| --- |
| cd ..  cp -rf ./release/linux-amd64/bin ./  cp -rf ./release/linux-amd64/bin ../fabric-samples/ |

打开配置文件

|  |
| --- |
| sudo gedit ~/.bashrc |

将最后一行的

|  |
| --- |
| export PATH=$PATH:$GOROOT/bin:$GOPATH/bin |

改成以下内容，保存并退出然后重启

|  |
| --- |
| export PATH=$PATH:$GOROOT/bin:$GOPATH/bin:/home/$USER/go/src/github.com/hyperledger/fabric/bin:$GOPATH |

使配置文件生效

|  |
| --- |
| source ~/.bashrc |

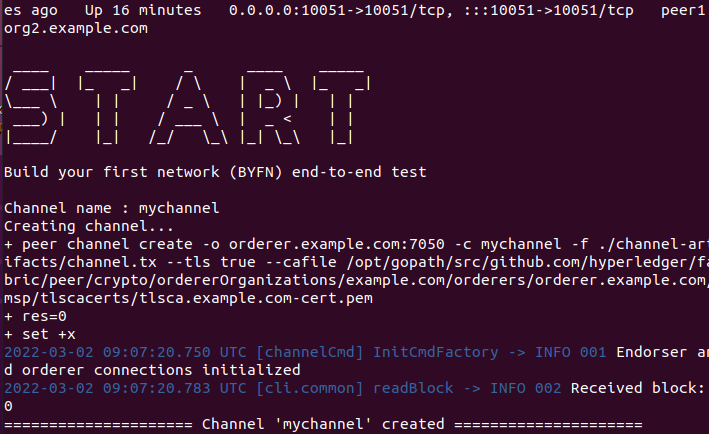
1. 网络测试

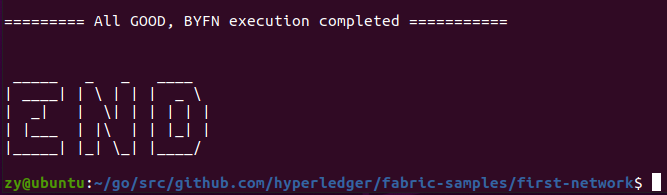
|  |
| --- |
| cd ~/go/src/github.com/hyperledger/fabric-samples/first-network |

sudo ./byfn.sh -m generate

|  |
| --- |
| sudo ./byfn.sh -m generate |

|  |
| --- |
| sudo ./byfn.sh -m up |





最终出现以上字样后，表示fabric安装成功。

输入

|  |
| --- |
| ./byfn.sh down |

关闭服务并清理环境。（重要）

#### 安装IPFS

1. 安装go-ipfs

下载go-ipfs 首先是资源下载，比较推荐用稳定点的版本，0.4.xx即可，使用配置比较方便，不要求go版本。新版本的ipfs需要1.14的go，但是有些环境需要1.4的go，所以0.4.xx的版本就可以了

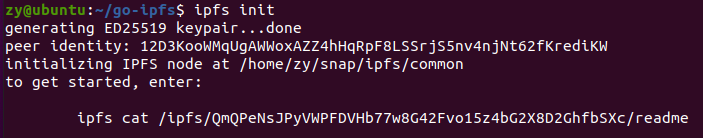
|  |
| --- |
| git clone https://github.com/ipfs/go-ipfs.git |

进入go-ipfs目录，输入

|  |
| --- |
| sudo snap install ipfs |

初始化ipfs

|  |
| --- |
| sudo snap install ipfs |

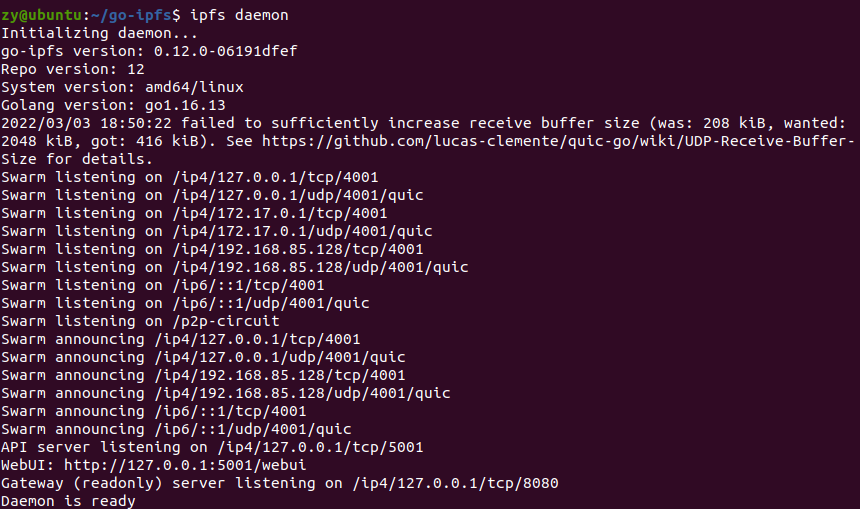


完成后输入

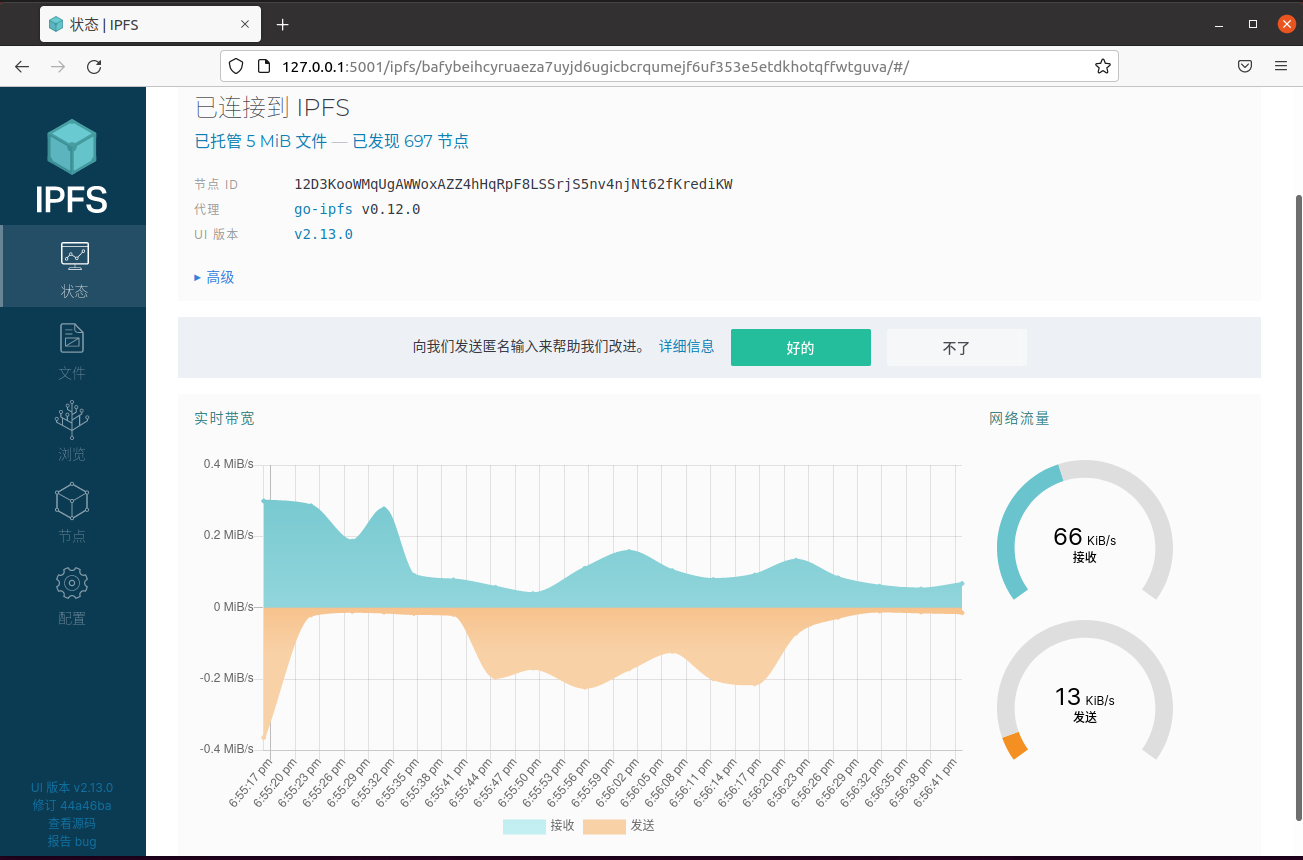
|  |
| --- |
| ipfs init |

完成后再输入

|  |
| --- |
| ipfs daemon |



完成后打开浏览器，在地址栏输入127.0.0.1：5001/webui

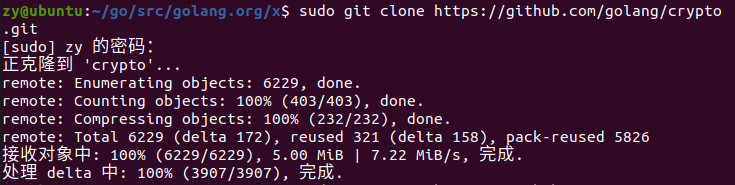


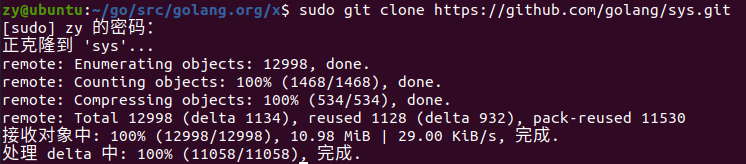
出现以上界面，表明go-ipfs安装成功

1. 安装go-ipfs-api

打开控制台，依次输入

|  |
| --- |
| mkdir -p $GOPATH/src/golang.org/x  cd $GOPATH/src/golang.org/x  sudo git clone <https://github.com/golang/crypto.git>  sudo git clone https://github.com/golang/sys.git |





出现以上界面后，再输入

|  |
| --- |
| go get -u github.com/ipfs/go-ipfs-api |

检查是否安装成功。新建一个test\_go\_ipfs.go文件，

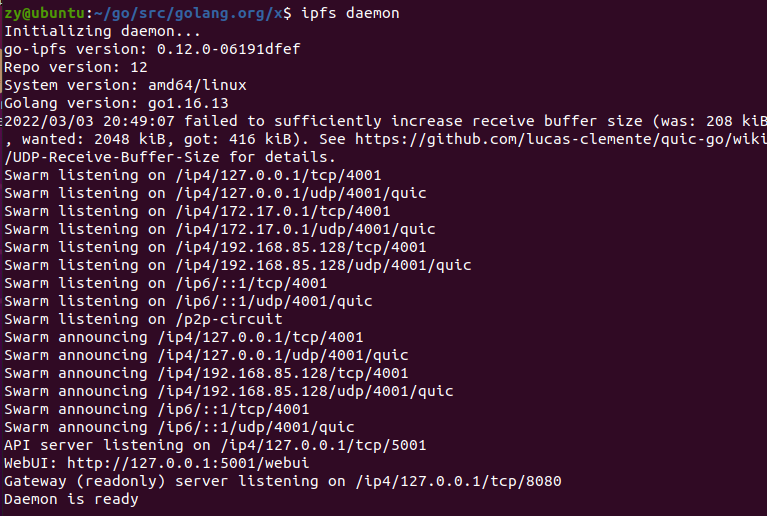
|  |
| --- |
| touch test\_go\_ipfs.go  gedit test\_gp\_ipfs.go |

输入以下代码

|  |
| --- |
| package main  import (  "fmt"  "strings"  "os"  shell "github.com/ipfs/go-ipfs-api"  )  func main() {  sh := shell.NewShell("localhost:5001")  cid, err := sh.Add(strings.NewReader("hello world!"))  if err != nil {  fmt.Fprintf(os.Stderr, "error: %s", err)  os.Exit(1)  }  fmt.Printf("added %s\n", cid)  } |

保存后，打开控制台，输入命令开启ipfs服务.

|  |
| --- |
| ipfs daemon |



完成后再打开一个新的控制台，进入test\_go\_ipfs.go所在路径，输入

|  |
| --- |
| cd ~/go/src/golang.org/x/  go run test\_go\_ipfs.go |



出现以上界面，表示安装成功。

#### 安装Beego框架

1）安装

在控制台输入以下两条命令

|  |
| --- |
| ~~go get github.com/astaxie/beego~~  ~~go get github.com/beego/bee~~  go get github.com/beego/beego/v2  go get github.com/beego/bee/v2 |

安装完成后，在控制台输入

|  |
| --- |
| bee version |

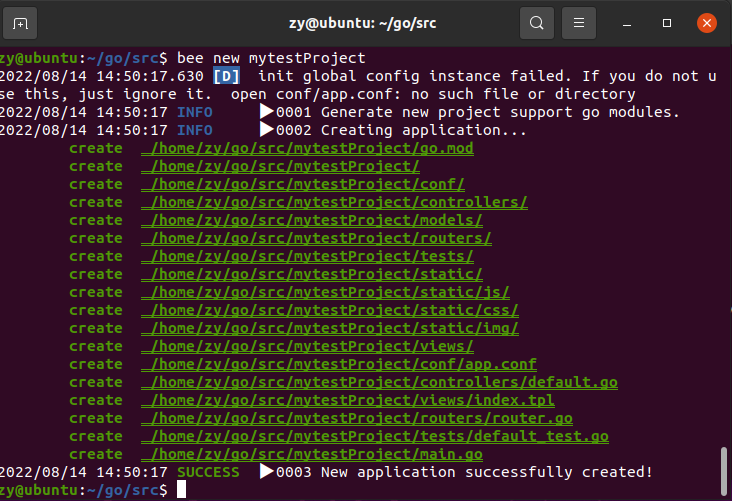


如果出现以上字样，表明安装成功。

2）检查是否安装成功

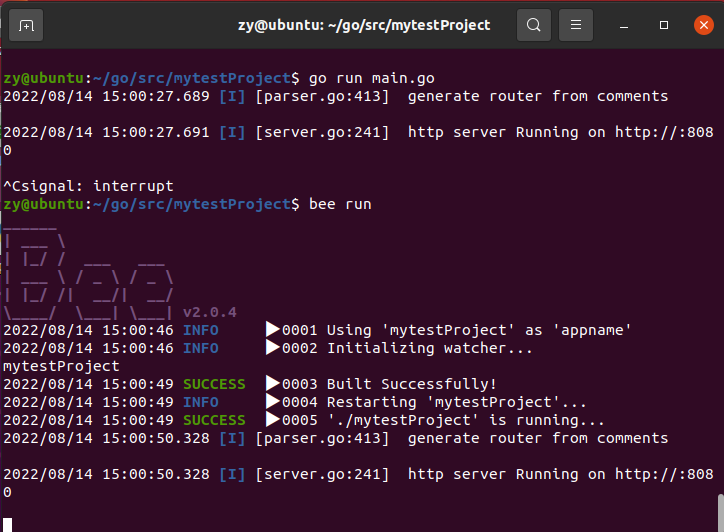
在 ~/go/src 目录下，在控制台输入

|  |
| --- |
| bee new mytestProject |

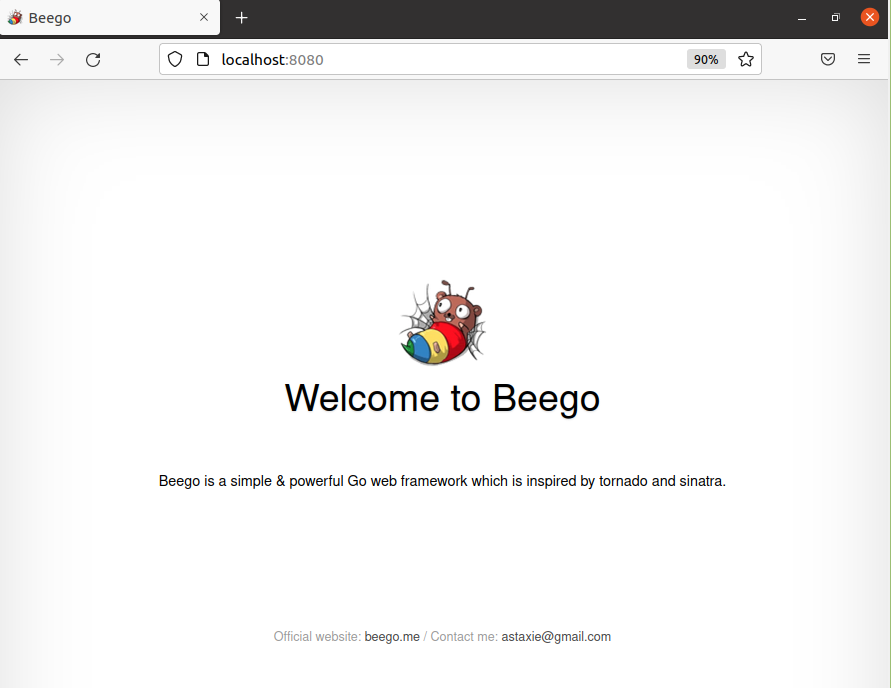


进入mytestProject目录之后输入

|  |
| --- |
| cd mytestProject  bee run |



打开浏览器访问127.0.0.1:8080



出现以上界面，表示beego安装成功。

#### 安装MySQL

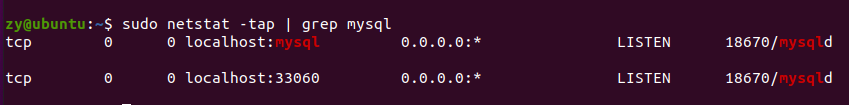
打开控制台，输入以下命令（安装过程中root用户密码设置为123456）

|  |
| --- |
| sudo apt install mysql-server |

安装完成后，在控制台输入

|  |
| --- |
| sudo apt install net-tools  netstat -tap | grep mysql |

显示mysql的端口处于监听状态表示安装成功。

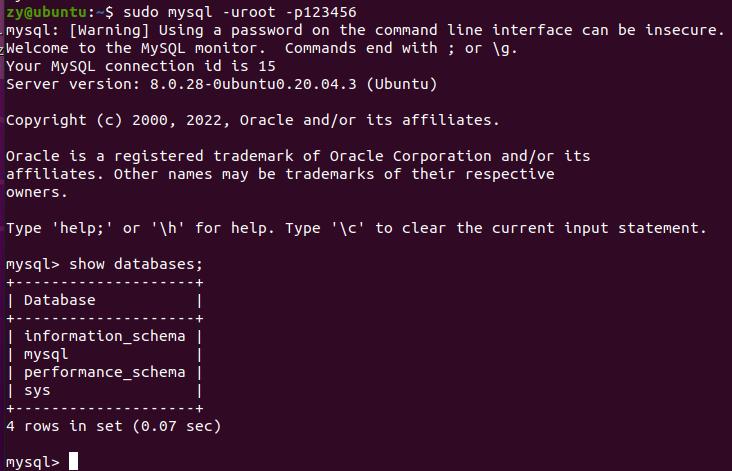


输入

|  |
| --- |
| mysql -uroot -p123456 |

进入mysql，输入

|  |
| --- |
| show databases; |



出现以上界面，表示mysql安装成功。

退出mysql操作

|  |
| --- |
| quit |

## 二、联盟链环境部署

#### （1）源码配置

将radi文件夹放置在非$GOPATH的路径下（路径中不要出现中文）,使用pwd命令查看自己电脑上radi文件夹的绝对路径如下图。



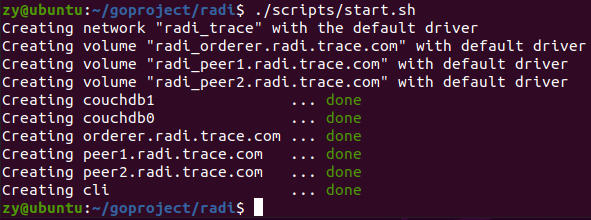
①将radi/scripts文件夹下start.sh和radiInit.sh脚本的export ProjectDir=/home/zy/goproject/radi替换为自己的radi文件夹的绝对路径；

②对radiProvenance文件夹下的scripts/webInit.sh脚本进行同样的路径替换。

返回radi目录下，输入

|  |
| --- |
| ./scripts/start.sh |

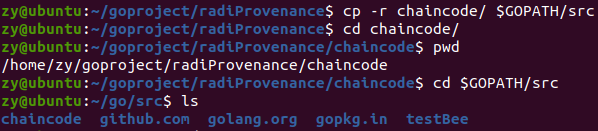
运行start.sh脚本，搭建联盟链测试环境（一组织、两peer节点、节点启用couchDB作为状态账本、一orderer节点），成功截图如下



#### （2）智能合约部署至peer节点

从radiProvenance文件夹下将chaincode文件夹拷贝至$GOPATH/src目录下

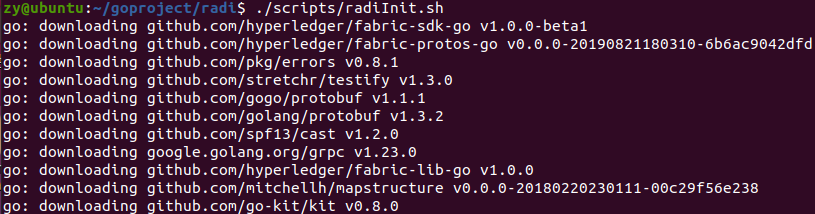
|  |
| --- |
| cp -r /home/zy/goproject/radiProvenance/chaincode $GOPATH/src |



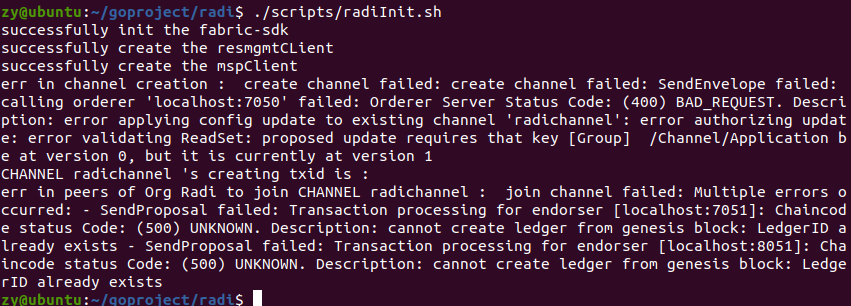
返回radi文件夹下，在radi文件夹下输入

|  |
| --- |
| ./scripts/radiInit.sh |

运行radiInit.sh脚本，如下图所示（此处首次运行需要连网，运行脚本时会下载一些依赖库）。



再次执行上诉命令，智能合约部署成功效果如图所示：

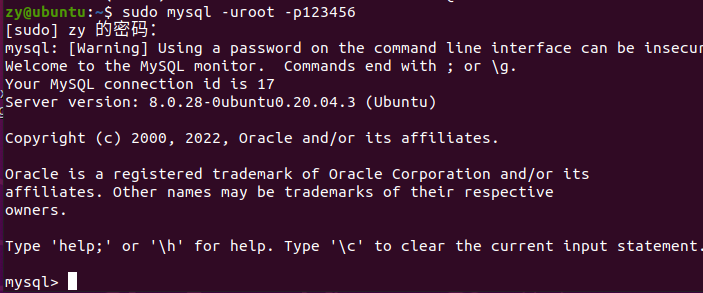


#### （3）创建Mysql数据库

在任意位置打开一个shell，输入

|  |
| --- |
| sudo mysql -uroot -p123456 |

进入mysql数据库，如下图所示。



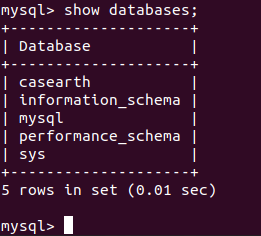
创建数据库casearth，如下图所示

|  |
| --- |
| create database casearth; |

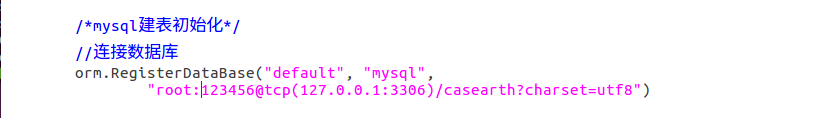


查看

|  |
| --- |
| show databases; |



|  |
| --- |
| [注]：这里需要mysql数据库用户名为root，密码为123456，创建数据库名为casearth。因为Web服务会连接此数据库，若自己设置的用户名密码或数据库名和如上所述不一致，则需要去radiProvenance/models/model.go文件夹中修改对应位置的代码使用自己的用户名、密码和数据库名，需要修改的地方如图所示 |

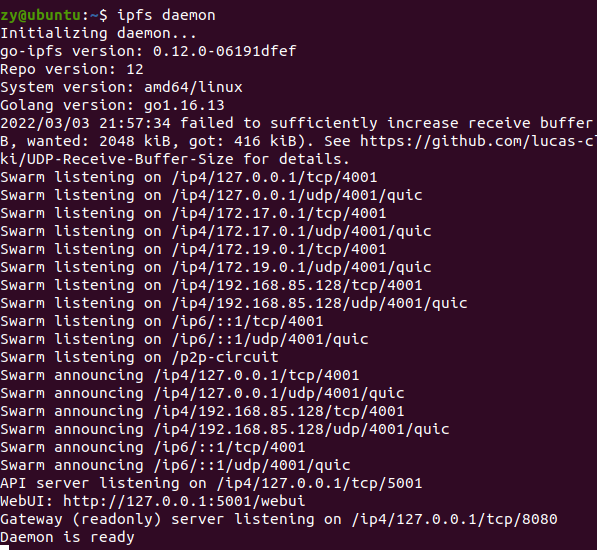


#### （4）启动IPFS文件系统的服务

打开任意的终端，输入

|  |
| --- |
| ipfs init #可省略  ipfs daemon |

启动ipfs文件系统后如图所示（命令可能需要加sudo）



#### （5）启动Web服务

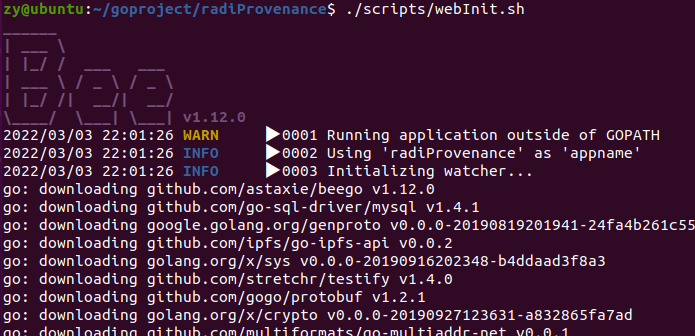
Shell cd进入radiProvenance文件夹

|  |
| --- |
| cd /home/zy/goproject/radiProvenance |

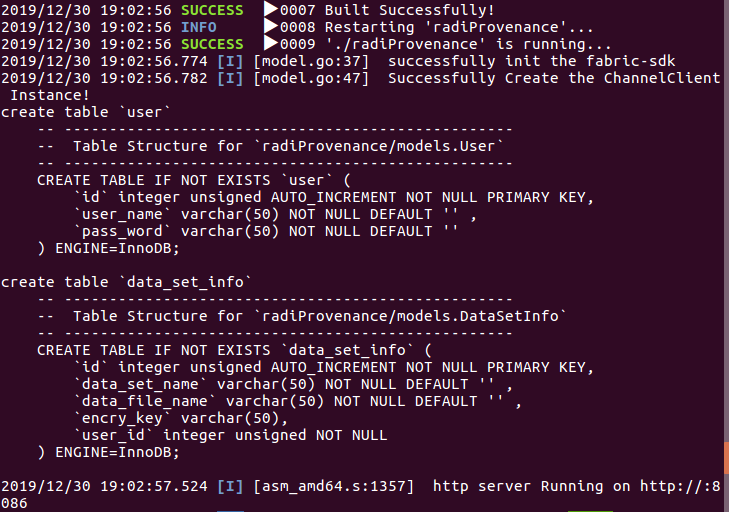
，在此文件夹路径下输入

|  |
| --- |
| ./scripts/webInit.sh |

启动web服务（此处首次运行需要连网，下载一部分依赖库，如下图所示）



Web服务成功启动效果如下图所示。



此时，Web服务运行在本机的8086端口,在浏览器访问127.0.0.1:8086/register

进入用户注册界面如下图所示，此时系统部署成功！（此时shell后台可以看web应用的日志信息）

