安装docker

sudo yum –y install docker

目前本机版本是Docker version 18.03.1-ce, build 9ee9f40

备注：docker基本上要用新版，不要低于docker-ce 17.0. 在centos 6.9上默认库的docker会比较旧，此时需要用官方库的repository来更新docker。

让当前普通用户有权限执行docker命令

在安装docker的时候默认会创建docker用户组。不在该用户组的用户没有执行docker命令的权限

应用用户加入docker用户组命令

sudo usermod –aG docker <用户名>

重启docker服务后生效

sudo systemctl restart docker

docker镜像应用加速器

编辑/etc/docker/daemon.json , 写入下列内容

{

"registry-mirrors": ["https://registry.docker-cn.com"]

}

需要重启docker服务。重启后如果docker pull命令显著变快即为生效。

镜像

hyperledger/fabric-baseos:x86\_64-0.4.6 必须

hyperledger/fabric-baseimage:x86\_64-0.4.6 必须

dashbase/kafka-manager-docker:latest 可选

下载镜像

docker pull hyperledger/fabric-baseos:x86\_64-0.4.6

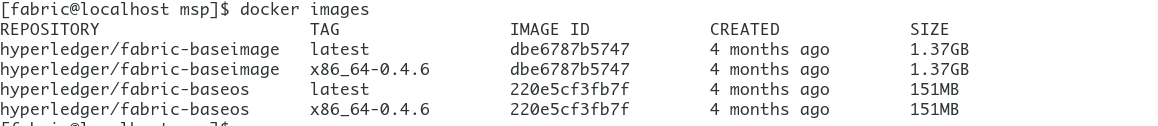
docker tag hyperledger/fabric-baseos:x86\_64-0.4.6 hyperledger/fabric-baseos:latest

docker pull hyperledger/fabric-baseimage:x86\_64-0.4.6

docker tag hyperledger/fabric-baseimage:x86\_64-0.4.6 hyperledger/fabric-baseimage:latest

结果如图：





安装golang环境

1. 安装golang

打开<https://golang.org/dl>， （需要翻墙） 下载go 1.9 版本的最新tar.gz, 当前是go1.9.7.linux-amd64.tar.gz

在此需要了解，fabric 1.1 至少需要golang 1.9

fabric-ca 1.1 至少需要golang 1.9.2

再次我们一般用go 1.9的最新版，截止今天是1.9.7来编译

是否可以用go 1.10最新版？一般可以，需要掌握golang版本升级的更新点

1. 安装目录

一般将go解压在/opt/go，然后新建/opt/gopath

在/opt/gopath下新建/opt/gopath/src, /opt/gopath/bin, /opt/gopath/pkg三个目录

1. 设置环境变量，可以编辑/etc/profile或者~/.bash\_profile

GOROOT=/opt/go

GOPATH=/opt/gopath

GOBIN=/opt/gopath/bin

PATH=$GOROOT/bin:$GOPATH/bin: $PATH

GOROOT,GOBIN,GOPATH均为Golang需要的环境变量，具体含义可以参考Golang文档

重新login或者用source命令使其生效

设置完毕后在命令行执行go version可以看到下列结果



下载附属依赖库

1. 下载golang下的库

golang.org下的包是必须的

由于golang.org被墙，所以golang无法直接拉取golang.org下的包

go get默认使用http通信。若要使用代理的话也可设置http\_proxy/https\_proxy

一般采取变通方法

golang.org下的包在github上有镜像

地址为github.com/golang下

我们通过git clone/pull方法来获取该包

mkdir -p /opt/gopath/src/golang.org/x

cd /opt/gopath/src/golang.org/x

git clone https://github.com/golang/crypto

git clone https://github.com/golang/image

git clone https://github.com/golang/lint

git clone https://github.com/golang/net

git clone https://github.com/golang/oauth2

git clone https://github.com/golang/perf

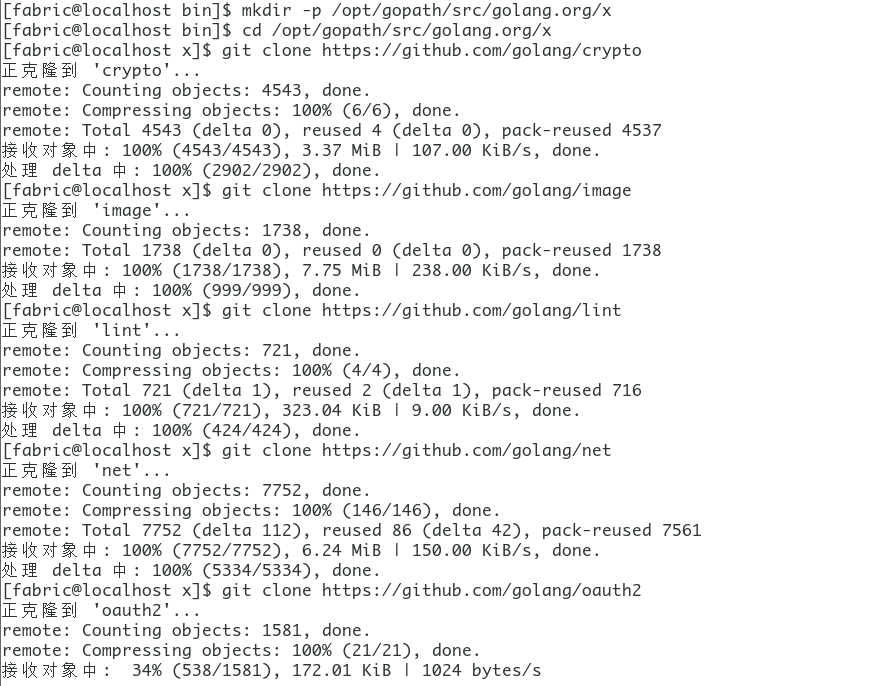
git clone https://github.com/golang/sys

git clone https://github.com/golang/term

git clone https://github.com/golang/text

git clone https://github.com/golang/time

git clone https://github.com/golang/tools





1. 下载github下的依赖库

由于github没有被墙，可以通过go get –u 拉取github下的代码

命令如下

（其中-u代表拉取最新代码）

go get -u github.com/axw/gocov/...

go get -u github.com/AlekSi/gocov-xml

go get -u github.com/kardianos/govendor

go get -u github.com/golang/lint/golint

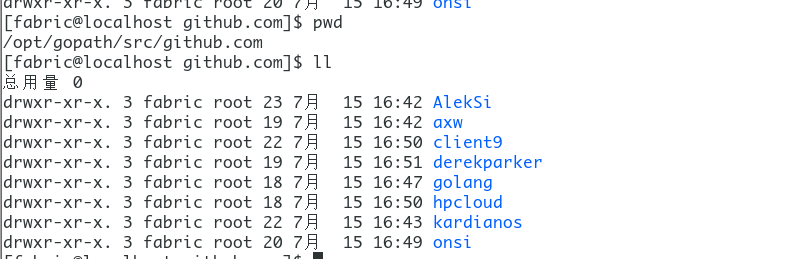
go get -u github.com/onsi/ginkgo/ginkgo

go get -u github.com/client9/misspell/cmd/misspell

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

备注：上述代码拉取下来之后，再进行更新也可以用go get –u 命令

go get –u 本质上通过git pull拉取代码库后进行编译安装到$GOBIN下



1. 下载克隆fabric 代码

由于对fabric 1.1的Makefile进行了改造，我维护了一个自己的分支。

可以从该分支拉取代码（下文会具体讲解改造内容）

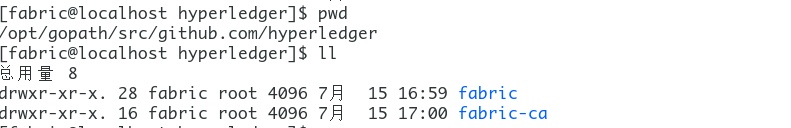
mkdir -p /opt/gopath/src/github.com/hyperledger

cd /opt/gopath/src/github.com/hyperledger

git clone https://gitee.com/heye/fabric -b smartheye/v1.1.0

结果

在/opt/gopath/src/github.com/hyperledger下有下列代码



1. 下载以太坊代码

开发智能合约的时候拉取的是1.7.5分支，目前最新到了1.8.12分支。由于我们用到的核心功能没有改动，实际上可以拉去1.8.12分支的代码。

mkdir -p /opt/gopath/src/github.com/ethereum

cd /opt/gopath/src/github.com/ethereum

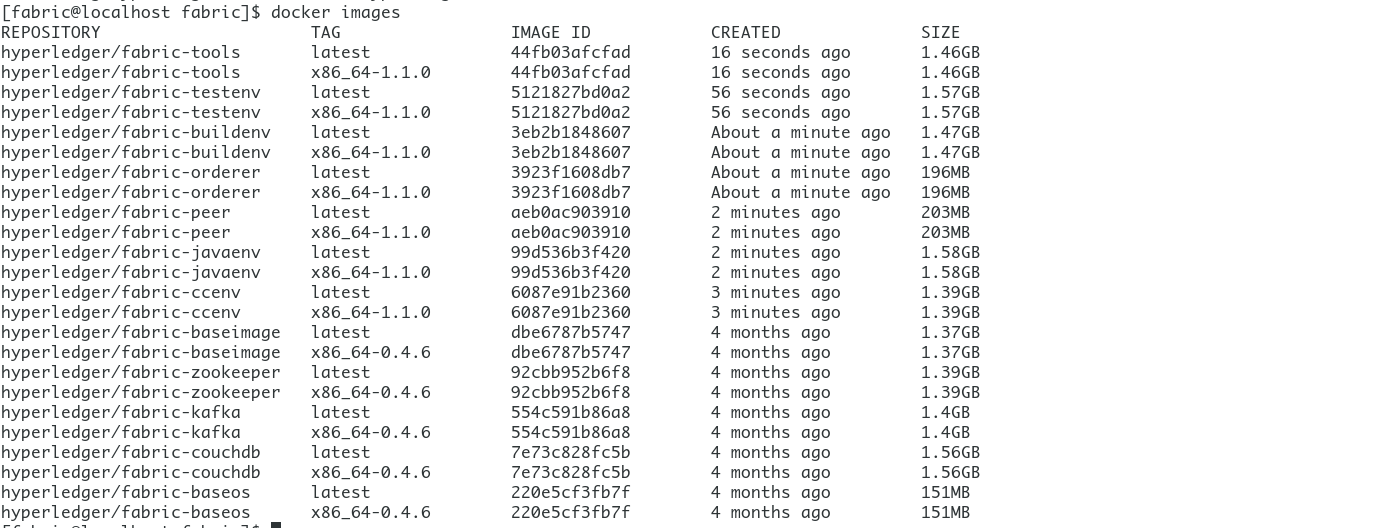
git clone https://github.com/ethereum/go-ethereum -b v1.8.12

编译

cd /opt/gopath/src/github.com/hyperledger/fabric

make docker

其中编译Java的ccenv会比较慢，需要等待，或者跳过java的ccenv编译也可。



为ccenv增加以太坊支持

由于智能合约用到了以太坊的代码，所以需要为ccenv添加以太坊的代码让智能合约在各个节点通过编译

备注：为何不能编译好智能合约发布到各个节点？因为实际上安装智能合约是把代码发布到各个节点，在初始化智能合约的时候各个节点自行编译，编译时使用ccenv作为智能合约。

为何不能添加在智能合约的govendor里？因为fabric在发布智能合约的时候做了过滤，没有go文件的目录会自动被忽略，由此libsecp256k1会被忽略，没有放入智能合约的signpackage导致无法编译。

步骤

在/opt/gopath/src/github.com/ethereum下建立一个Dockerfile

内容为

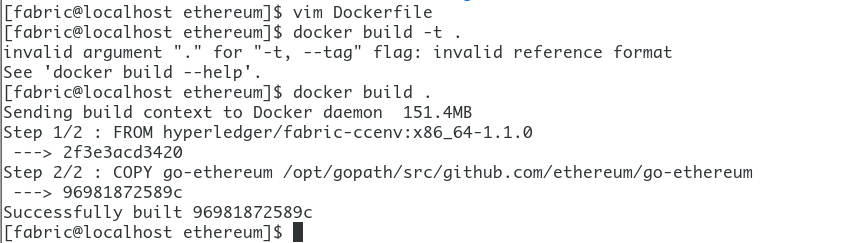
FROM hyperledger/fabric-ccenv:x86\_64-1.1.0

COPY go-ethereum /opt/gopath/src/github.com/ethereum/go-ethereum

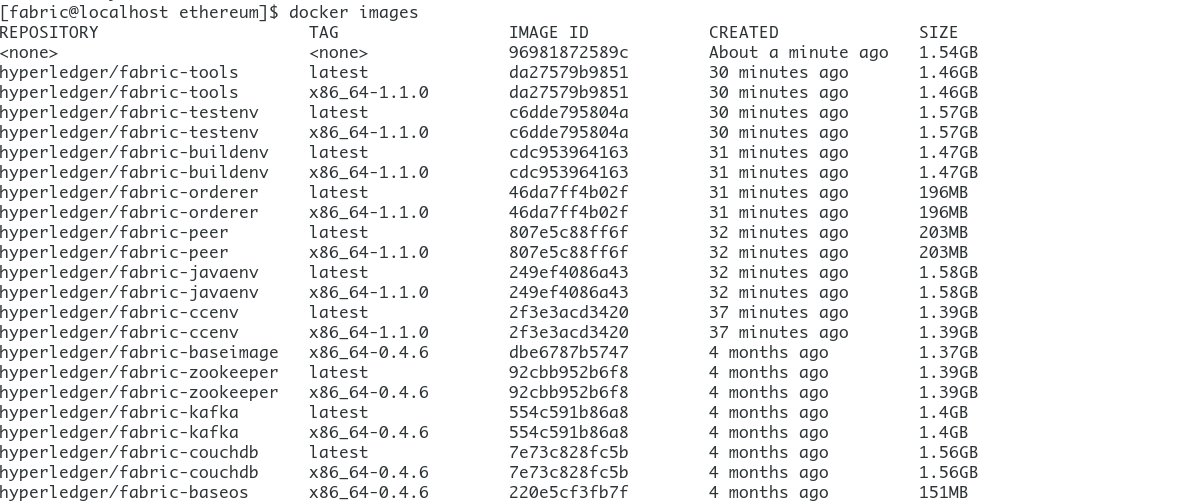
然后在/opt/gopath/src/github.com/ethereum目录下打下列命令

docker build .

注意不要忽略最后一个点，少了该点会无法编译



结果看到最后有一个镜像被编译出来，hashcode为96981872589c



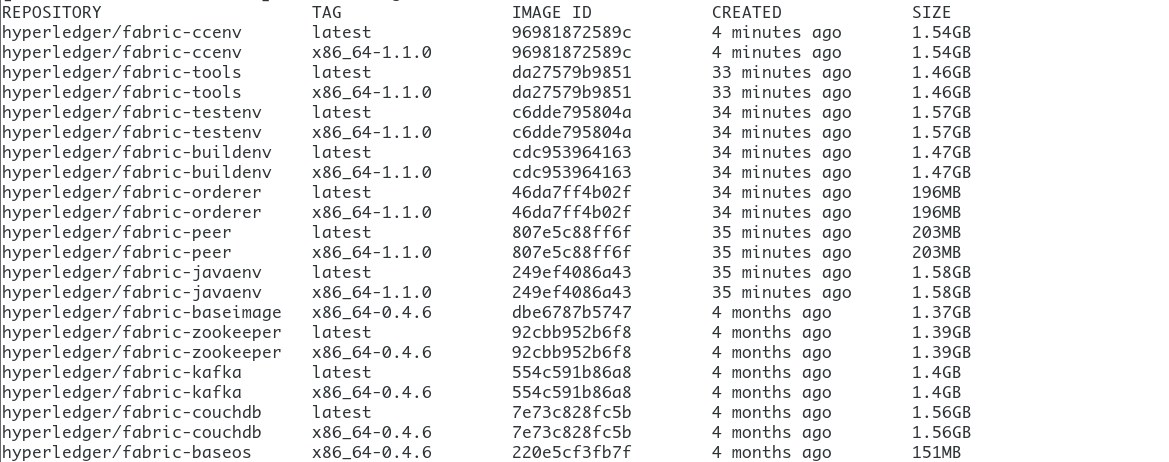
第一个tag 为<non> ImageID为96981872589c的镜像即为编译好的镜像。在此我们为它打上ccenv的tag。由于每次编译出来的镜像的HashCode都不一样，所以ImageID会在每个环境不一样，这是正常情况。

打标签

docker tag 969 hyperledger/fabric-ccenv:x86\_64-1.1.0

docker tag 969 hyperledger/fabric-ccenv:latest

执行docker images，可以看到fabric-ccenv:latest和fabric-ccenv:x86\_64-1.1.0已经换为新容器



编译fabric ca

拉取代码

mkdir -p /opt/gopath/src/github.com/hyperledger

cd /opt/gopath/src/github.com/hyperledger

git clone https://gitee.com/heye/fabric-ca -b smartheye/v1.1.0

编译

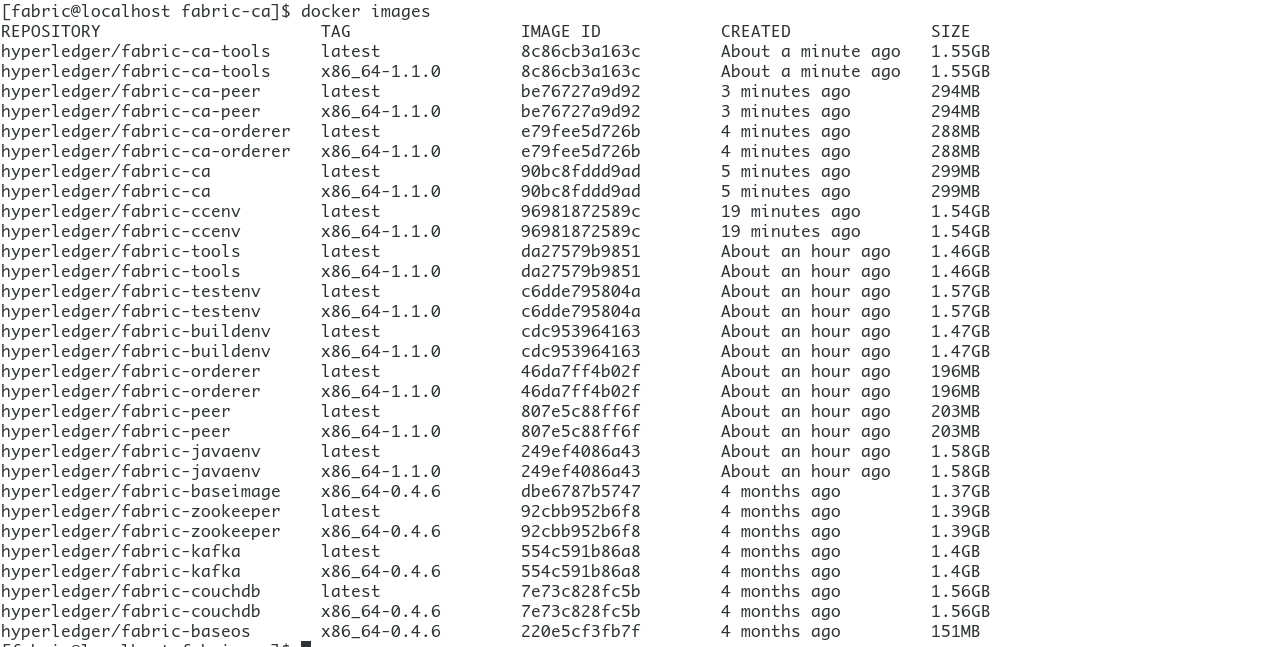
cd /opt/gopath/src/github.com/hyperledger/fabric-ca

make fabric-ca-server

make fabric-ca-client

make docker

然后会编译出fabric-ca的docker



其中只有hyperledger/fabric-ca是有用的。hyperledger/fabric-ca-tools hyperledger/fabric-ca-peer和hyperledger/fabric-ca-orderer目前没有用。

备注

fabric-ca编译报错，ltdl.h: no such file

在编译pkcs11时需要libltdl库

解决办法

【CentOS】

sudo yum –y install libtool-ltdl-devel

【Ubuntu】

sudo apt install libltdl3-dev

常见问题一览

1. 创建容器时候报错：WARNING: IPv4 forwarding is disabled. Networking will not work.

然后在编译java ccenv时候报错。原因是无法访问服务器https://plugins.gradle.org/api/gradle/2.12/plugin/use/com.google.protobuf/0.8.0

FAILURE: Build failed with an exception.

\* Where:

Build file '/root/core/chaincode/shim/java/build.gradle' line: 30

\* What went wrong:

Error resolving plugin [id: 'com.google.protobuf', version: '0.8.0']

> Could not GET 'https://plugins.gradle.org/api/gradle/2.12/plugin/use/com.google.protobuf/0.8.0'.

> plugins.gradle.org: Temporary failure in name resolution

\* Try:

Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get more log output.

解决办法

【CentOS】

vim /usr/lib/sysctl.d/00-system.conf

添加

net.ipv4.ip\_forward=1

重启network服务

systemctl restart network