

编译环境搭建

HUST H3C 实验室

2015-4-19

朱林峰

目录

编译环境搭建	2
1 配置 Ubuntu 虚拟机	2
1.1 官网下载 Ubuntu server 14.04	2
1.2 Virtualbox 新建虚拟机	2
1.3 启动虚拟机进行初始配置	3
2 安装并配置基础软件	12
2.1 启动 Ubuntu server	12
2.2 确认能够连接外网	12
2.3 创建一些熟悉的目录	12
2.4 系统更新	13
2.5 安装基础软件软件	13
2.6 下载 opendaylight 代码	13
2.7 配置 Samba 服务	13
2.8 配置网桥连接	14
2.9 访问 Ubuntu 共享文件	14
3 配置 JDK&maven	14
3.1 配置 java 环境变量	14
3.2 安装并配置 maven 环境	15
4 配置 maven 镜像服务	16
4.1 启动 nexus 代理服务	16
5 编译 controller	16

编译环境搭建

1 配置 Ubuntu 虚拟机

1.1 官网下载 Ubuntu server 14.04

网址: <http://www.ubuntu.com/download/server/>

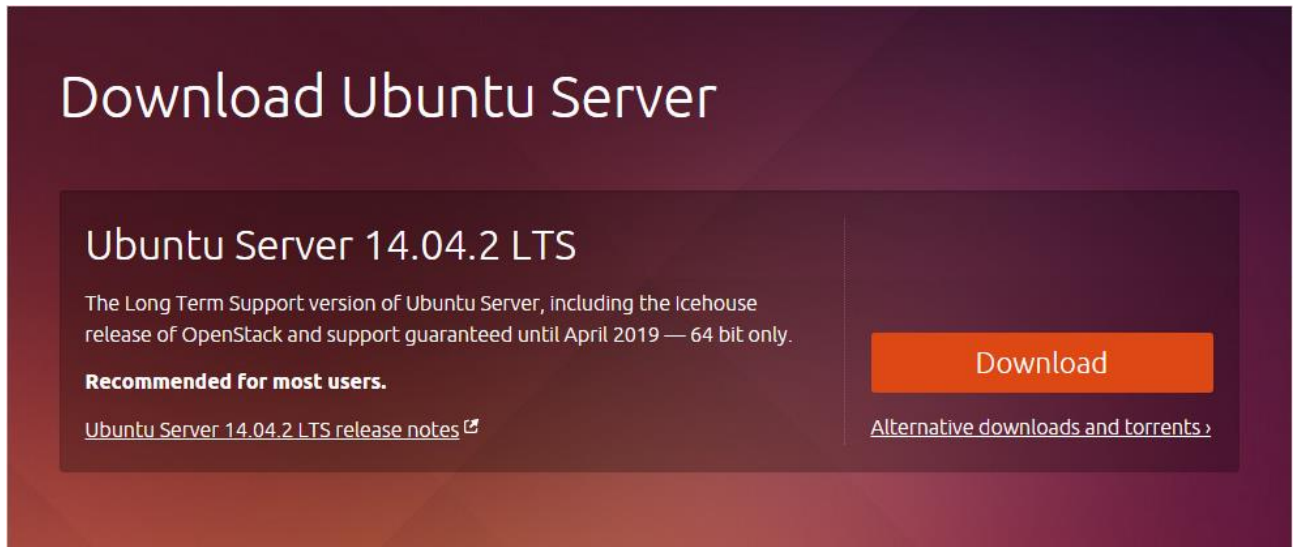


图 1-1: Ubuntu 官网下载链接

1.2 Virtualbox 新建虚拟机

点击新建，弹出下图：



图 1-2: virtualbox 新建虚拟机

名称: Ubuntu

类型：Linux

版本：Ubuntu(64 bit)

下一步→内存大小(2048M)→下一步→现在创建虚拟硬盘→VDI→下一步→动态分配→选择虚拟磁盘存放
路径→虚拟硬盘大小(30G)→创建

1.3 启动虚拟机进行初始配置

如果主机电脑是 32 位，以前没有用 Virtualbox 运行过 64 位虚拟机，会报出错误，说 32 位系统不支持运行 64 位虚拟机。这时需要重启电脑，在 BIOS 加载操作系统前进行 BIOS 设置，设置允许 CPU 虚拟化功能。每个人电脑主板不同，具体怎么设置还得自己摸索。

Virtualbox 选中刚才新建的虚拟机→启动→选择启动盘→选择下载好的 Ubuntu server 镜像→启动

下面开始 Ubuntu server 的初始配置。

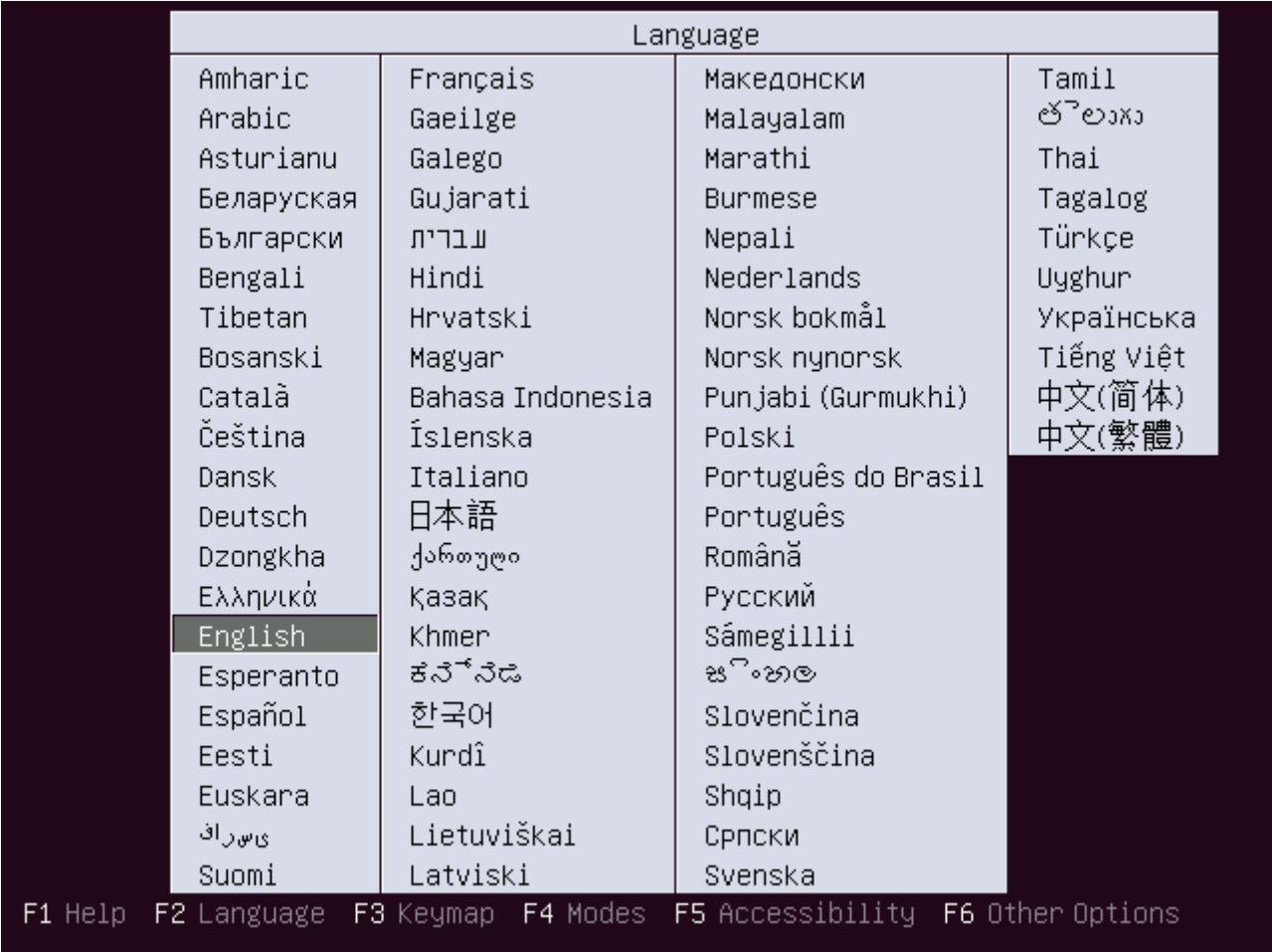


图 1-3：Ubuntu server 选择语言→English

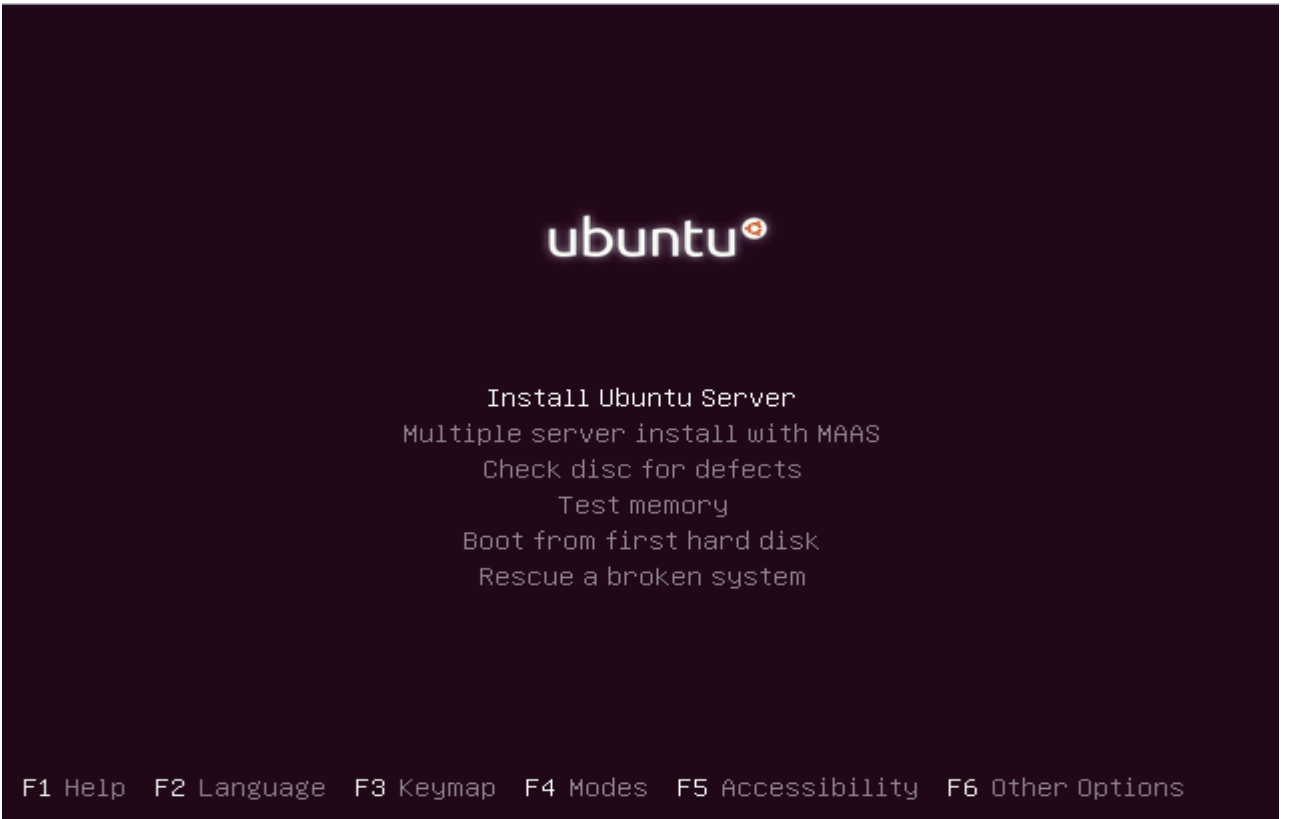


图 1-4：选择 Install Ubuntu Server

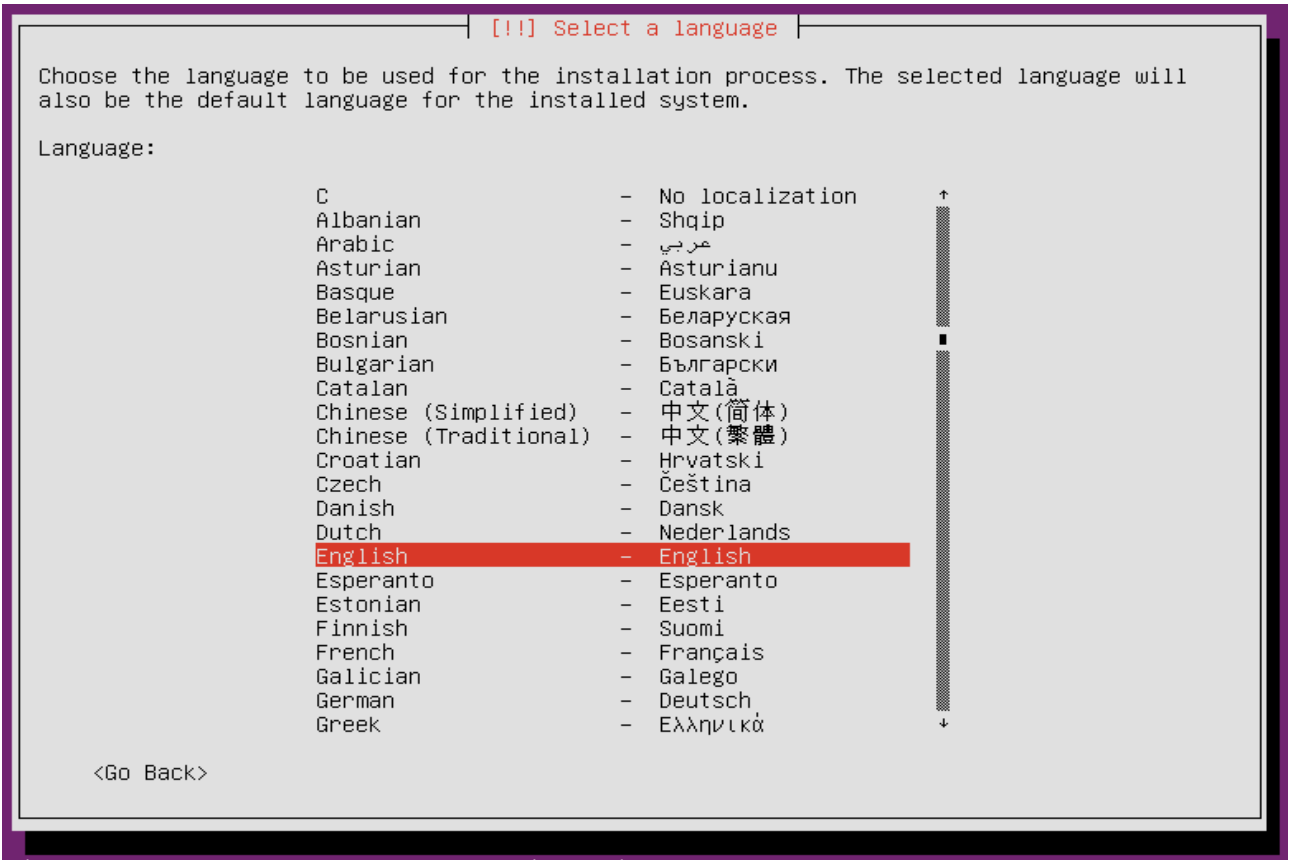


图 1-5：选择语言→English

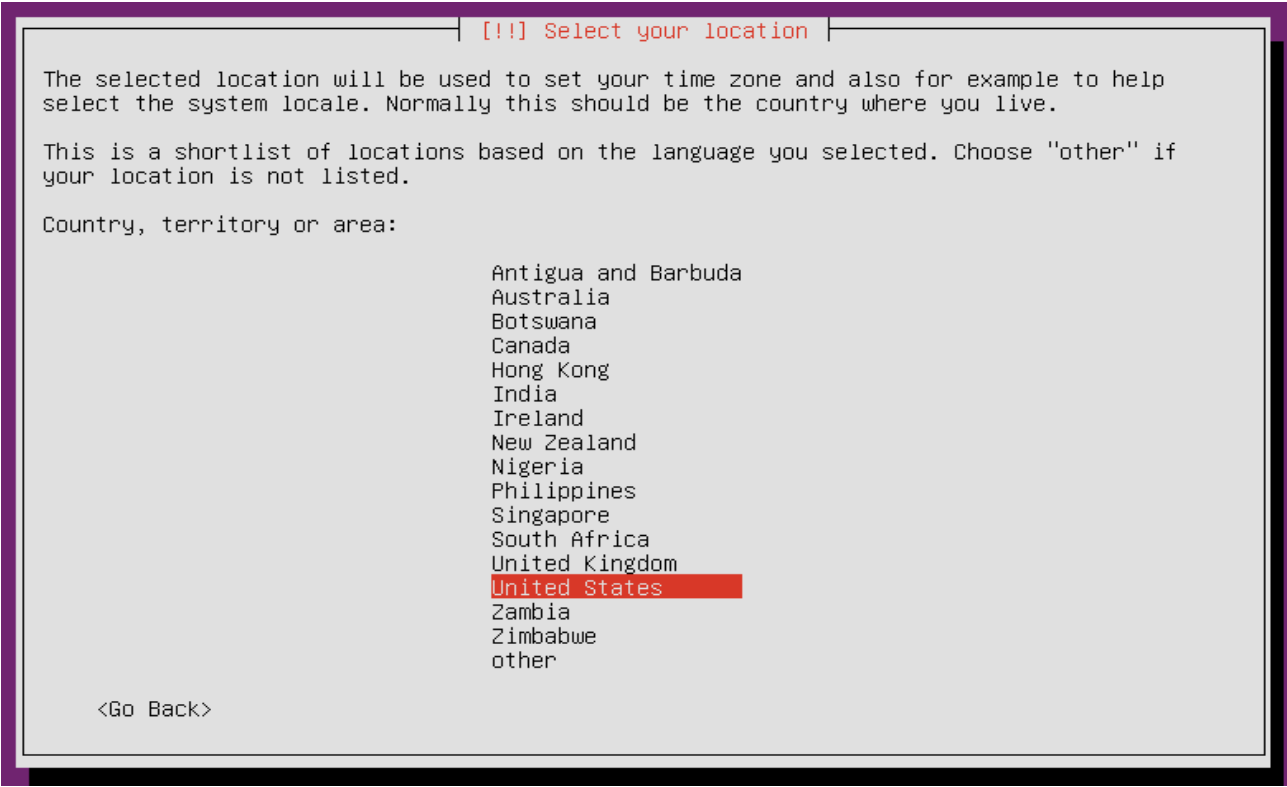


图 1-6：选择地区→Asia

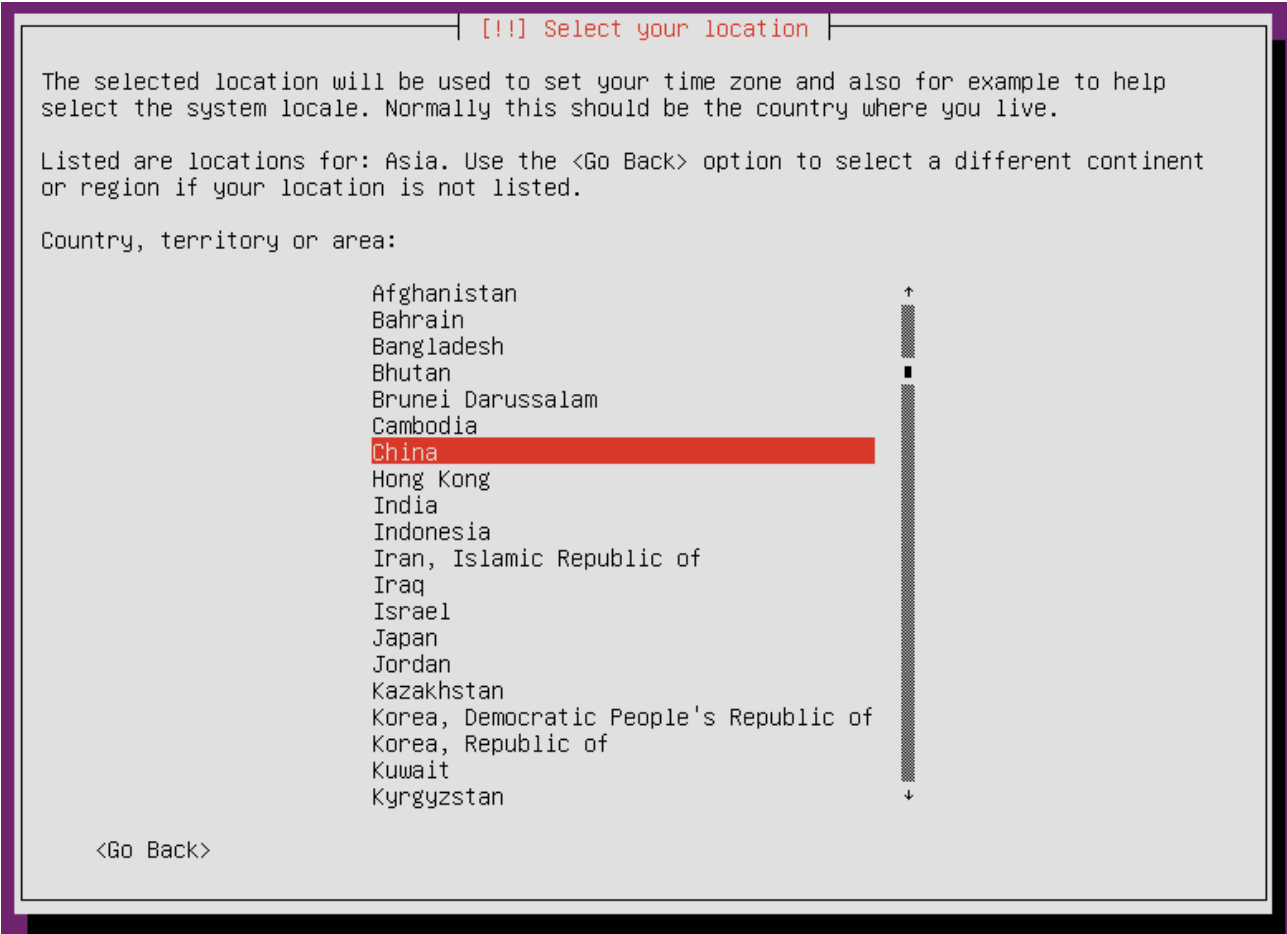


图 1-7：选择国家→China

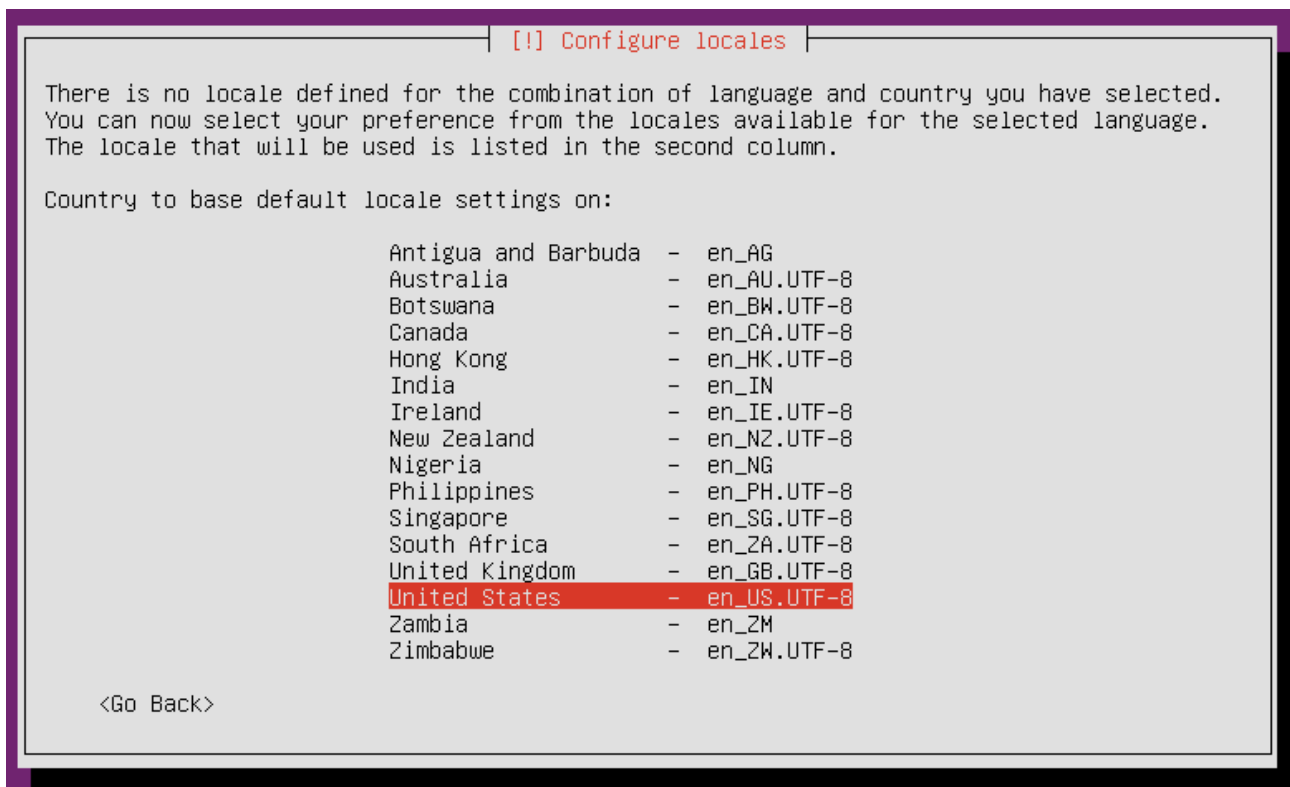


图 1-8：选择 United States

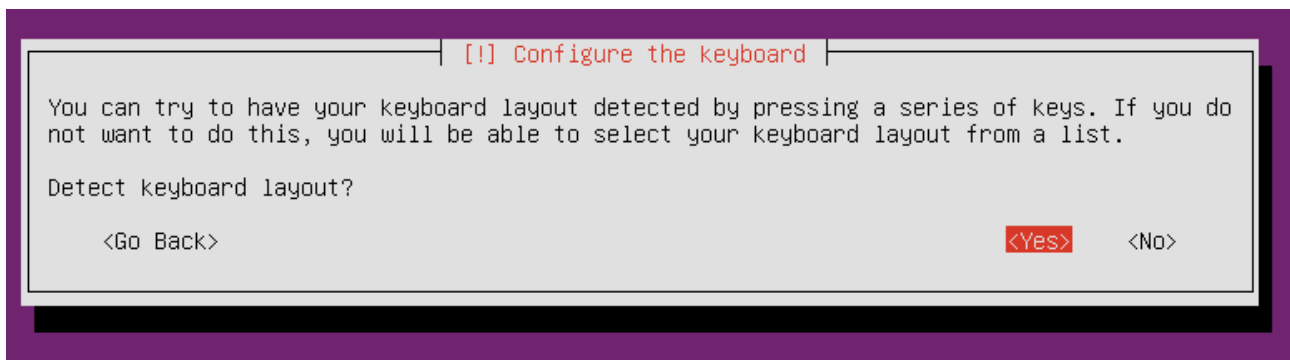


图 1-9：测试键盘布局→yes

键盘测试过程忽略，照提示选择 yes/no 即可。

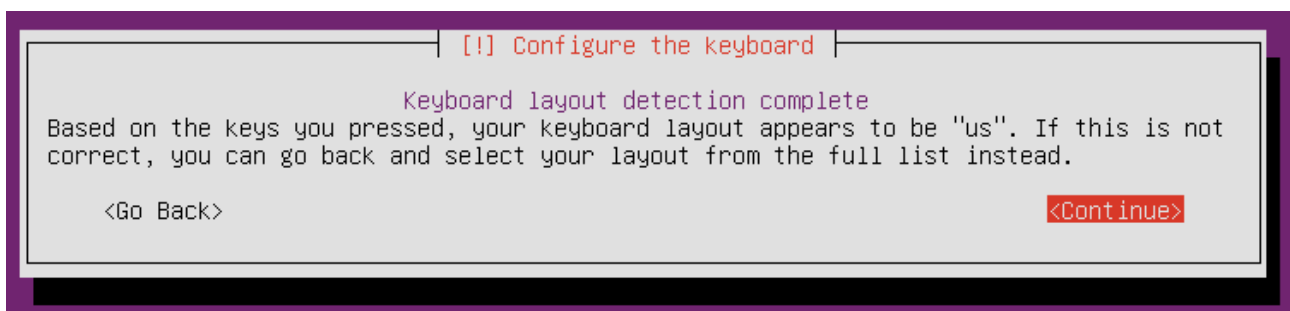


图 1-10：键盘测试完毕，结果是 us 键盘布局，选择 continue

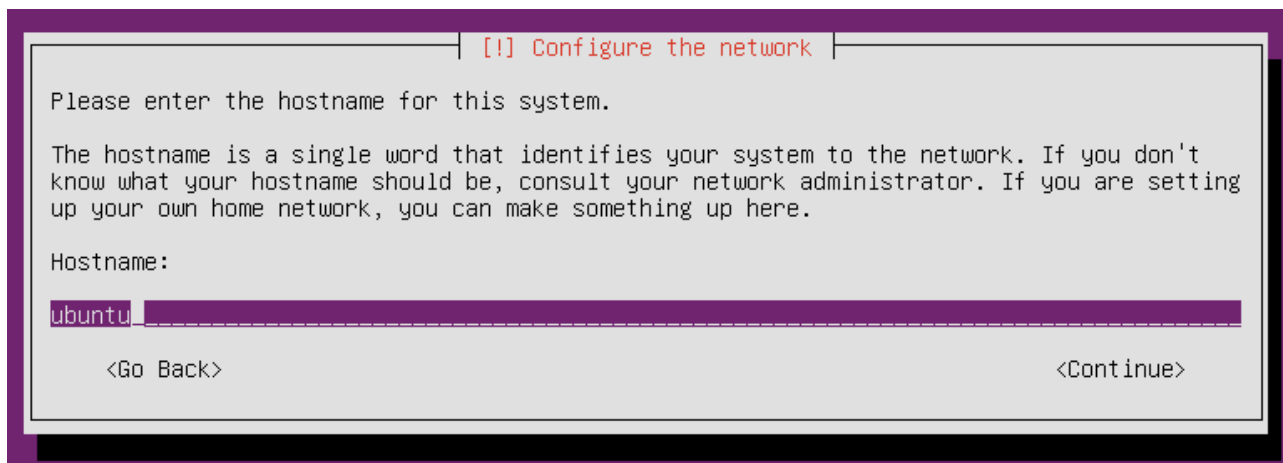


图 1-11: 配置 Hostname→ubuntu

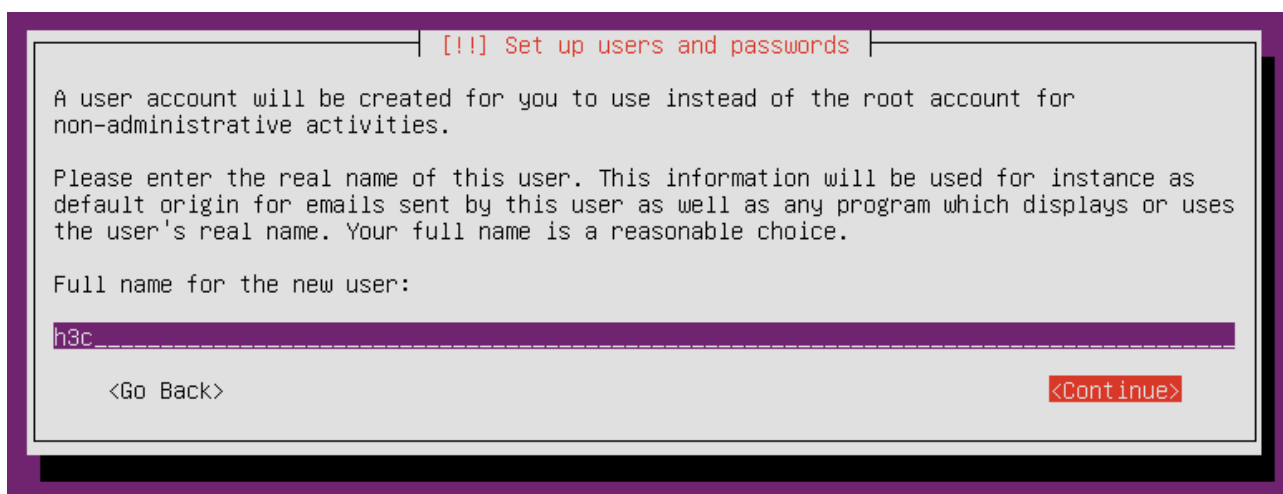


图 1-12: 创建一个账户→h3c

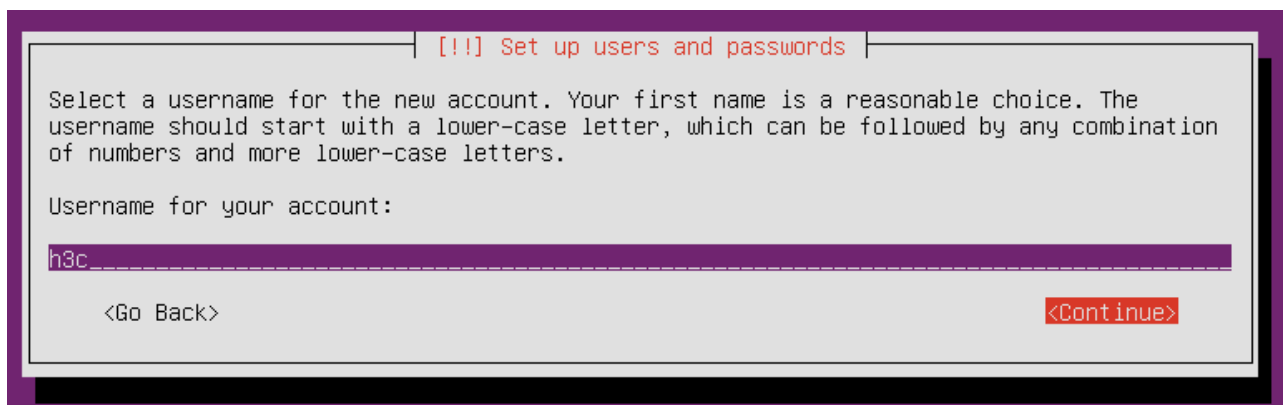


图 1-13: 用户名→h3c

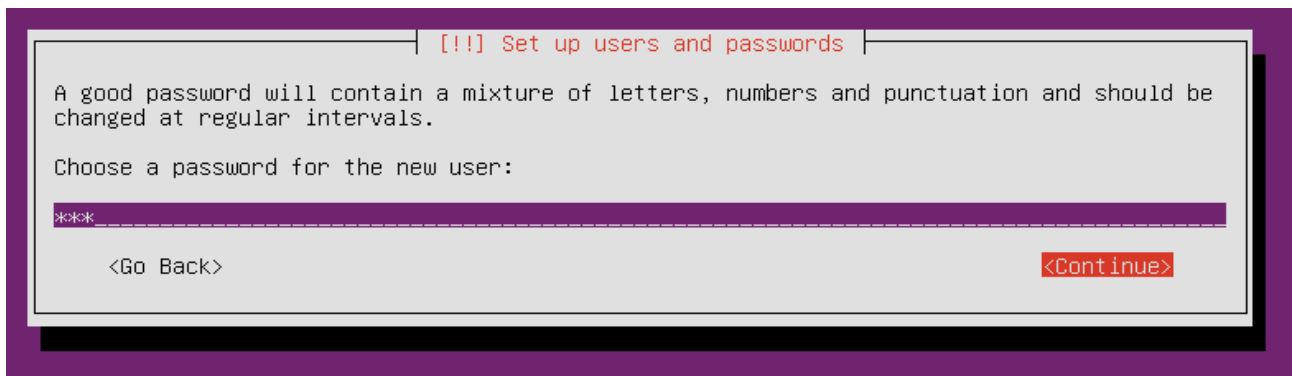


图 1-14: password→123

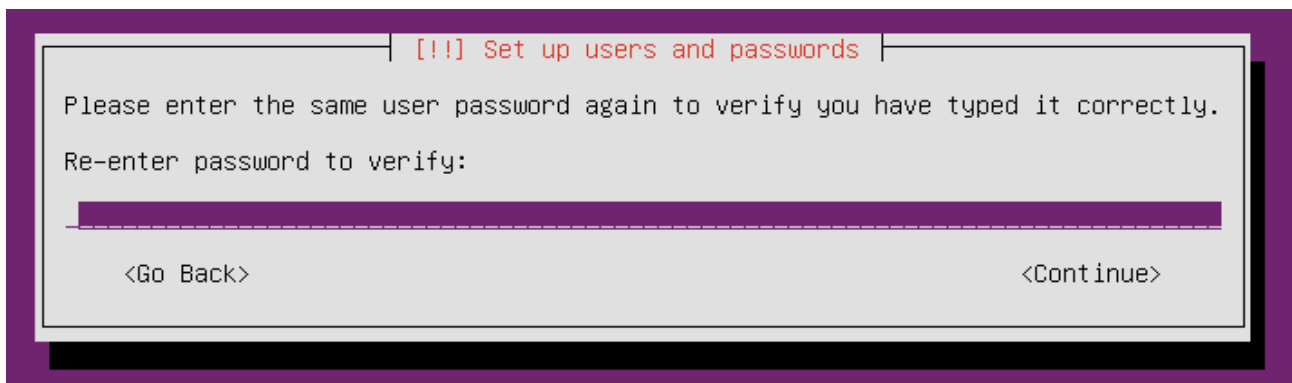


图 1-15: 确认密码

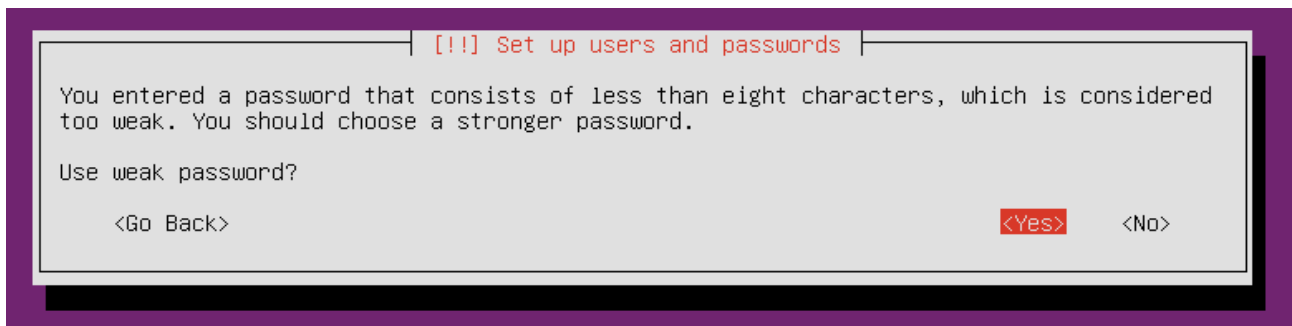


图 1-16: 它说密码太弱, 但不管它, 我行我素→yes

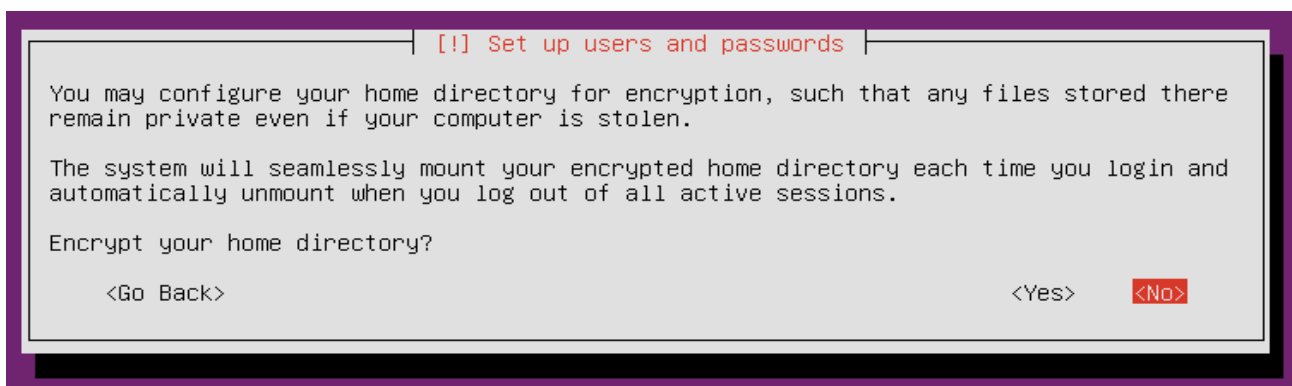


图 1-17: 文件系统是否加密→No

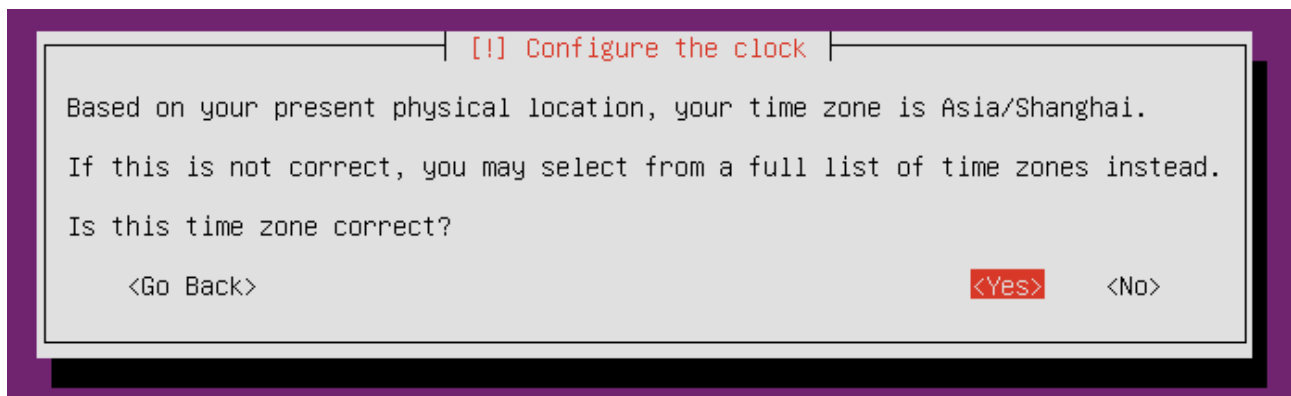


图 1-18: 确认时区是上海→Yes

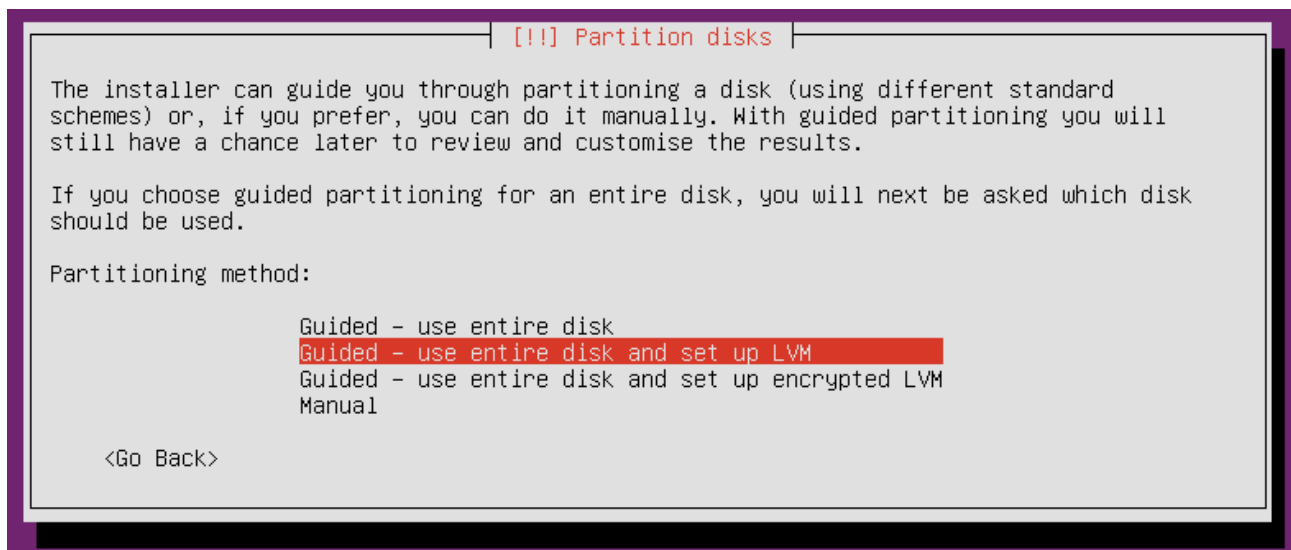


图 1-19: 关于磁盘分区→不知道, 使用默认选择

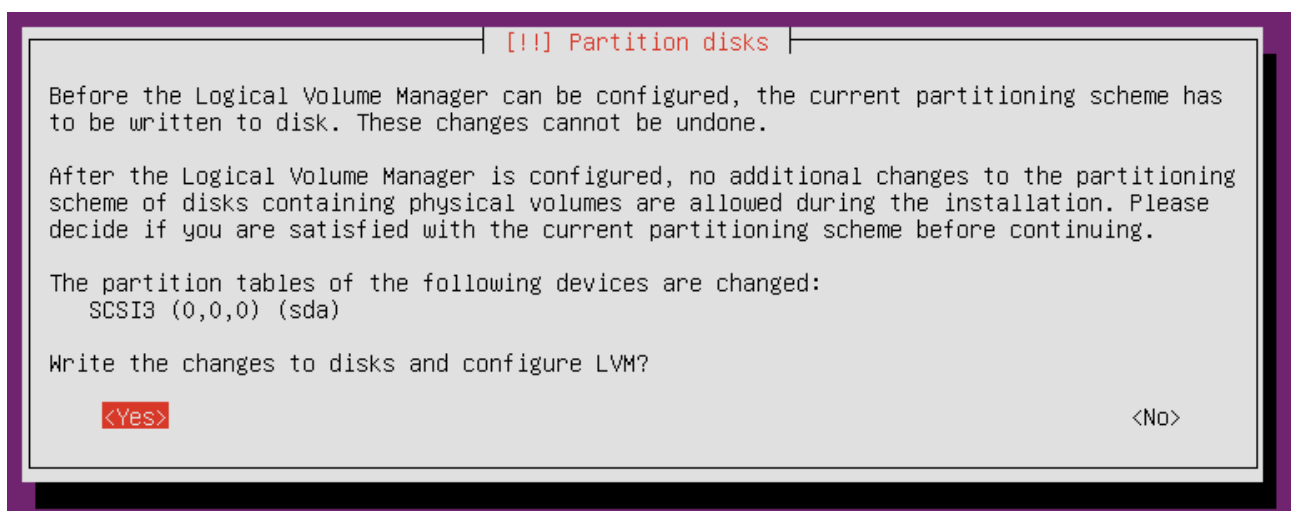


图 1-20: Yes



图 1-21: 32G 磁盘全部分到一个分区, 这是最方便的

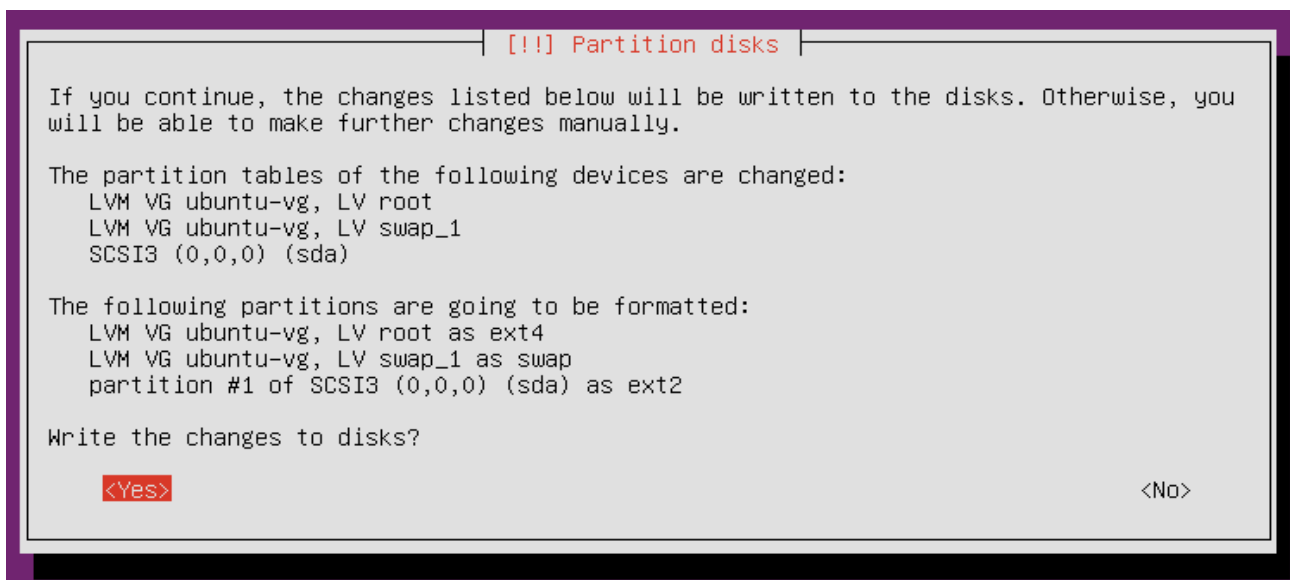


图 1-22: Yes

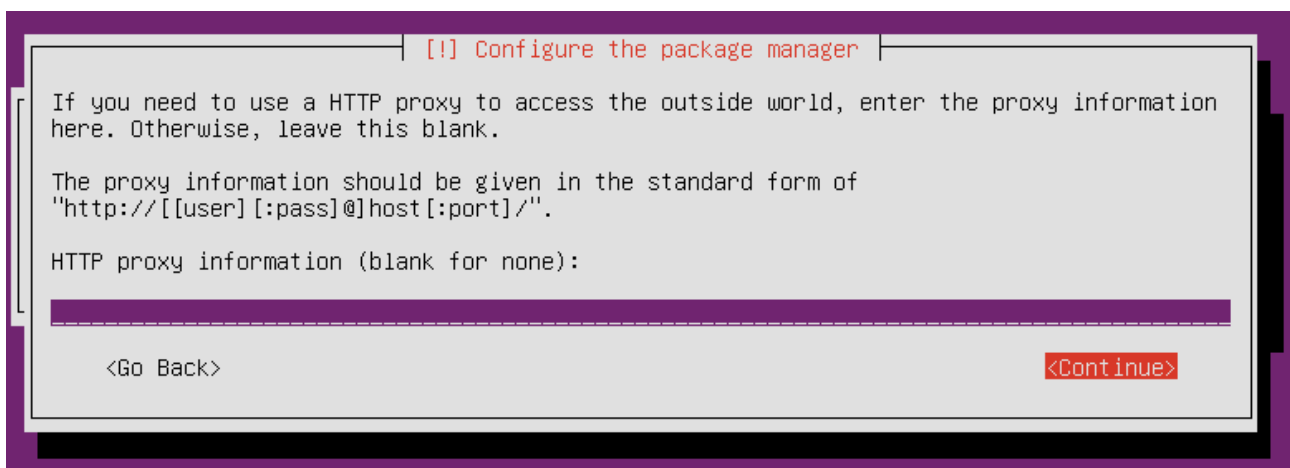


图 1-23: 我们不需要 http 代理, 所以直接略过

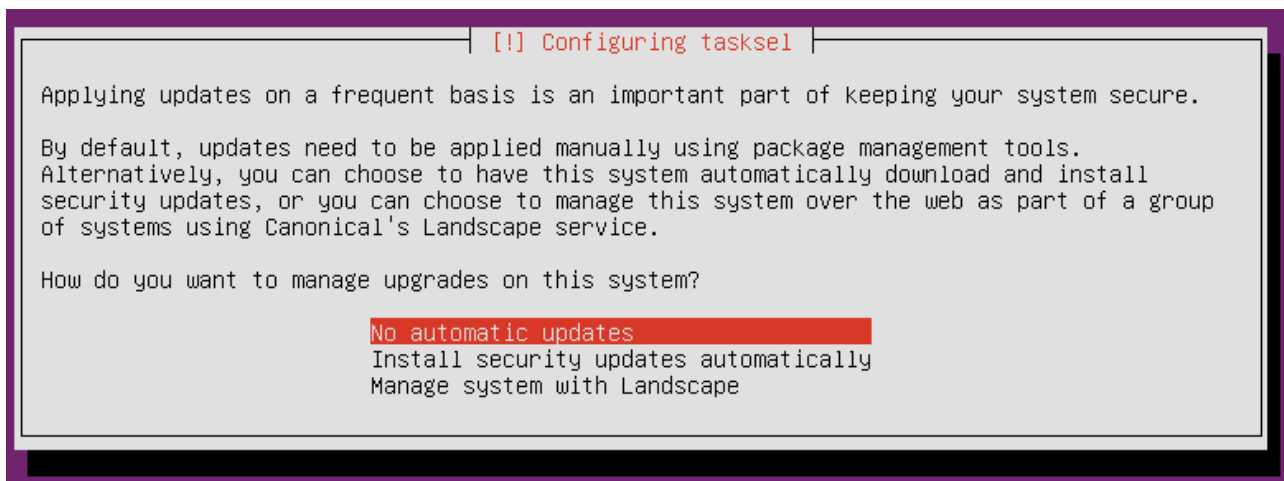


图 1-24：既然虚拟机不联网，没必要自动更新

按空格选择预装服务，忘了截图：

OpenSSL, DNS, Samba

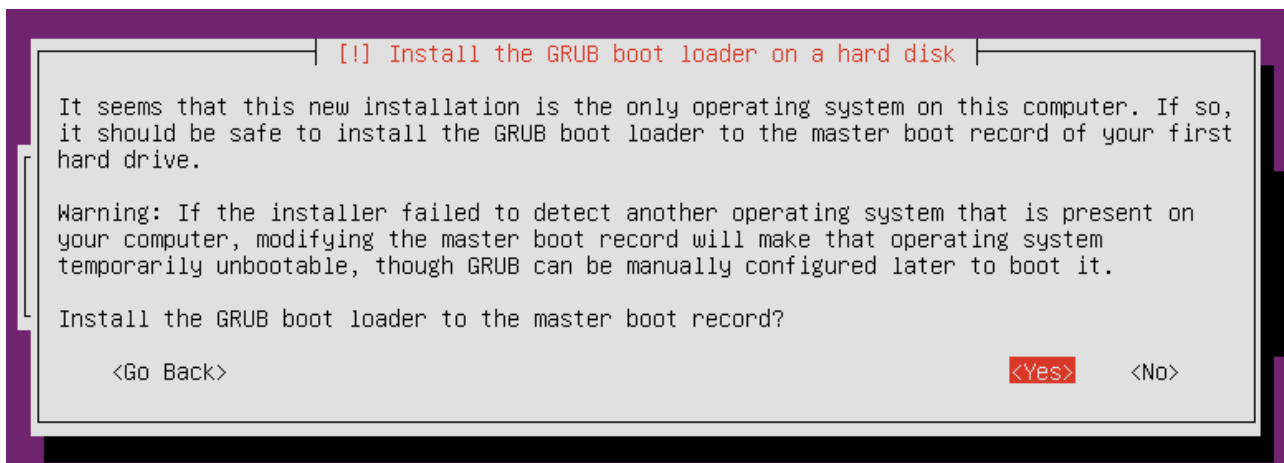


图 1-25：安装 GRUB 引导→Yes

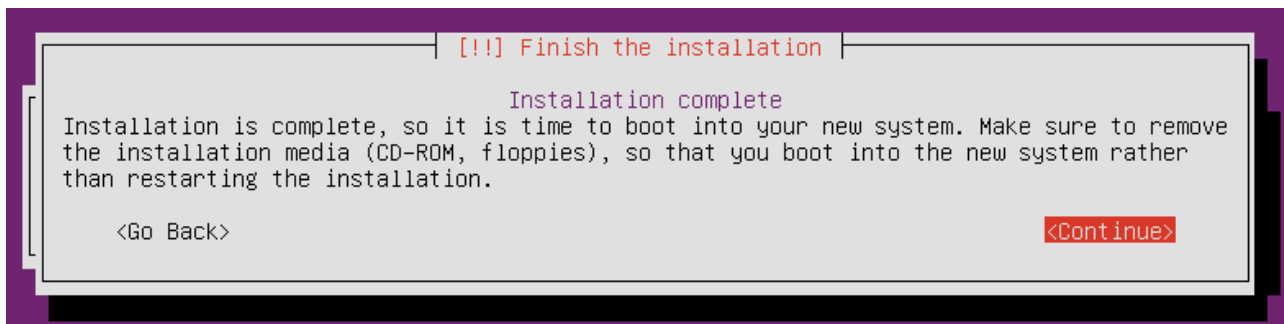
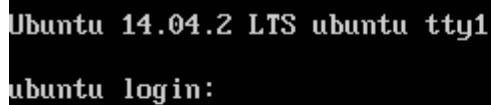


图 1-26：almost done→Continue

2 安装并配置基础软件

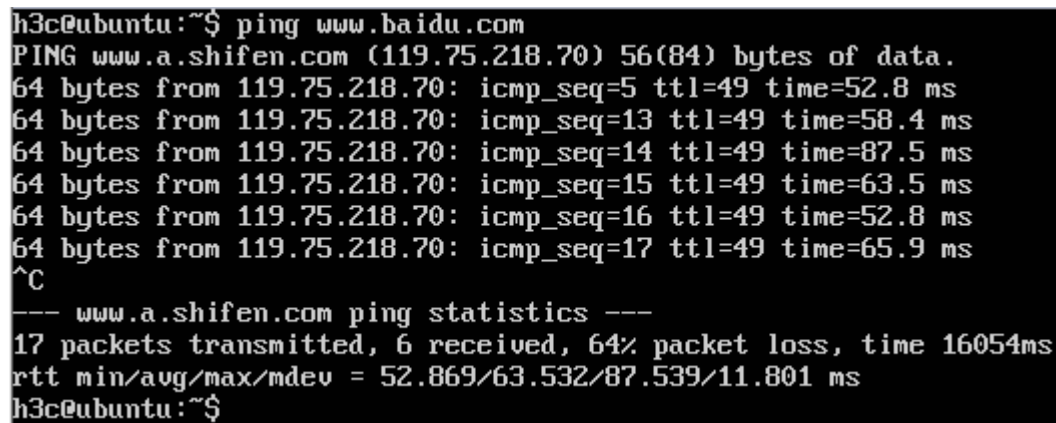
2.1 启动 Ubuntu server



```
Ubuntu 14.04.2 LTS ubuntu tty1
ubuntu login:
```

图 2-1: 启动 Ubuntu, user:h3c; passwd:123

2.2 确认能够连接外网

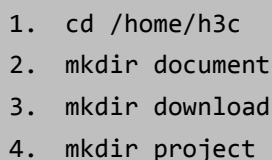


```
h3c@ubuntu:~$ ping www.baidu.com
PING www.a.shifen.com (119.75.218.70) 56(84) bytes of data.
64 bytes from 119.75.218.70: icmp_seq=5 ttl=49 time=52.8 ms
64 bytes from 119.75.218.70: icmp_seq=13 ttl=49 time=58.4 ms
64 bytes from 119.75.218.70: icmp_seq=14 ttl=49 time=87.5 ms
64 bytes from 119.75.218.70: icmp_seq=15 ttl=49 time=63.5 ms
64 bytes from 119.75.218.70: icmp_seq=16 ttl=49 time=52.8 ms
64 bytes from 119.75.218.70: icmp_seq=17 ttl=49 time=65.9 ms
^C
--- www.a.shifen.com ping statistics ---
17 packets transmitted, 6 received, 64% packet loss, time 16054ms
rtt min/avg/max/mdev = 52.869/63.532/87.539/11.801 ms
h3c@ubuntu:~$
```

图 2-2: ping www.baidu.com

2.3 创建一些熟悉的目录

代码 2-1: 创建常规目录



```
1. cd /home/h3c
2. mkdir document
3. mkdir download
4. mkdir project
```

ls 命令后:



```
h3c@ubuntu:~$ ls
document download project
h3c@ubuntu:~$ _
```

图 2-3: 显示自己创建的目录

2.4 系统更新

代码 2-2：系统更新

```
1. sudo apt-get update
2. sudo apt-get upgrade
```

为什么要系统更新？因为才安装的系统，许多软件都是比较过时的，有一些 bug 有待修复，所以需要在更新中获取最新的系统组件，这样系统才能胜任稳定的工作，为后面的一系列配置打好基础。

2.5 安装基础软件软件

代码 2-3：安装基础软件

```
1. sudo apt-get install git           //git 版本控制工具
2. sudo apt-get install emacs24       //好用的编辑器
3. sudo apt-get install openjdk-7-jre
4. sudo apt-get install openjdk-7-jdk
5. sudo apt-get install cloc           //代码统计工具，可不装
6. sudo apt-get install tree           //以树形形式打印目录的工具
7. sudo apt-get install gcc            //C 语言编译工具，非项目相关
8. sudo apt-get install gdb            //C 语言调试工具，非项目相关
9. sudo apt-get install g++            //C++编译工具，非项目相关
```

2.6 下载 opendaylight 代码

```
1. cd /home/h3c/project
2. git clone -b stable/helium https://git.opendaylight.org/gerrit/p/controller.git
3. git clone -b stable /helium https://git.opendaylight.org/gerrit/p/integration.git
4. git clone -b stable /helium https://git.opendaylight.org/gerrit/p/l2switch.git
5. git clone -b stable /helium https://git.opendaylight.org/gerrit/p/yangtools.git
```

2.7 配置 Samba 服务

将 home 目录设置为共享目录：

```
1. sudo chmod 777 /home
```

配置 Samba，sudo 打开/etc/samba/smb.conf 文件进行编辑。

在 `workgroup = WORKGROUP` 下添加一行 `security = share`。

在文件底端添加如下内容：

代码 2-4：配置 smb.conf

```
1. [share]
2.     comment = share all
3.     path = /tmp/samba
4.     browseable = yes
5.     public = yes
```

6. writable = yes

最后启动 Samba, `sudo /etc/init.d/smb start`。

2.8 配置网桥连接

关闭虚拟机, `设置→网络→网卡 1→连接方式(选择桥接网卡)→确定`, 重启虚拟机。

1. `sudo emacs /etc/network/interfaces` //打开网卡配置文件

代码 2-5: 配置静态桥接网卡

```
1. auto eth0
2. iface eth0 inet static
3. address xxx.xxx.xxx.xxx
4. netmask 255.255.255.0
```

IP 地址一定要配置的和主机处于同一子网才可。

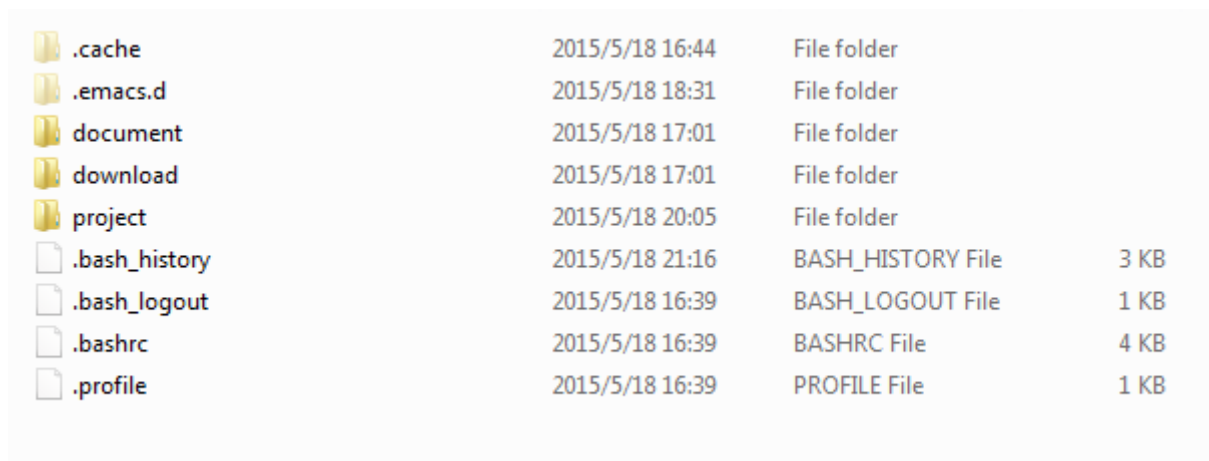
使配置好的网卡生效, 如果没有生效就重启虚拟机好了:

```
1. sudo ifconfig eth0 inet up
2. sudo /etc/init.d/networking restart
```

虚拟机和主机互相 ping, 看是否配置成功。

2.9 访问 Ubuntu 共享文件

windows 主机下, `windows + R→输入\\虚拟机 IP→回车`, 即可访问共享文件了。



.cache	2015/5/18 16:44	File folder	
.emacs.d	2015/5/18 18:31	File folder	
document	2015/5/18 17:01	File folder	
download	2015/5/18 17:01	File folder	
project	2015/5/18 20:05	File folder	
.bash_history	2015/5/18 21:16	BASH_HISTORY File	3 KB
.bash_logout	2015/5/18 16:39	BASH_LOGOUT File	1 KB
.bashrc	2015/5/18 16:39	BASHRC File	4 KB
.profile	2015/5/18 16:39	PROFILE File	1 KB

图 2-4: Windows 访问 Ubuntu 共享文件

3 配置 JDK&maven

3.1 配置 java 环境变量

JDK 安装目录: `/usr/lib/jvm/java-7-openjdk-amd64`。

sudo 打开文件 `/etc/profile` 在末尾进行编辑:

代码 3-1:配置 JAVA 环境变量

```
1. #java path
2. export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
3. export JRE_HOME=${JAVA_HOME}/jre
4. export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
5. export PATH=${JAVA_HOME}/bin:${JRE_HOME}/bin:$PATH
```

编辑完毕保存，在命令行输入 `source /etc/profile` 使环境变量生效，接下来进行测验。

命令行输入 `echo $JAVA_HOME` 显示的路径为配置的 `JAVA_HOME` 即为正确，输入 `echo $PATH`，显示的路径里面包含配置的 `java` 路径即为正确，命令行输入 `echo $CLASSPATH`，显示的路径里面包含配置的 `CLASSPATH` 路径即为正确。

3.2 安装并配置 maven 环境

通过命令行在线安装的 `maven` 版本太低，无法编译 `controller`，所以需要将正确版本的 `maven` 安装包拷入 `Ubuntu` 进行安装。`maven` 版本为 **apache-maven-3.2.5-bin**，下载链接：<http://maven.apache.org/download.cgi>。

首先命令行输入 `echo $JAVA_HOME` 等检查 `java` 环境变量。

然后将 `maven` 解压后的目录拷贝至 `/usr/local` 路径下，为了日后方便升级，可以创建一个平行的符号链接，符号链接类似于 `Windows` 下的快捷图标，下面相当于创建了一个快捷图标 `apache-maven` 指向原来的 `maven` 安装目录 `apache-maven-3.2.5`：

```
1. sudo ln -s apache-maven-3.2.5 apache-maven
```

然后配置 `maven` 环境变量指向刚才新建的符号链接，依旧 `sudo` 打开 `/etc/profile` 文件进行编辑：

代码 3-2: 配置 maven 环境变量

```
1. #maven path
2. export M2_HOME=/usr/local/apache-maven
3. export PATH=$PATH:$M2_HOME/bin
4. export MAVEN_OPTS='-Xmx1024m -XX:MaxPermSize=1024m'
```

编辑完毕保存，在命令行输入 `source /etc/profile` 使环境变量生效，接下来进行测验。

命令行输入 `echo $M2_HOME`，显示 `maven` 路径 `/usr/local/apache-maven` 即为正确；命令行输入 `mvn -v`，显示 `maven` 路径和 `java` 路径即为正确。

在让虚拟机能够连接外网的前提下，命令行输入 `mvn help:system`，这会让 `maven` 在本地初始化一个目录为 `m2` 的仓库，并从中央仓库下载 `maven-help-plugin` 至本地仓库。

将 `settings.xml` 拷贝至 `m2` 目录，以后对 `maven` 的配置就只会影响到此 `m2` 仓库：

```
1. cd /usr/local/apache-maven/conf
```



```
2. sudo cp settings.xml /home/h3c/.m2
```

4 配置 maven 镜像服务

4.1 启动 nexus 代理服务

nexus 安装在 Windows 下，切换至 nexus 的安装目录，然后进入 `nexus-2.11.2-06\bin\jsw\windows-x86-64` 目录，运行 `consols-nexus.bat` 启动 nexus。启动成功后，在 nexus 所在主机的浏览器地址栏输入 `localhost:8081/nexus` 即进入 nexus web 服务的首页。

sudo 打开 `.m2/settings.xml` 文件进行编辑，在 `<mirrors></mirrors>` 标签内添加如下代码后保存退出，即可进行项目编译：

```
1. <mirror>
2.   <id>mirrorId</id>
3.   <mirrorOf>*</mirrorOf>
4.   <name>mirrorId</name>
5.   <url>http://192.168.0.11:8081/nexus/content/groups/public</url>
6. </mirror>
```

5 编译 controller

如果没有对项目权限进行变更，那么编译时很可能会报错，所以事先需要更改项目目录的权限，使之可读可写可执行：`sudo chmod 777 -R projectDirectory`。

进入 controller 根目录，输入命令 `mvn clean install -DskipTests`，进行编译。