1. 安装物理机或虚拟机

物理机，此处不再多解释，大家去网上搜一下。

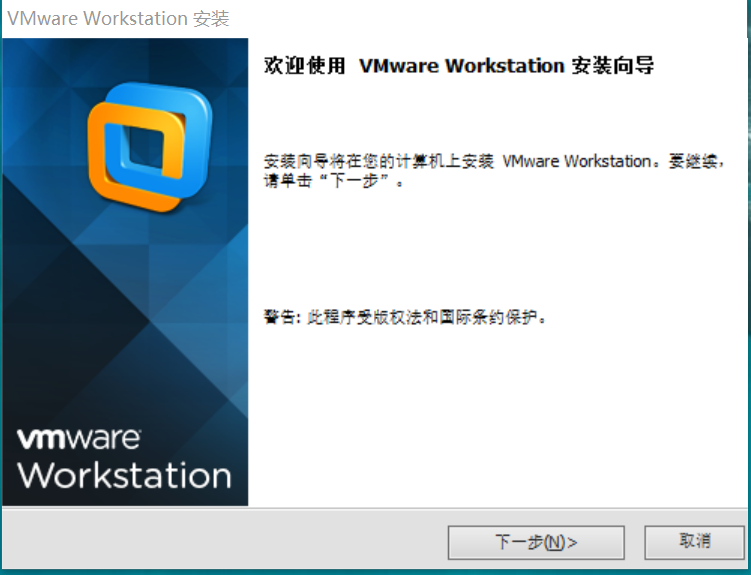
虚拟机等于用软件模拟一台PC，可以随便修改、直接删除，非常便于调试，推荐这种方式。

虚拟机下载地址：

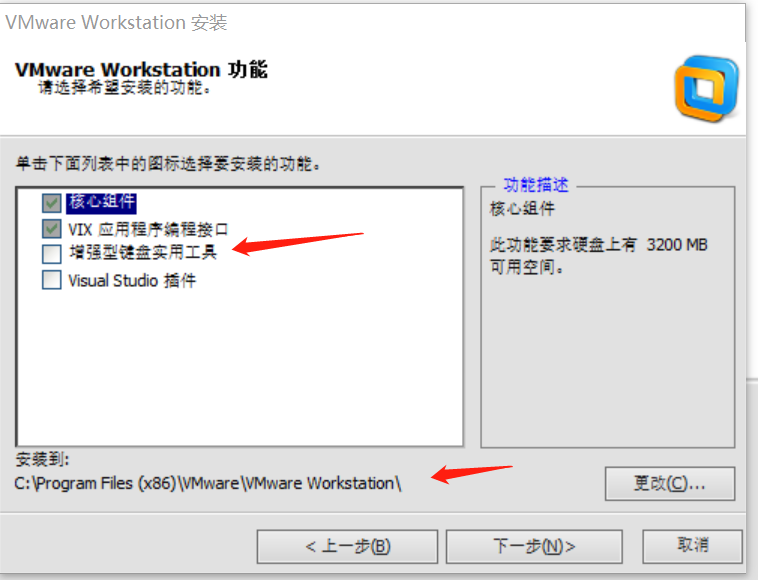
<https://pan.baidu.com/s/1KRnugxcn0ZJhiI0uiwzs1g> 9zrc

<https://pan.baidu.com/s/1XvjIh0y1cLDhPDOXQRejpw> roq5

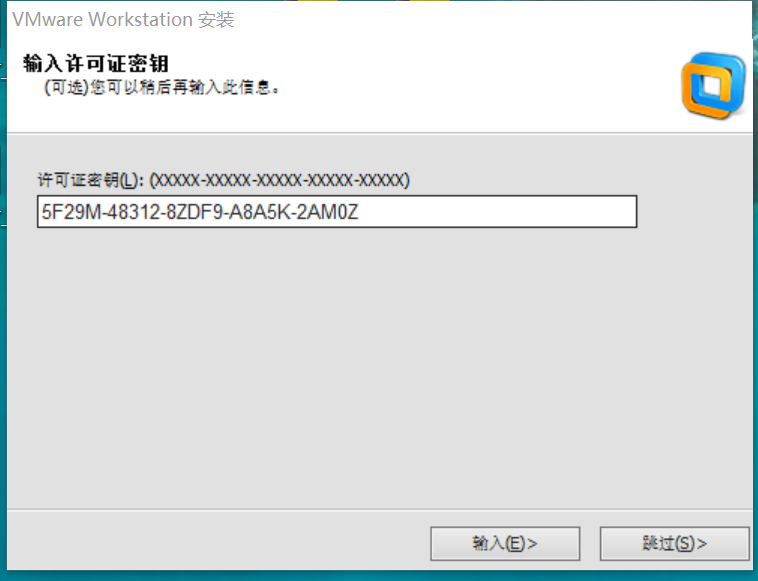
有些同学系统版本太高，请选高版本，如果高版本还不行，自行去网上搜更高版本。







对于想在虚拟机和物理机来回切换的，建议勾选实用工具，其它几乎全是下一步。



这个东西，网盘里有。大家自行直接复制粘贴。

安装成功后，桌面如下:



1. 虚拟机创建

LINUX厂商及版本：UBUNTU 18.04

LINUX镜像地址：<http://mirrors.ustc.edu.cn/ubuntu-releases/18.04.6/ubuntu-18.04.6-live-server-amd64.iso>

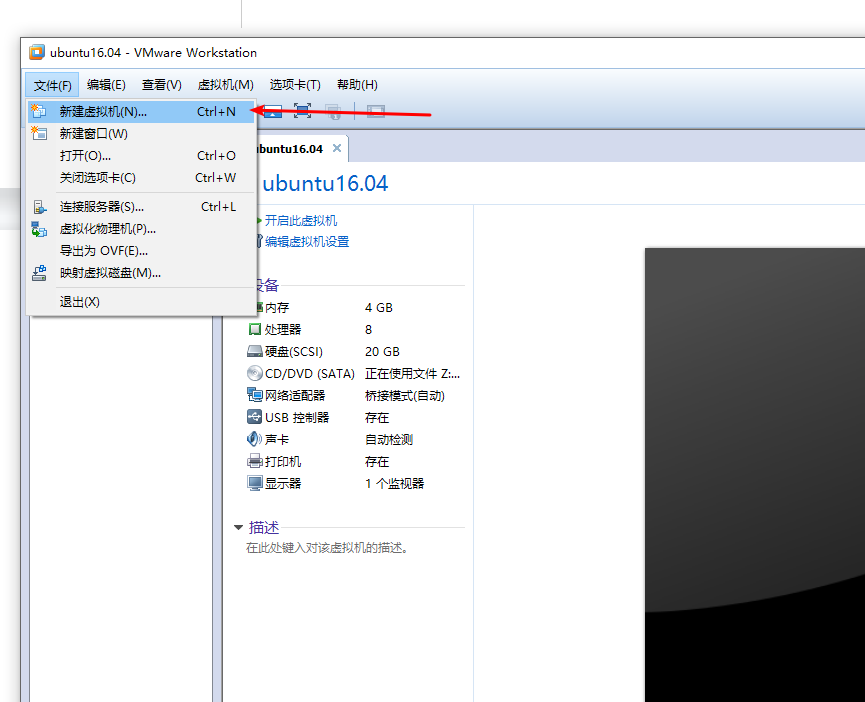
<http://mirrors.163.com/ubuntu-releases/18.04.6/ubuntu-18.04.6-live-server-amd64.iso>

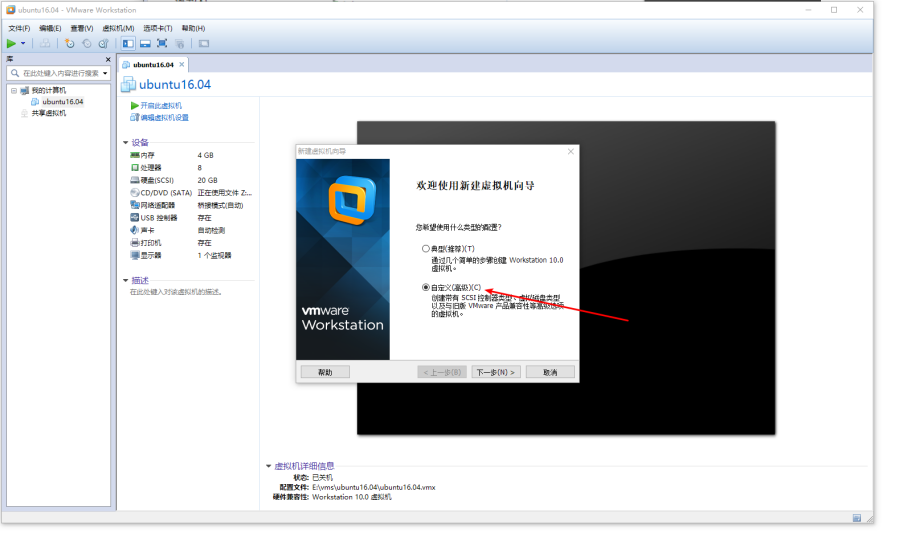
以上是SERVER版本，默认不含桌面，在实际实验中也无需桌面，建议选取该版本。如果不习惯，可以在相应目录下自行寻找。

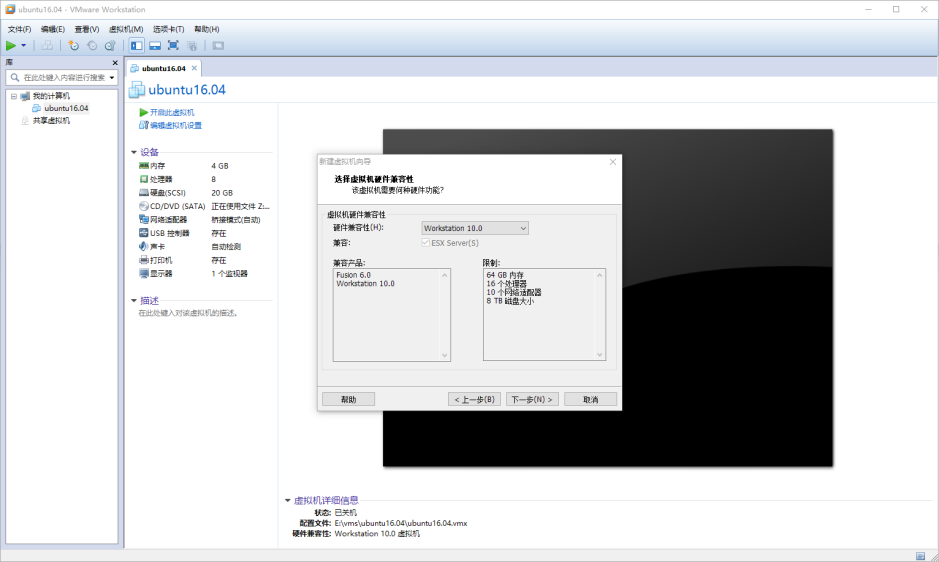
物理机，可以刻盘，或用可引导U盘。方式去网上搜（关键字 UBUNTU可启动U盘）。

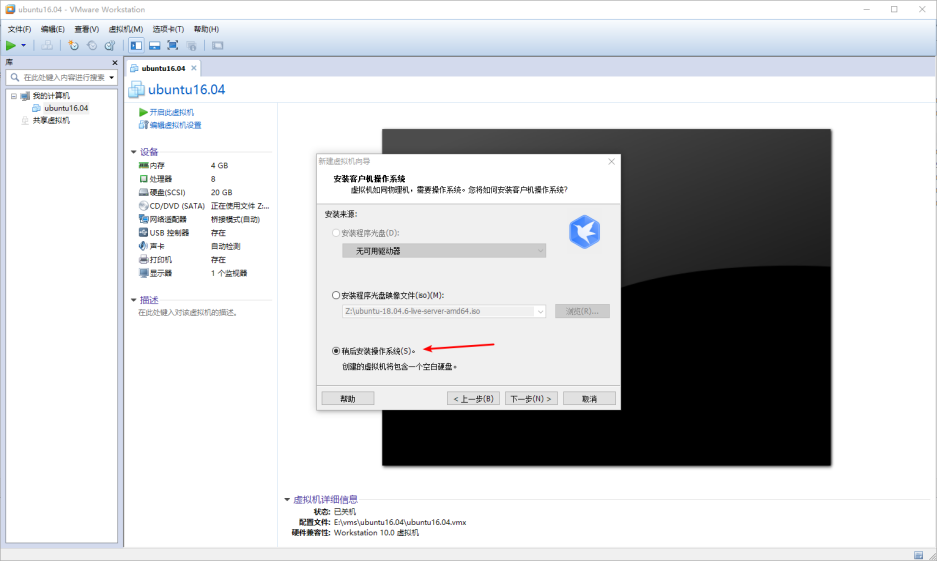
虚拟机，以光驱方式载入, 一步步安装。

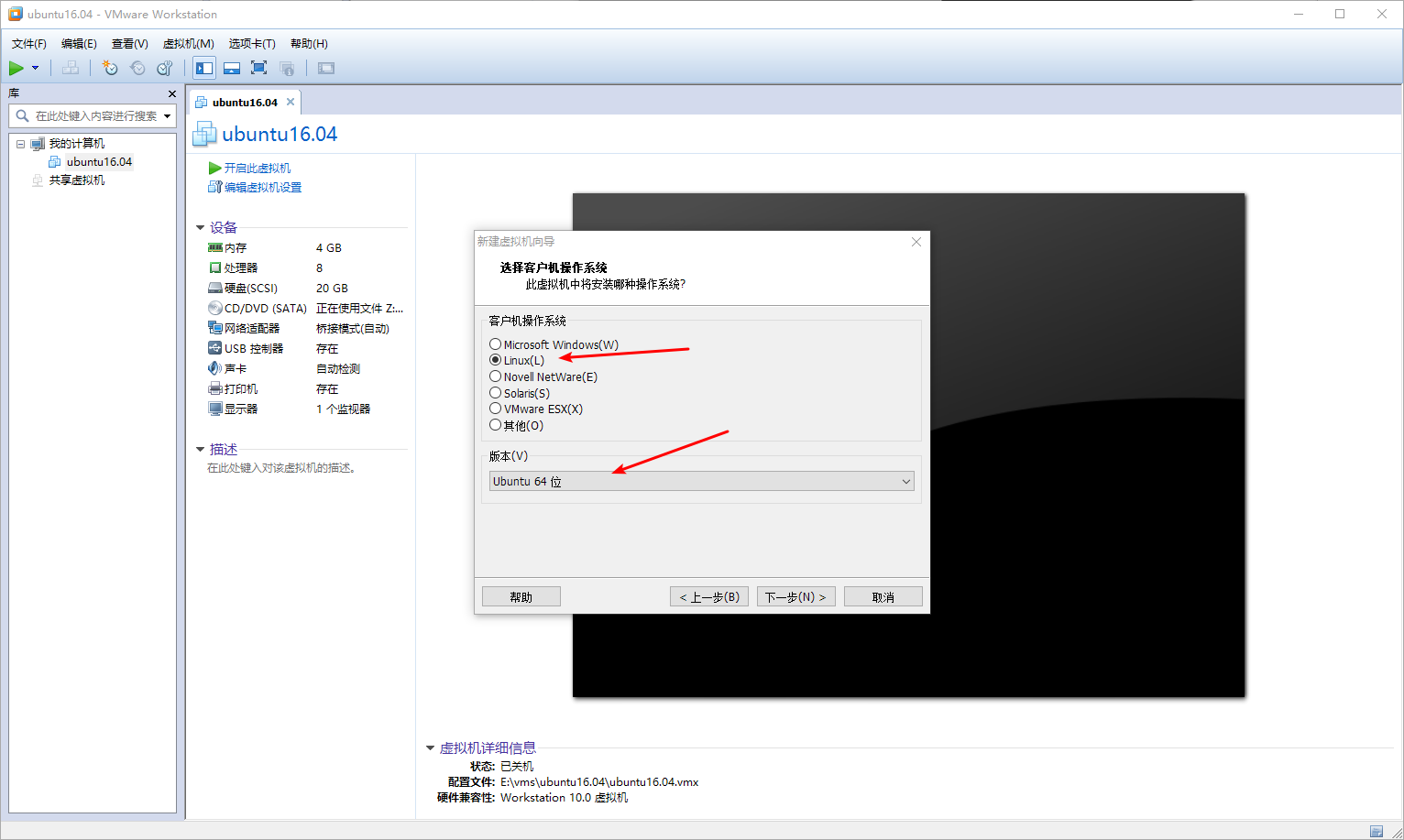
如下图：

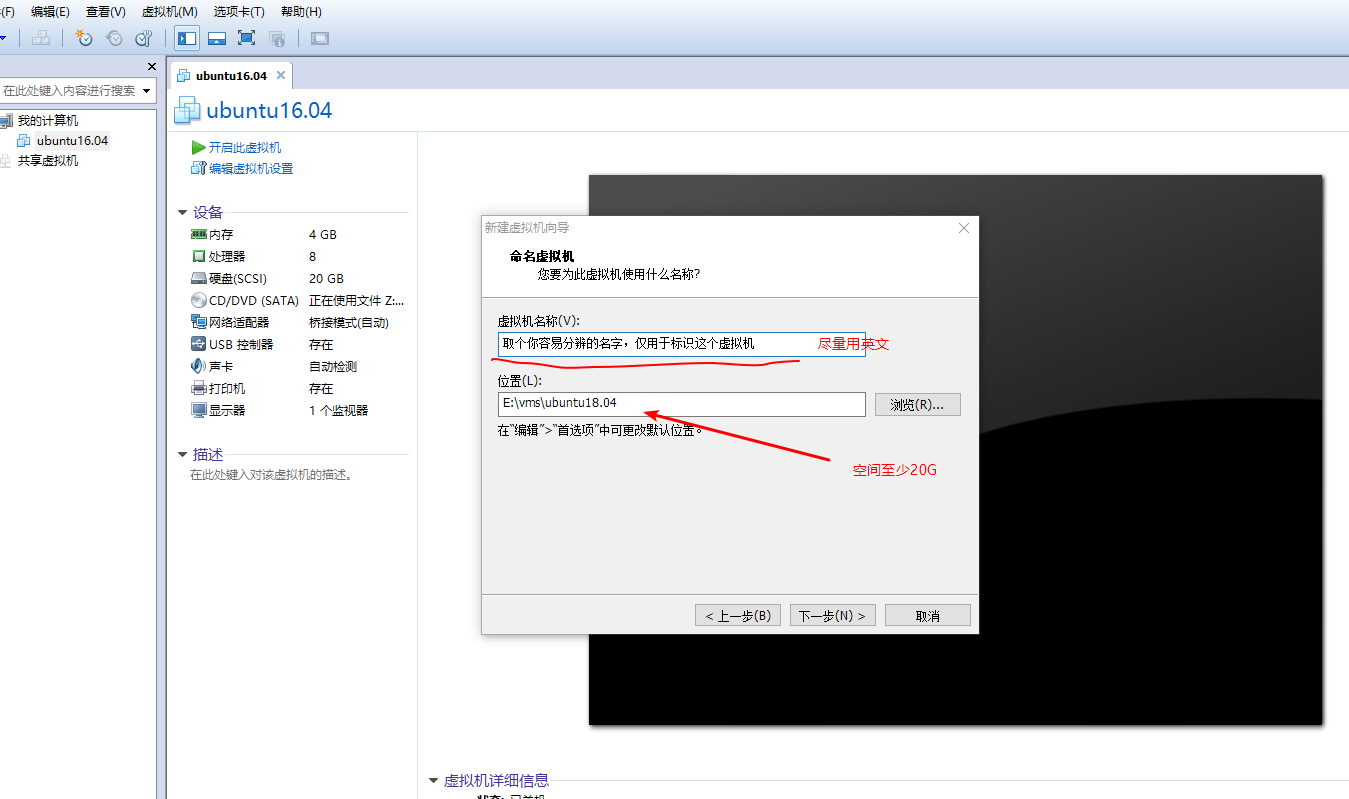


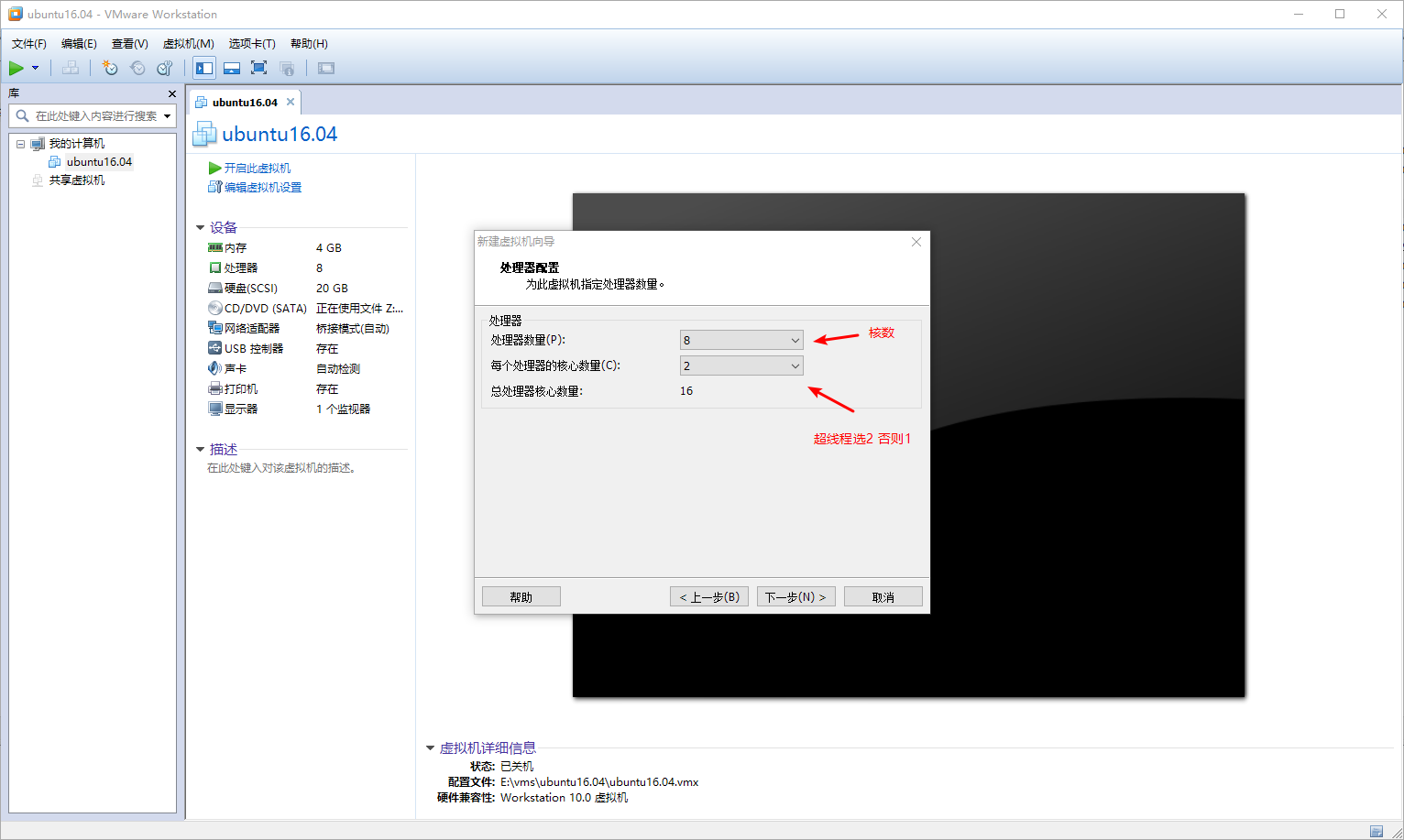


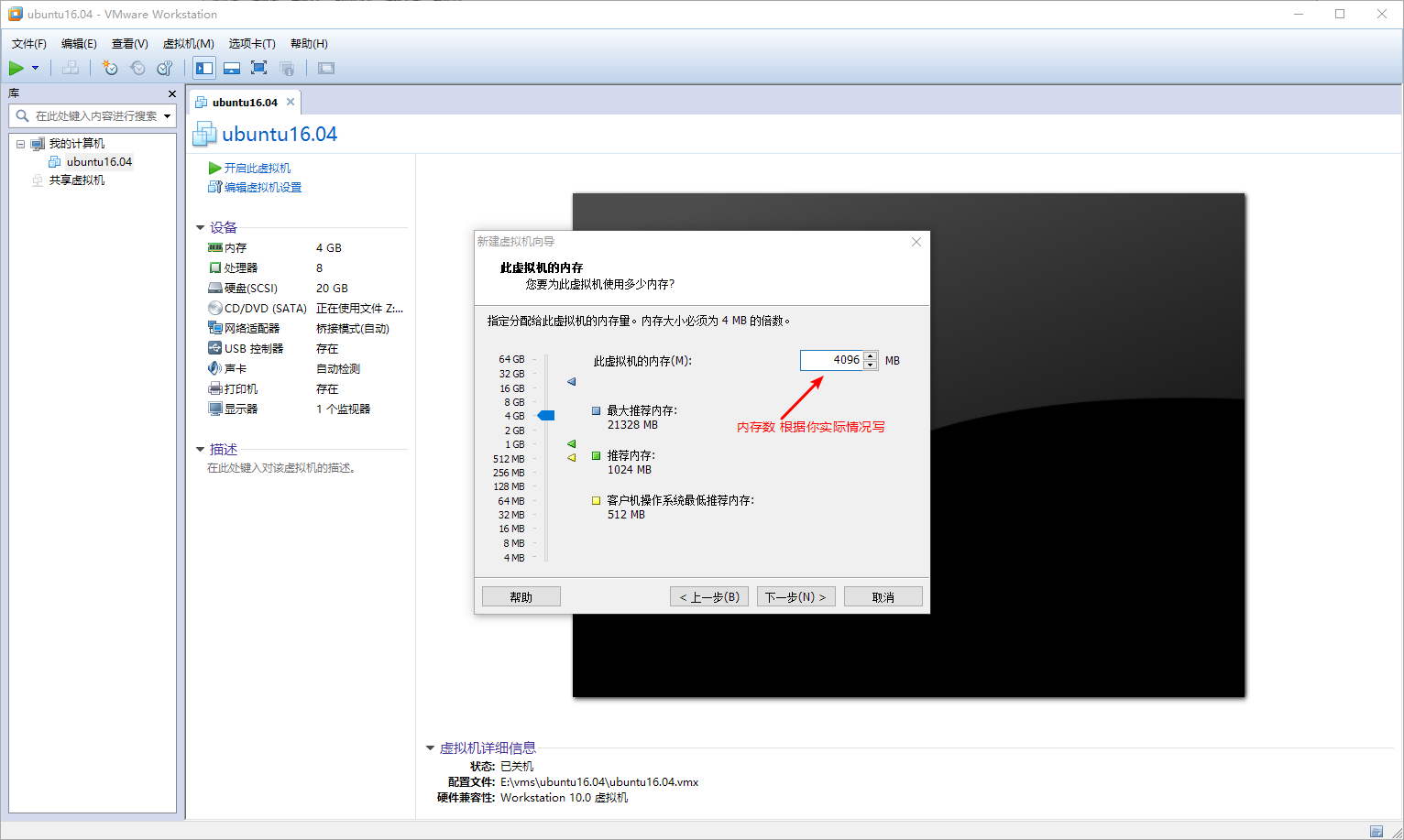


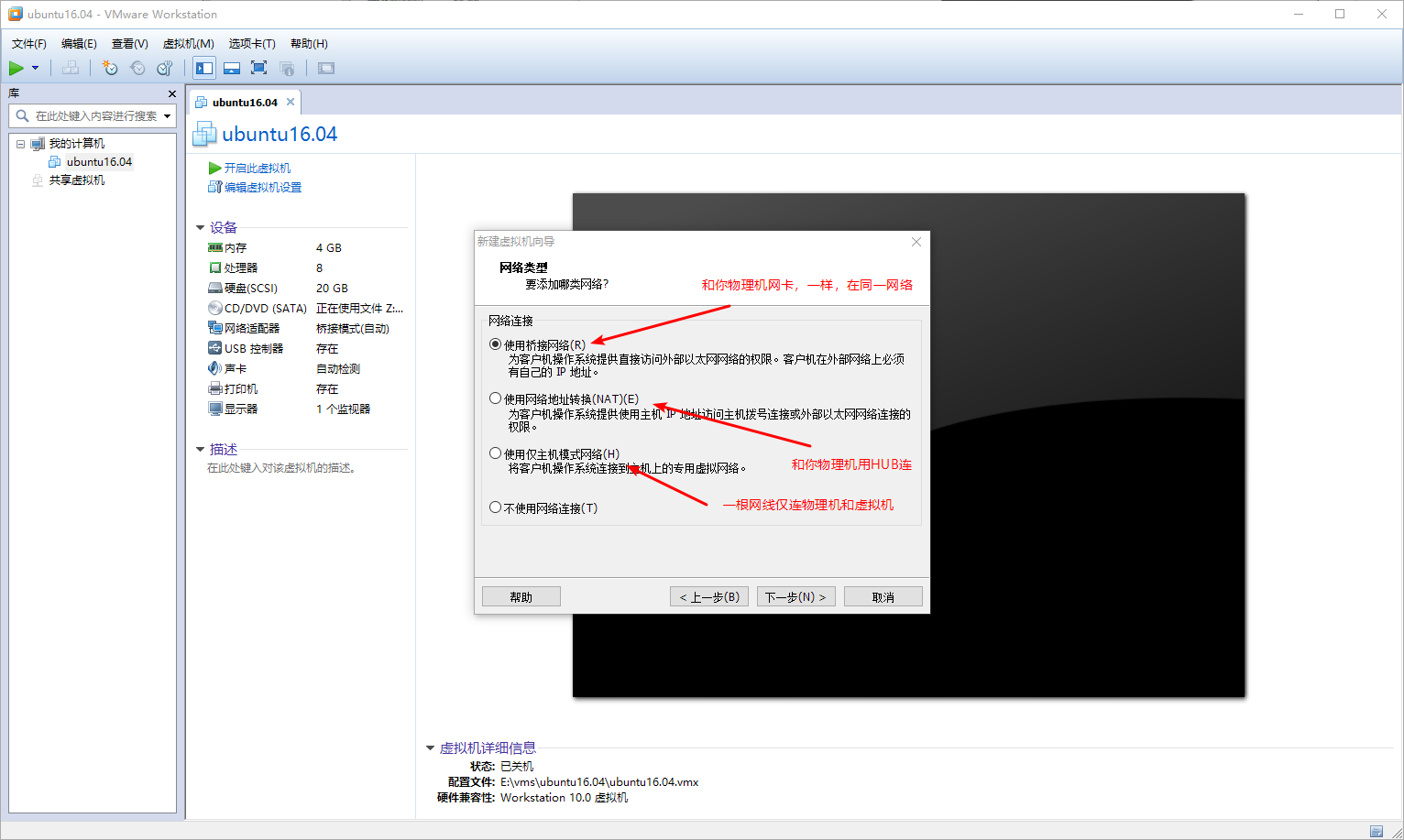


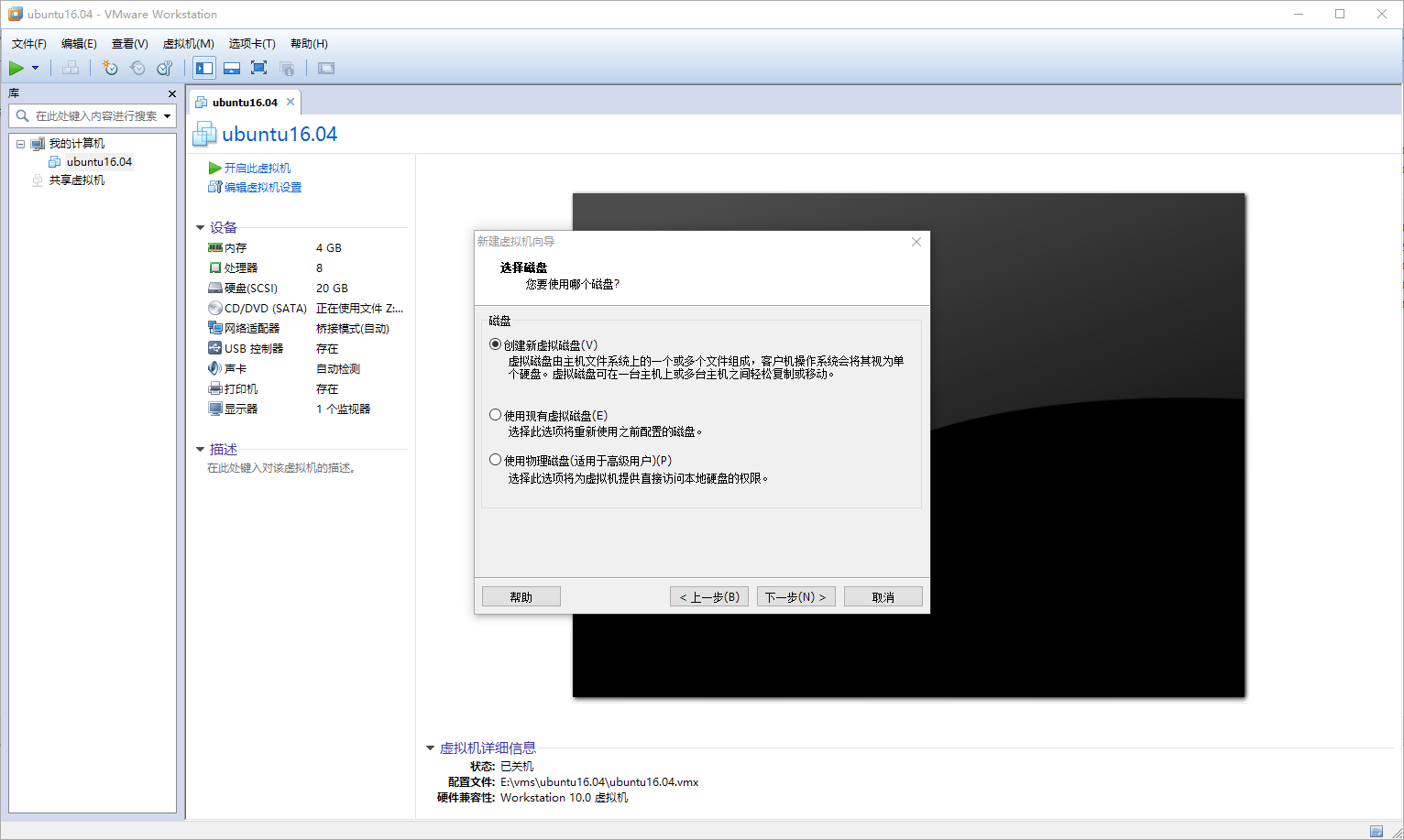


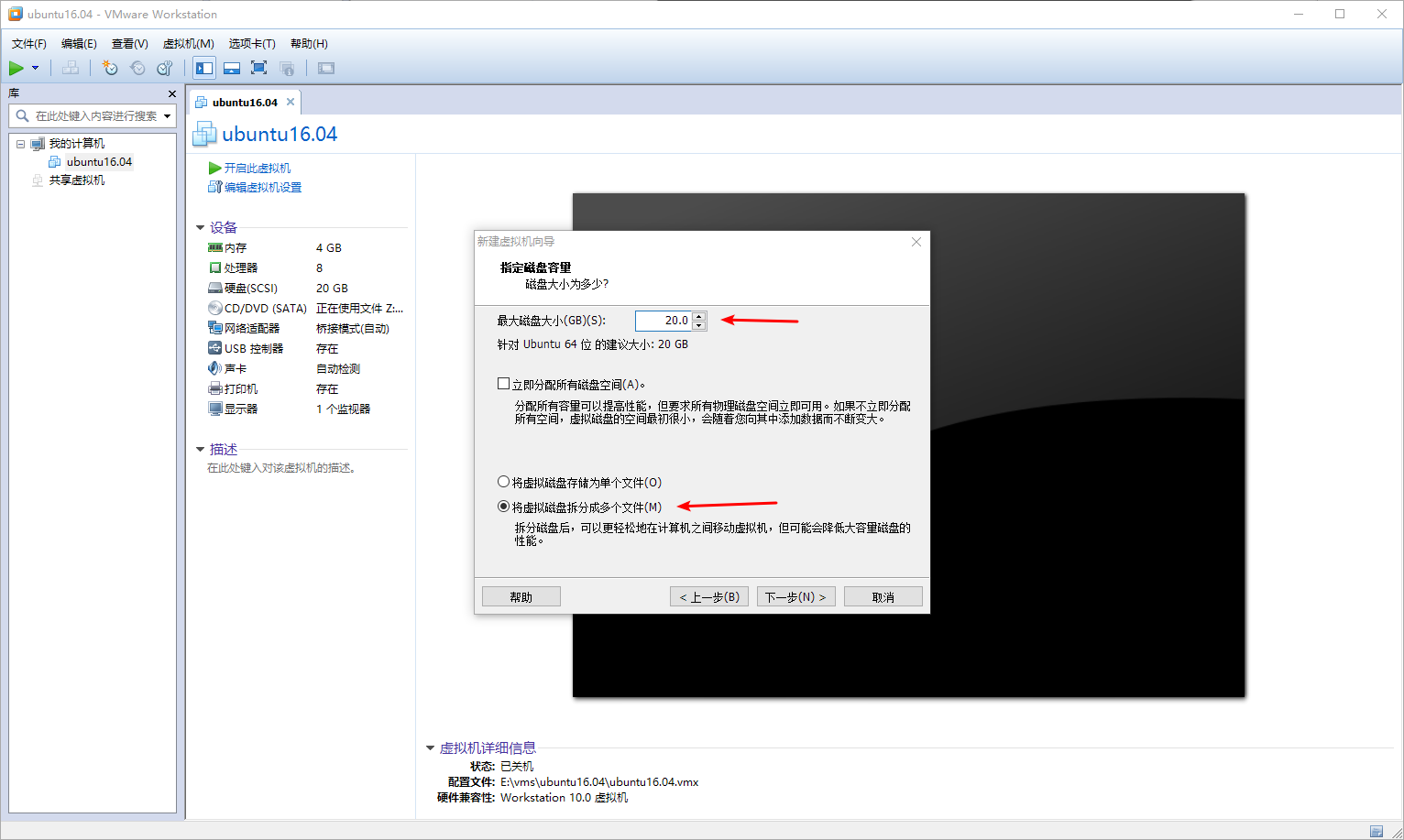


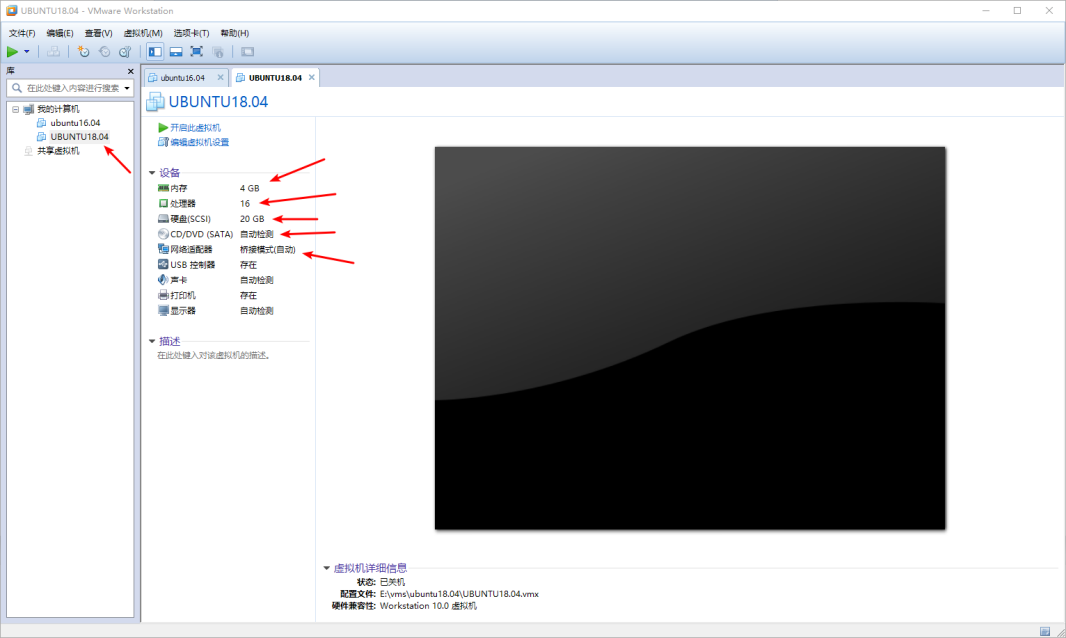


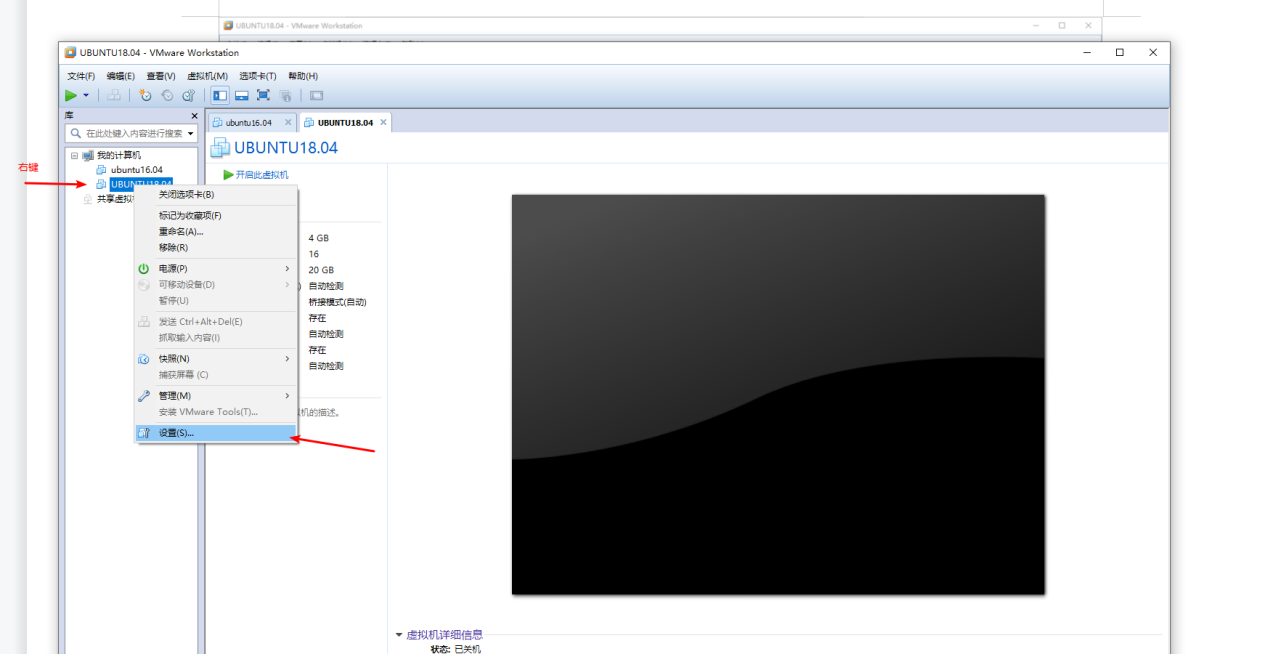


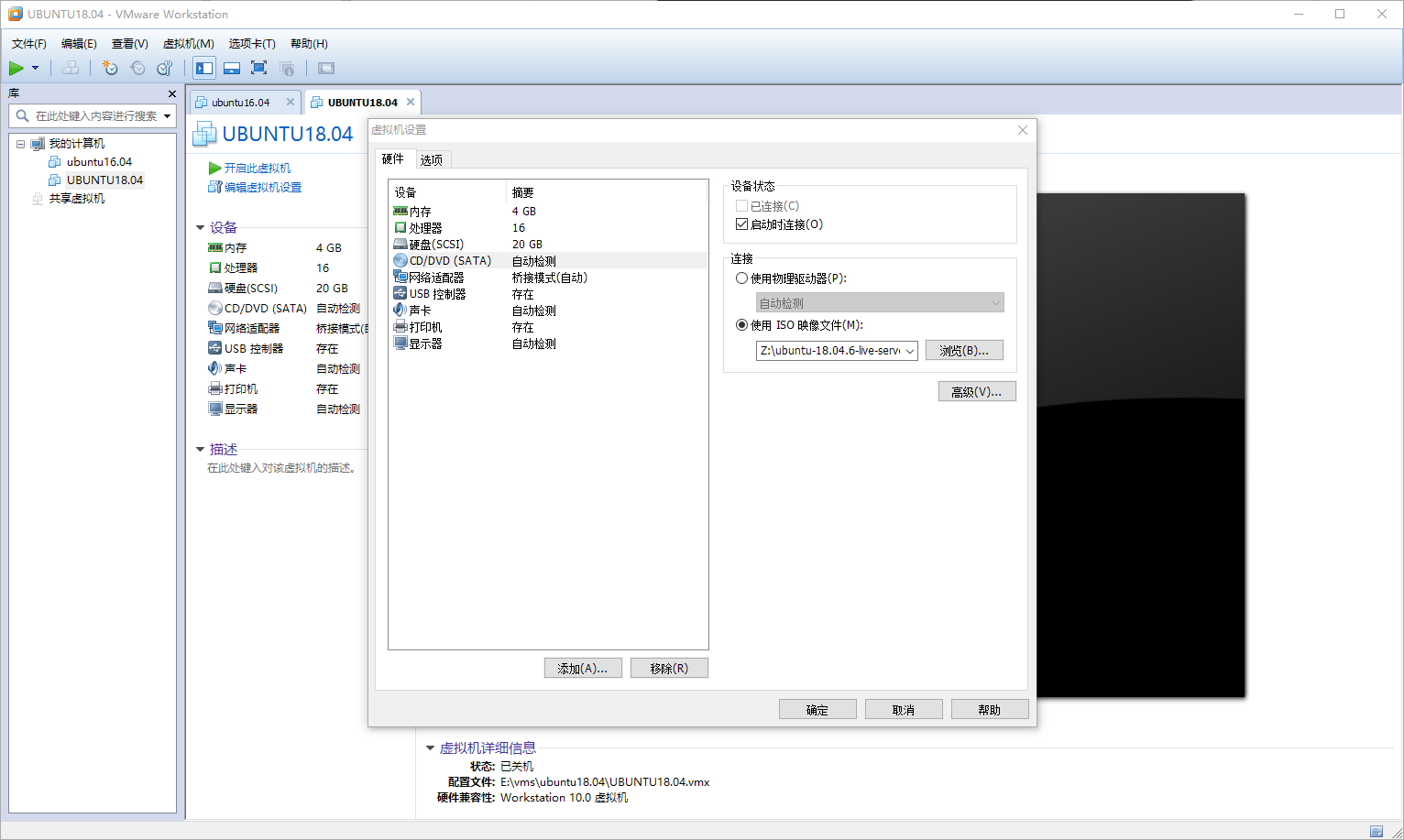




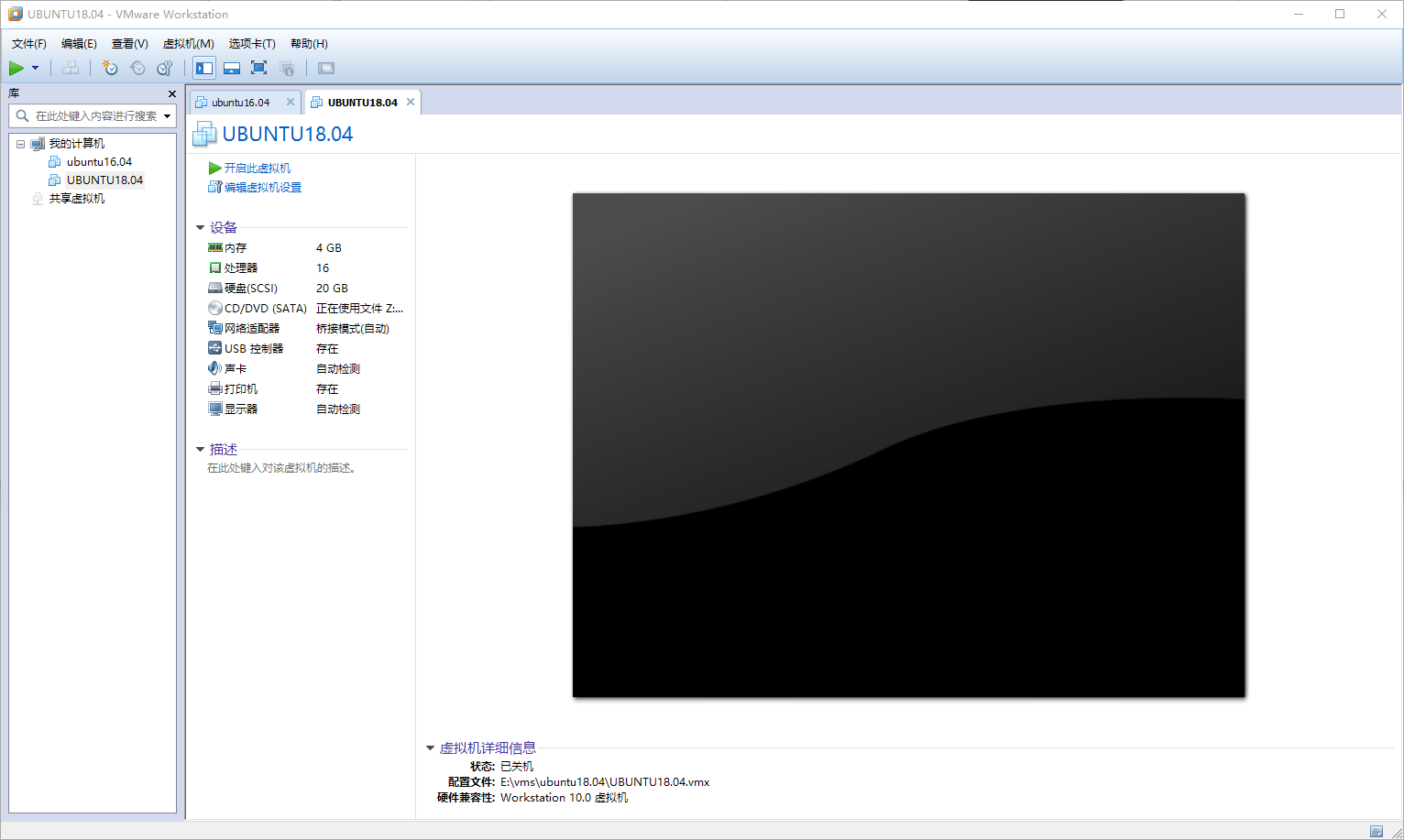








点光驱，选择ISO镜像，这样虚拟机会把这个ISO放到虚拟光驱里。



点击开启虚拟机。

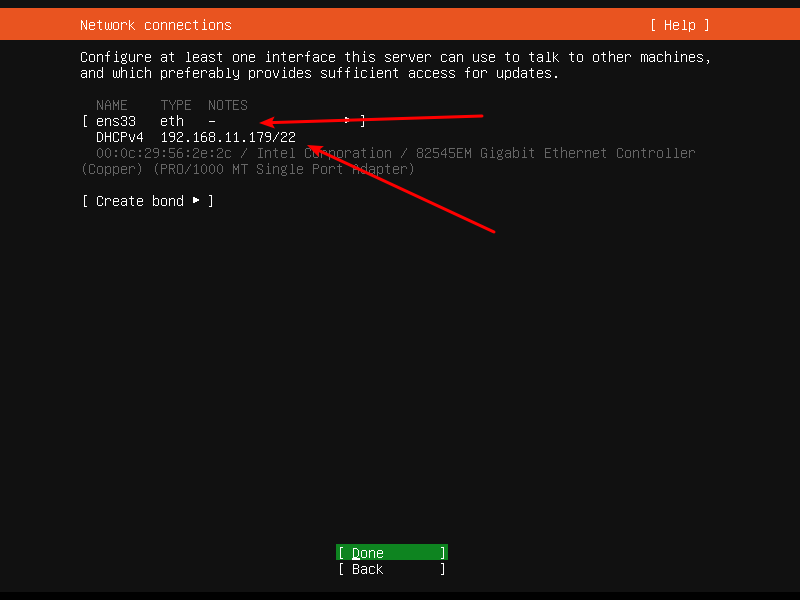


此界面表明进入系统安装

1. 系统安装

接上面最后界面。系统安装 选英语

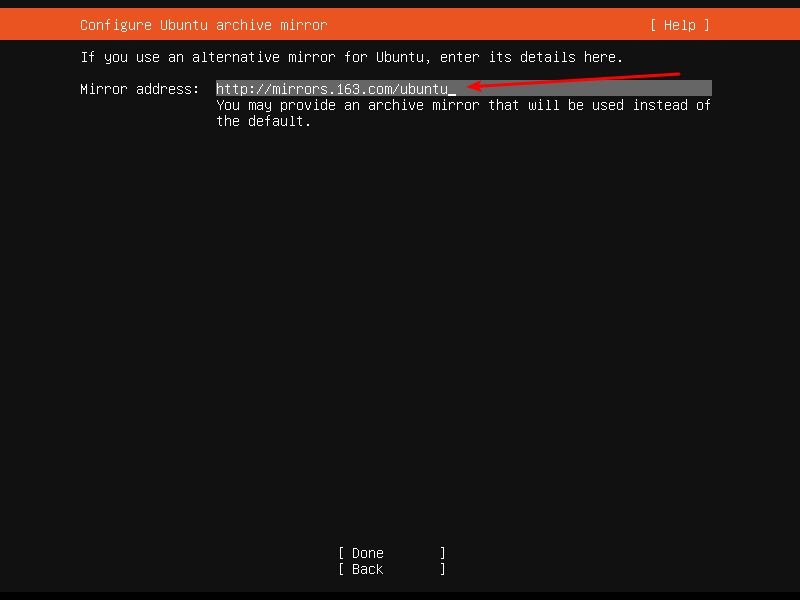
键盘不变 标准美式键盘



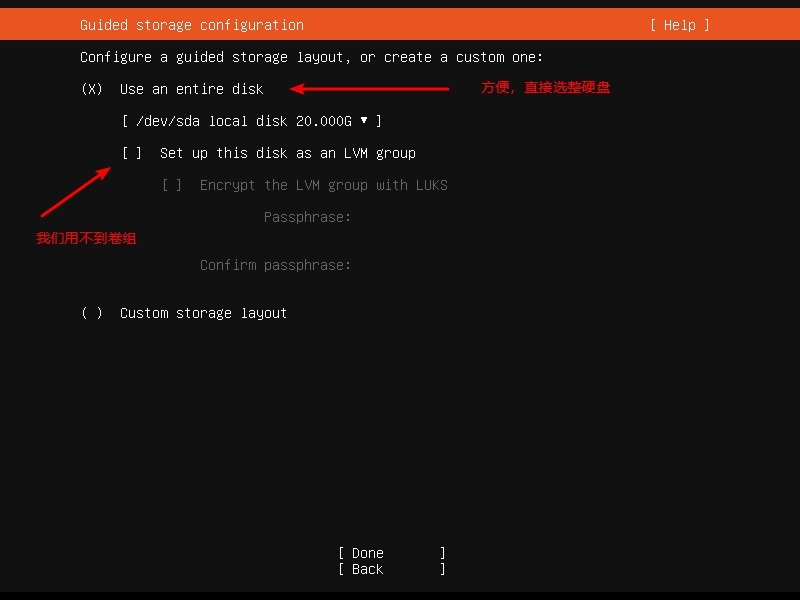
网络，如果你选的是桥接，此处虚拟机和你物理机等于是同一个局域网，如果有DHCP服务器，此处应该已经出IP了。

如果不是，此处请设置静态IP，注意，该静态IP可以和你现在局域网在一个网络，也可以不在。

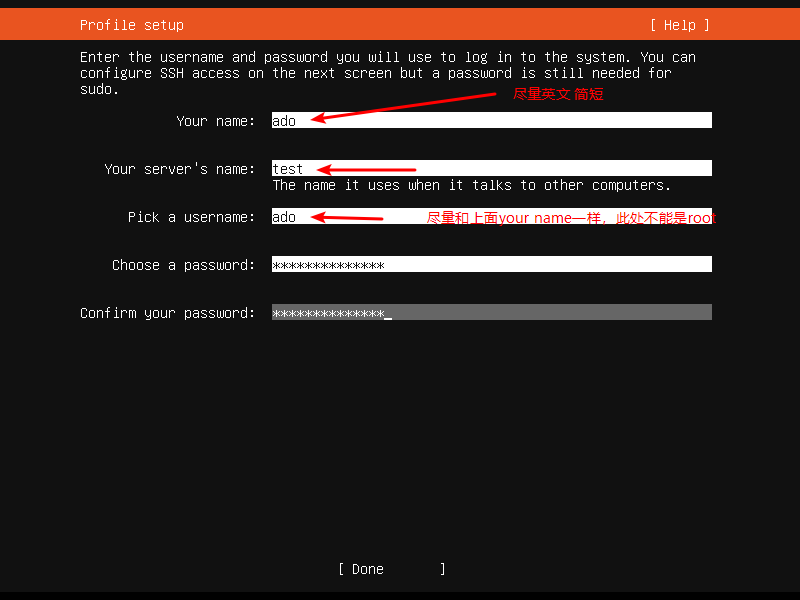
后面很多都是下一步，此处不再多解释，只解释重点部分。

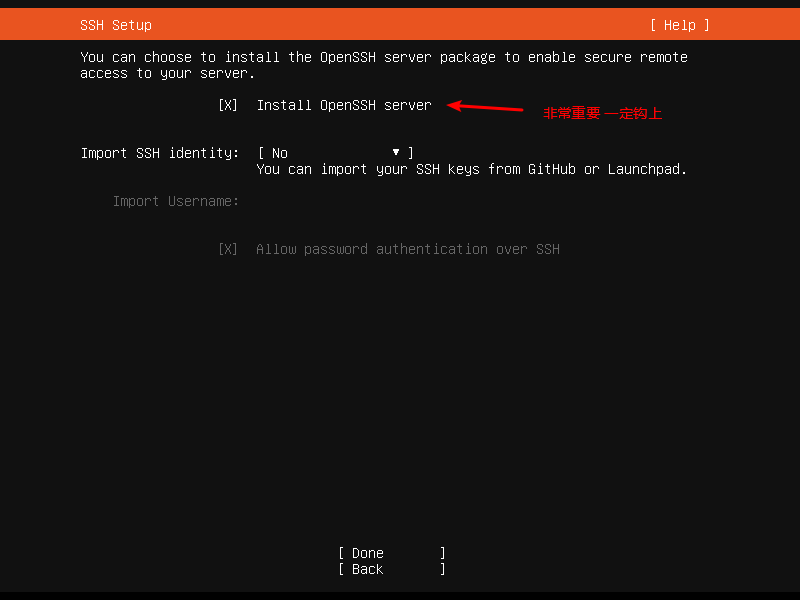


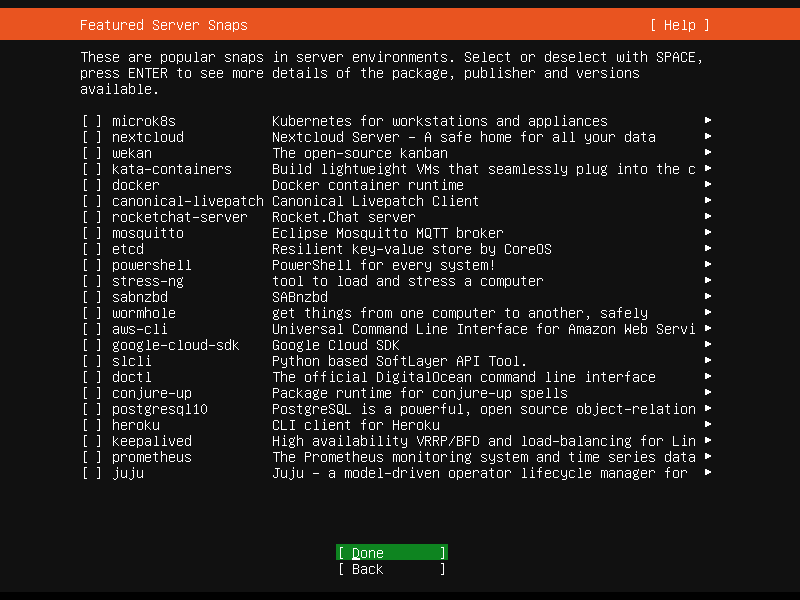
<http://mirrors.163.com/ubuntu>



这里的硬盘是虚拟的，大家不用担心物理机上的文件会不会被删除之类的。

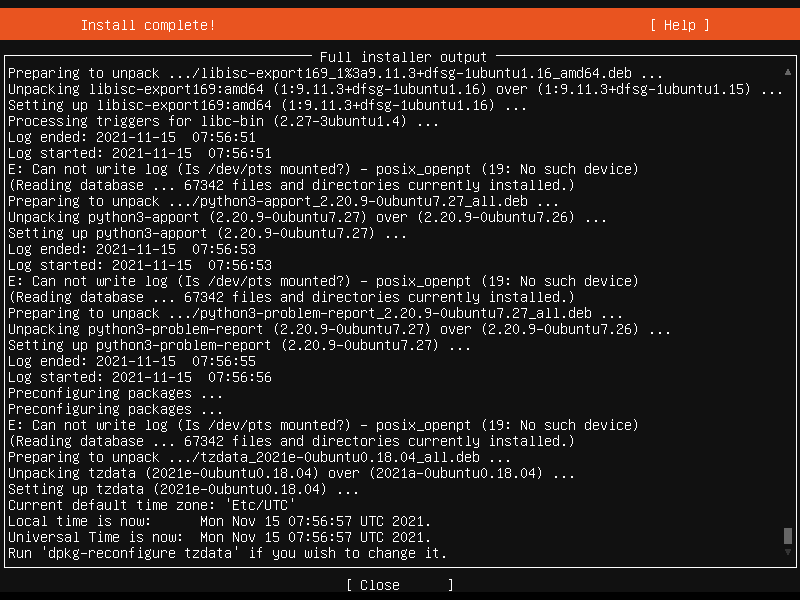


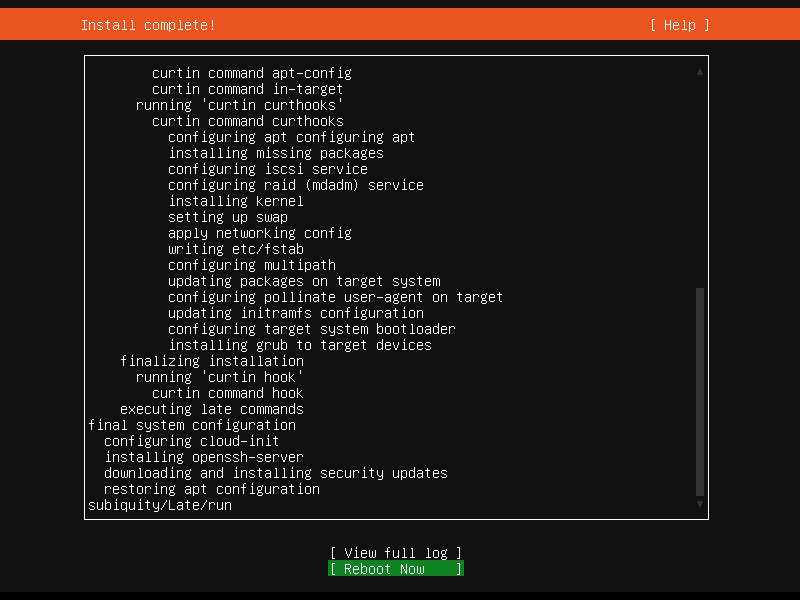




都不选

目前应该已经开始安装了，如果你网络较好，此处会自动更新，请静等安装完成。





1. 系统配置

4.1 终端工具

XSHELL

大家找到官网下教育版（几乎无限制），也可以从以下地址下绿色版：

<https://pan.baidu.com/s/1uA9rVo_1oiZAFePrPUpUGQ> rr8c

XSHELL仅是终端工具，同公司还有产品XFTP，但个人认为不好用。

如果大量文件传输，请下载免费版的FILLZILLA。

4.2 第一次进系统的一些配置。

以ado（上面图中示例用户）用户为例，

进系统之后如下图



执行

sudo passwd root

第一次输入ado密码，用于验证ado用户

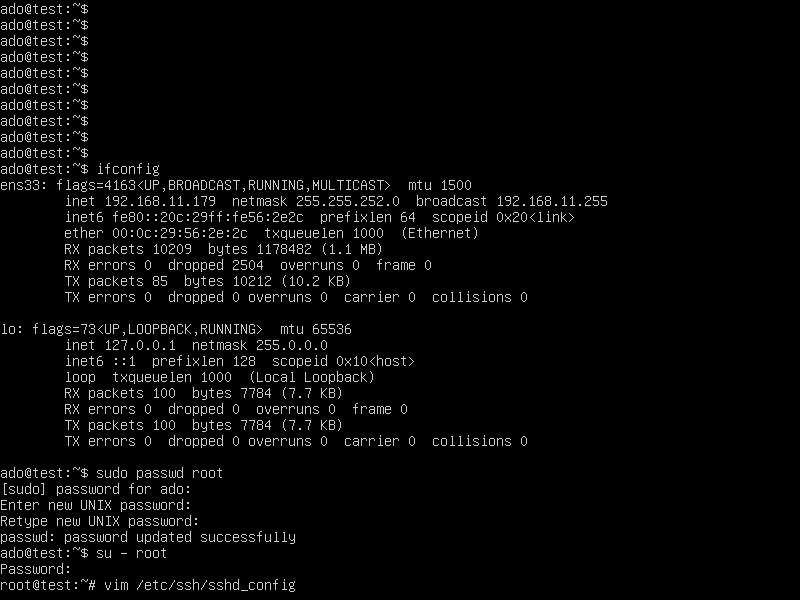
第二次输入root密码

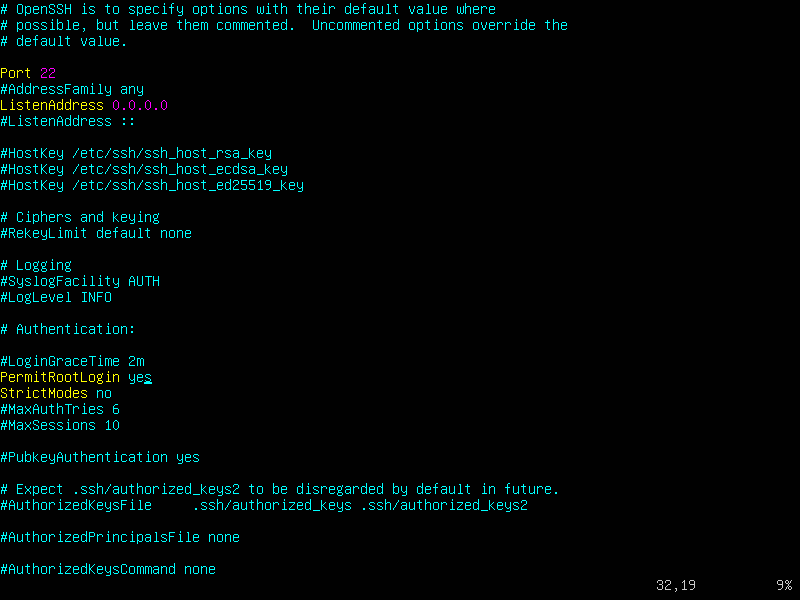
第三次输入root密码 验证

显示成功后，

su - root

用编辑器(vi或nano)打开/etc/ssh/sshd\_config





彩色部分，按我上面的修改。

然后

systemctl restart ssh

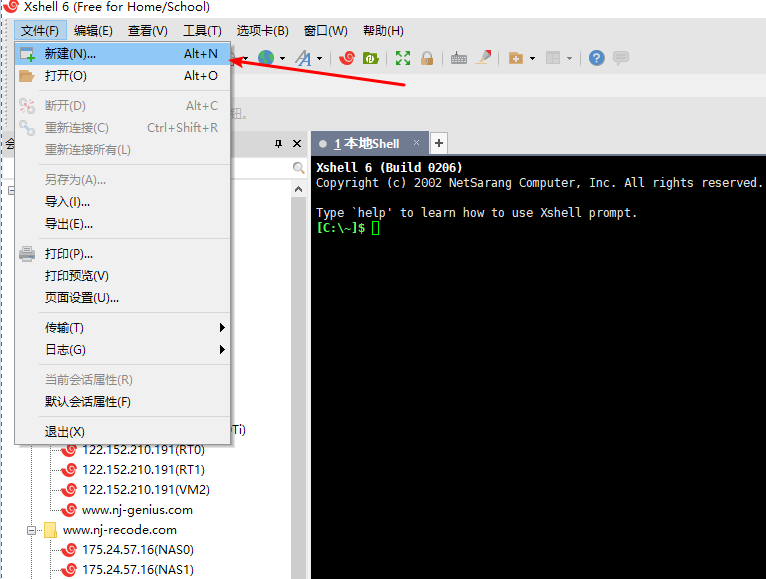
然后

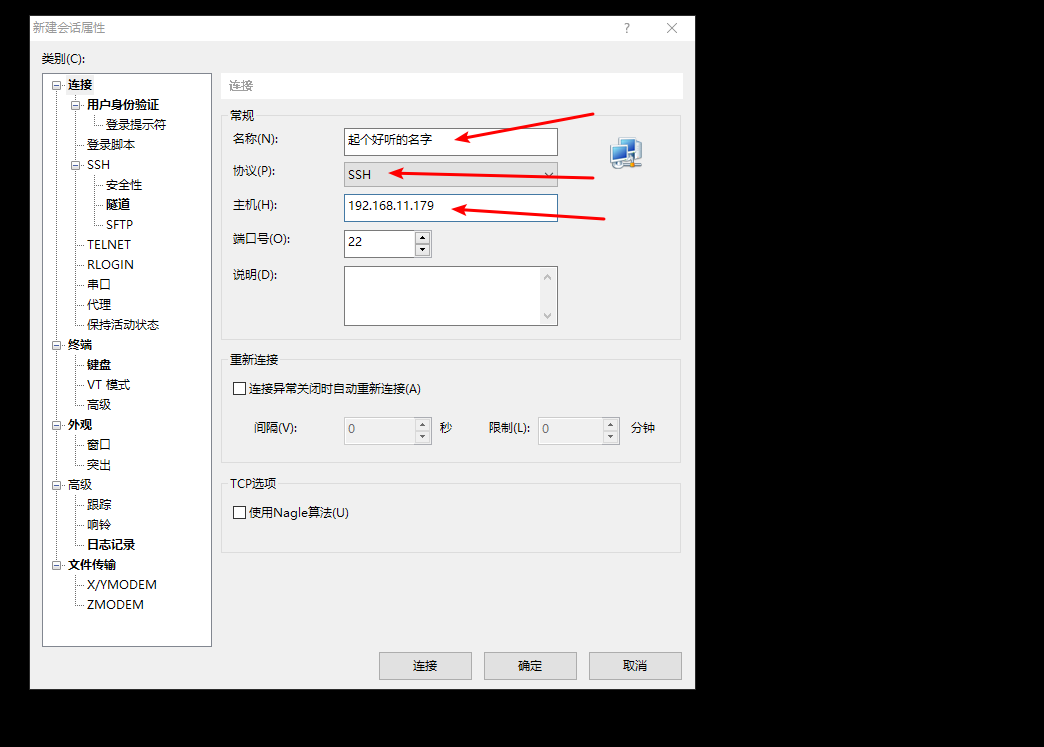
exit 退出root

exit 退出ado

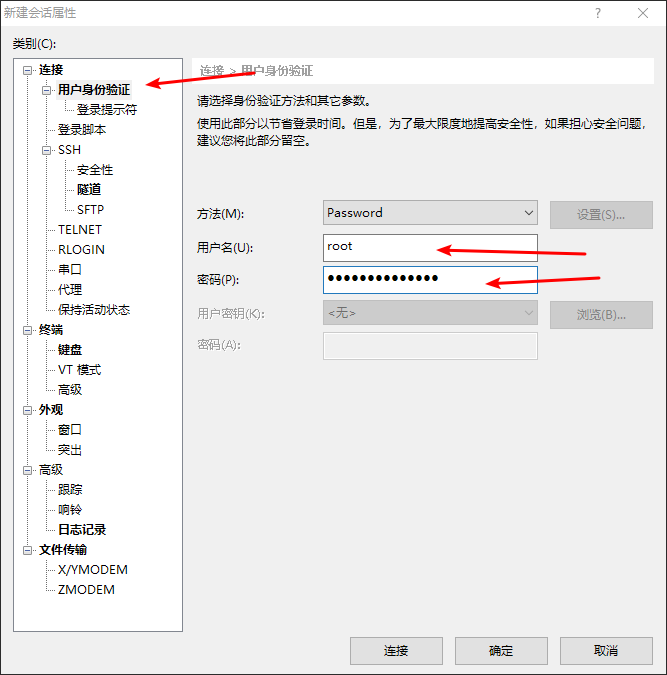
此时，最小化虚拟机。以后用终端和虚拟机交互。

4.3 XSHELL配置



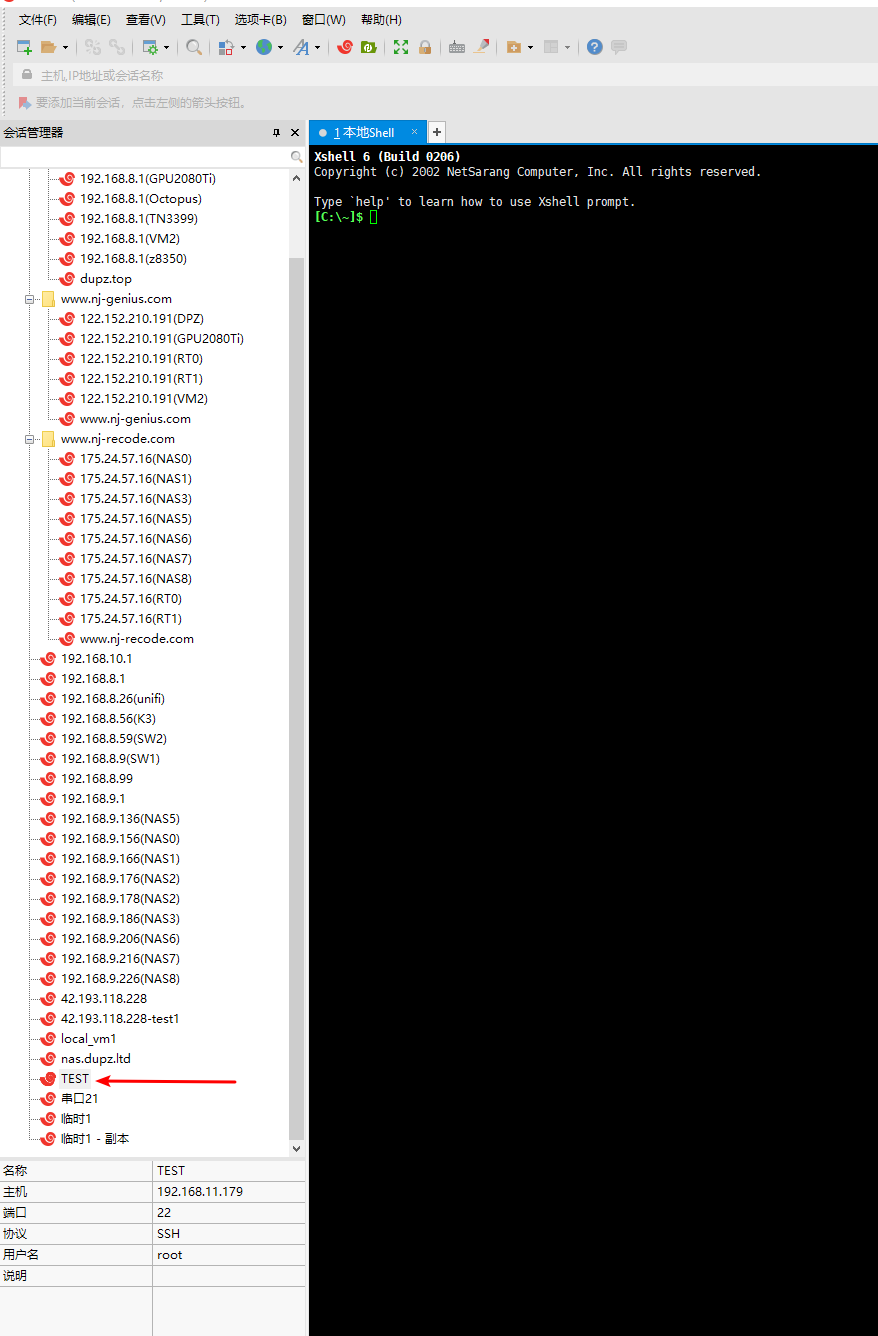


这个IP是刚才ifconfig看到的ip



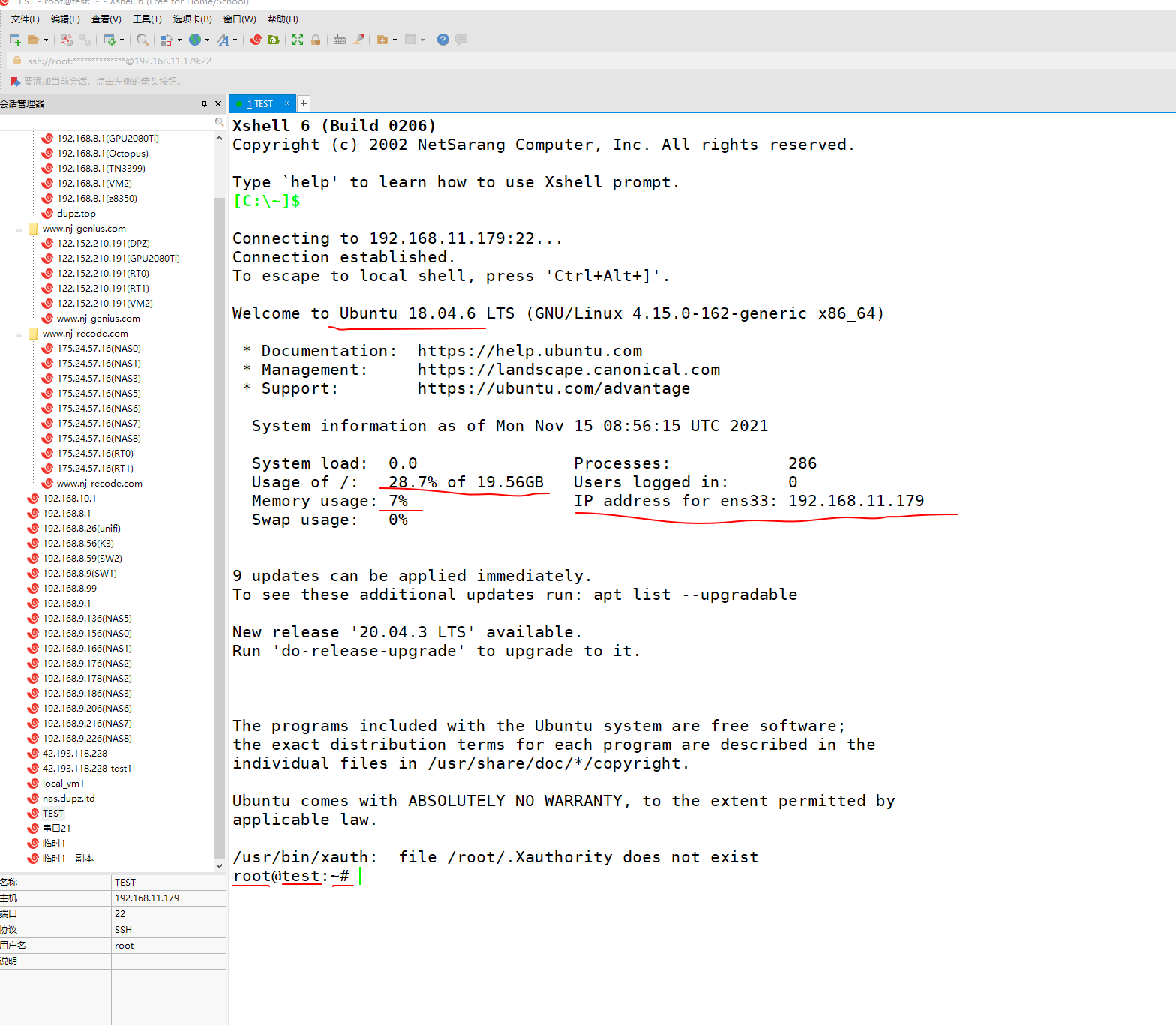
这个用户，我们用root 密码就是你刚才设置的密码

实际上在这个界面，外观可以设置，编码也可以设置，字体大小都可以设置。大家多摸索一下。这里不再解释了。



成功后会出现在这里，双击进入系统

出现弹窗点确认即可



这是登录后的界面。

如下是我个人的一些爱好设置，一些是我们实验必设的，我直接罗列

1. vim 设置 请原封不动的把下面粘贴进去，后面均是这样，不再说明

echo "" >> /etc/vim/vimrc

echo "" >> /etc/vim/vimrc

echo "set nocp" >> /etc/vim/vimrc

echo "set ru" >> /etc/vim/vimrc

echo "set background=light" >> /etc/vim/vimrc

echo "set nobk" >> /etc/vim/vimrc

echo "set is" >> /etc/vim/vimrc

echo "syn on" >> /etc/vim/vimrc

echo "set backspace=indent,eol,start" >> /etc/vim/vimrc

echo "set whichwrap=b,s,<,>,[,]" >> /etc/vim/vimrc

echo "set sw=4" >> /etc/vim/vimrc

echo "set ts=4" >> /etc/vim/vimrc

echo "set lbr" >> /etc/vim/vimrc

echo "set sm" >> /etc/vim/vimrc

echo "set cin" >> /etc/vim/vimrc

echo "set softtabstop=4" >> /etc/vim/vimrc

echo "set autoindent" >> /etc/vim/vimrc

echo "set number" >> /etc/vim/vimrc

echo "set hls" >> /etc/vim/vimrc

echo "set fo+=mB" >> /etc/vim/vimrc

echo "set vb t\_vb=" >> /etc/vim/vimrc

echo "set nobackup" >> /etc/vim/vimrc

echo "" >> /etc/vim/vimrc

2. ssh

systemctl reenable ssh

systemctl restart ssh

3. profile

echo "alias chkconfig=\"systemctl list-units --type=service\"" >> /etc/profile

echo "alias astylec=\"astyle -A1t4xkxVSNwYM80pDHk1W1yxLnvz2\"" >> /etc/profile

echo "alias tailf=\"tail -f\"" >> /etc/profile

echo "ulimit -SHn 1000000" >> /etc/profile

echo "ulimit -SHu 65535" >> /etc/profile

echo "fs.file-max = 1000000" >> /etc/sysctl.conf

echo "\* soft nofile 999999" >> /etc/security/limits.conf

echo "\* hard nofile 1000000" >> /etc/security/limits.conf

echo "\* soft nproc 65535" >> /etc/security/limits.conf

echo "\* hard nproc 65535" >> /etc/security/limits.conf

echo "net.ipv4.ip\_forward=1" >> /etc/sysctl.conf

4. localtime

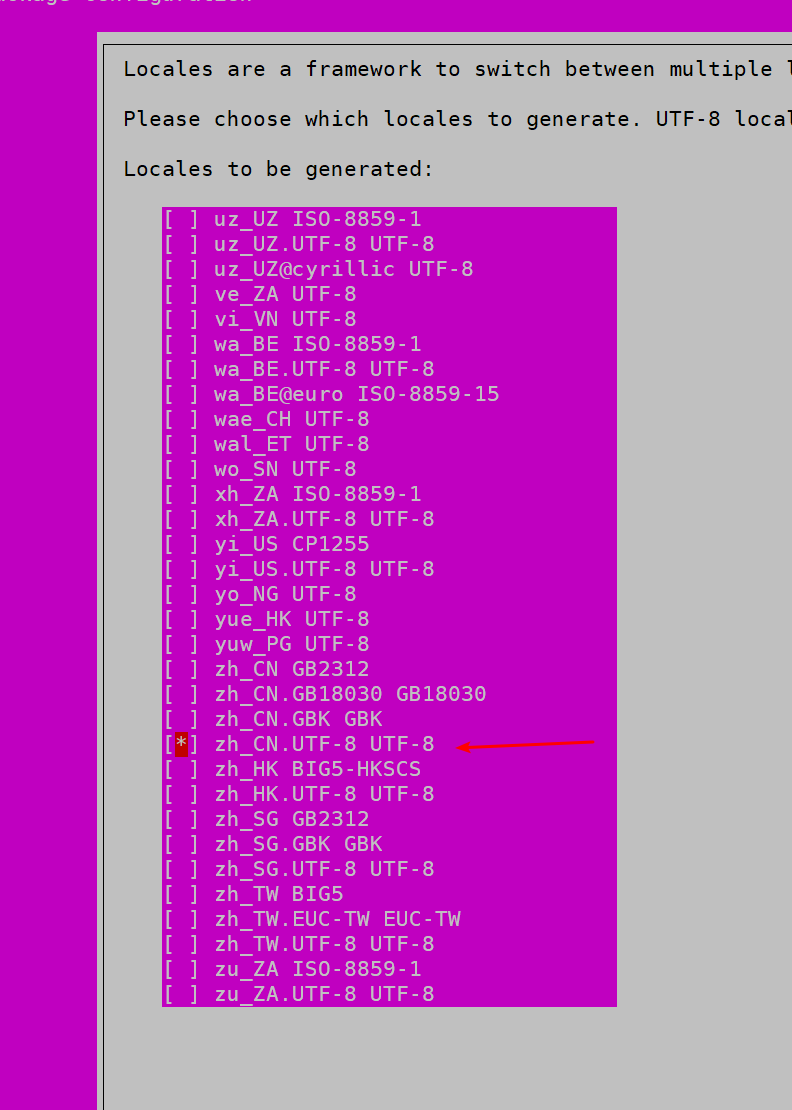
rm /etc/localtime

ln -s /usr/share/zoneinfo/PRC /etc/localtime

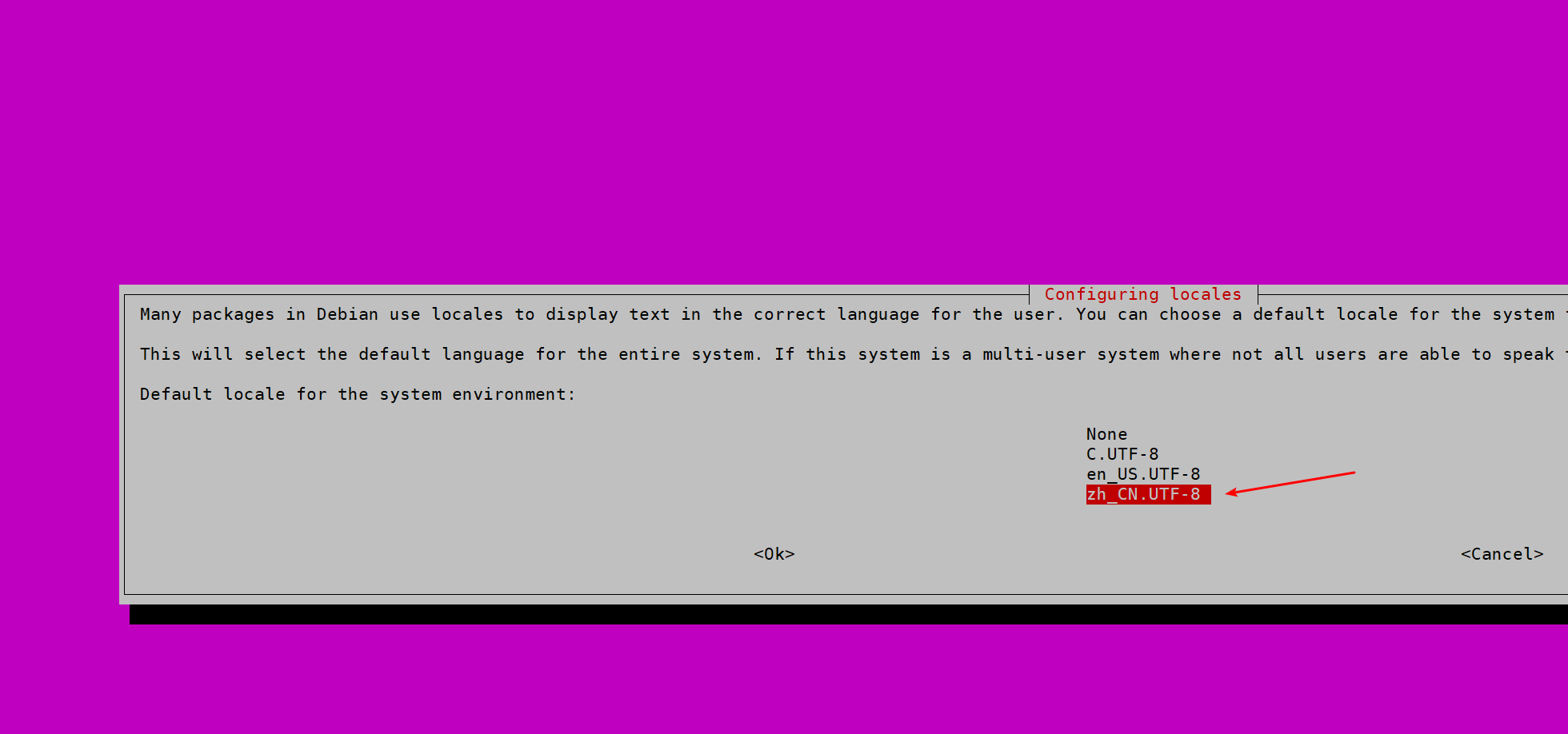
5 zh\_CN

dpkg-reconfigure locales

先执行上面这一句，等执行完



选中这一句，其它不要管



然后直接粘贴下面两句

echo "LANGUAGE=zh\_CN:zh" >> /etc/default/locale

echo "LC\_ALL=zh\_CN.UTF-8" >> /etc/default/locale

6. apt

apt-get update && apt-get upgrade -y && apt-get dist-upgrade -y && apt-get autoremove -y && apt-get clean

该命令时间较长，需要网络

7. 禁用一些不必要的服务

systemctl disable hddtemp.service

systemctl disable ModemManager.service

systemctl disable rpimonitor.service

systemctl disable apparmor.service

systemctl disable apport.service

systemctl disable atd.service

systemctl disable console-setup.service

systemctl disable ebtables.service

systemctl disable lvm2-lvmetad.service

systemctl disable lvm2-monitor.service

systemctl disable lxcfs.service

systemctl disable lxd-containers.service

systemctl disable polkit.service

systemctl disable snapd.seeded.service

systemctl disable snapd.service

systemctl disable ufw

/lib/systemd/systemd-sysv-install disable ufw

systemctl stop ufw

systemctl disable YDService

/lib/systemd/systemd-sysv-install disable YDService

systemctl stop YDService

systemctl disable polkitd

systemctl stop polkitd

systemctl disable postfix

/lib/systemd/systemd-sysv-install disable postfix

systemctl stop postfix

/etc/init.d/apparmor stop

8. 删除无用目录

rm /lost+found /media -rf

9. 安装实验需要的包

apt-get install language-pack-zh-hans apt-utils gcc g++ gdb binutils build-essential make bison flex autoconf automake patch gawk bzip2 gettext pkg-config libtool zlib1g-dev libgmp3-dev libmpfr-dev libmpc-dev libreadline-dev vsftpd lrzsz lftp telnet curl astyle exfat-fuse ethtool sysstat bridge-utils p7zip-full

1. 清理 重启

apt-get clean && reboot

系统重启

1. 获取源码并编译

选一个目录

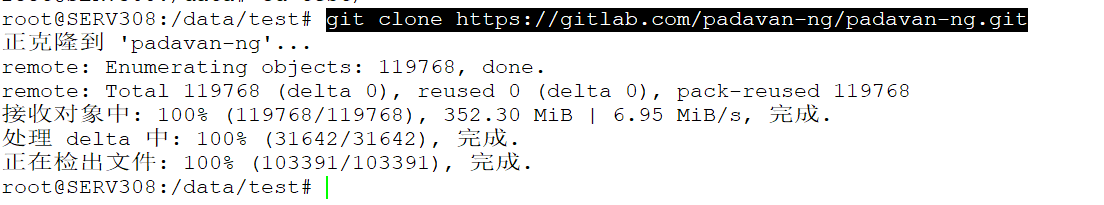
cd /root

mkdir test

以上随意

git clone <https://gitlab.com/padavan-ng/padavan-ng.git>

成功的话，会出现如下界面



然后，编译工具链

我来解释一下工具链。

VCC只是一个IDE，集成开发环境。里面包含了编译器，名字叫cl.exe

LINUX下一般不用IDE，直接用编译器加MAKEFILE，编译器就是鼎鼎大名的gcc

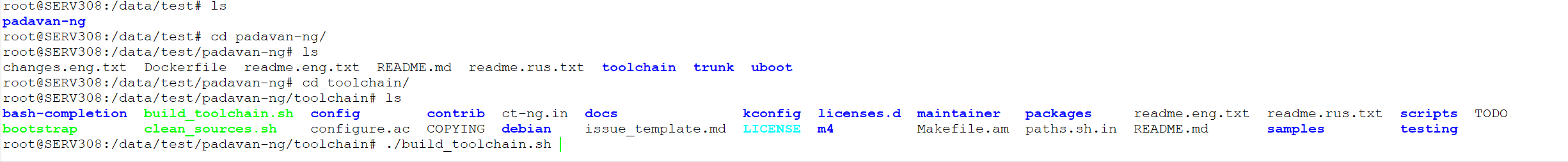
编译器只是一个工具，实质上，我们用这个工具编译出来的东西，可以运行在任意机器上。

一些嵌入式裸机上，没有操作系统，或者本身就要编译操作系统（假设是一个ARM板子）。这时候，我们要借助于另一台机器进行目标为ARM架构的编译。我们用于编译的机器称为开发机，而最终运行我们编译出来的二进制的机器称为目标机。再比如51单片机，根本没有能力有系统，也没有能力自己编译，只有通过开发机，把代码编译为了51单片机可执行的二进制，再下载到51上执行。

这整个过程称为交叉编译。而，编译一个二进制，我们不仅要编译器，还要有链接器，汇编、库，头文件等等。这些东西合起来，就叫交叉工具链。

我们这次实验是编译一个极小的全功能LINUX系统（目标MIPS），并放在MIPS板子上。

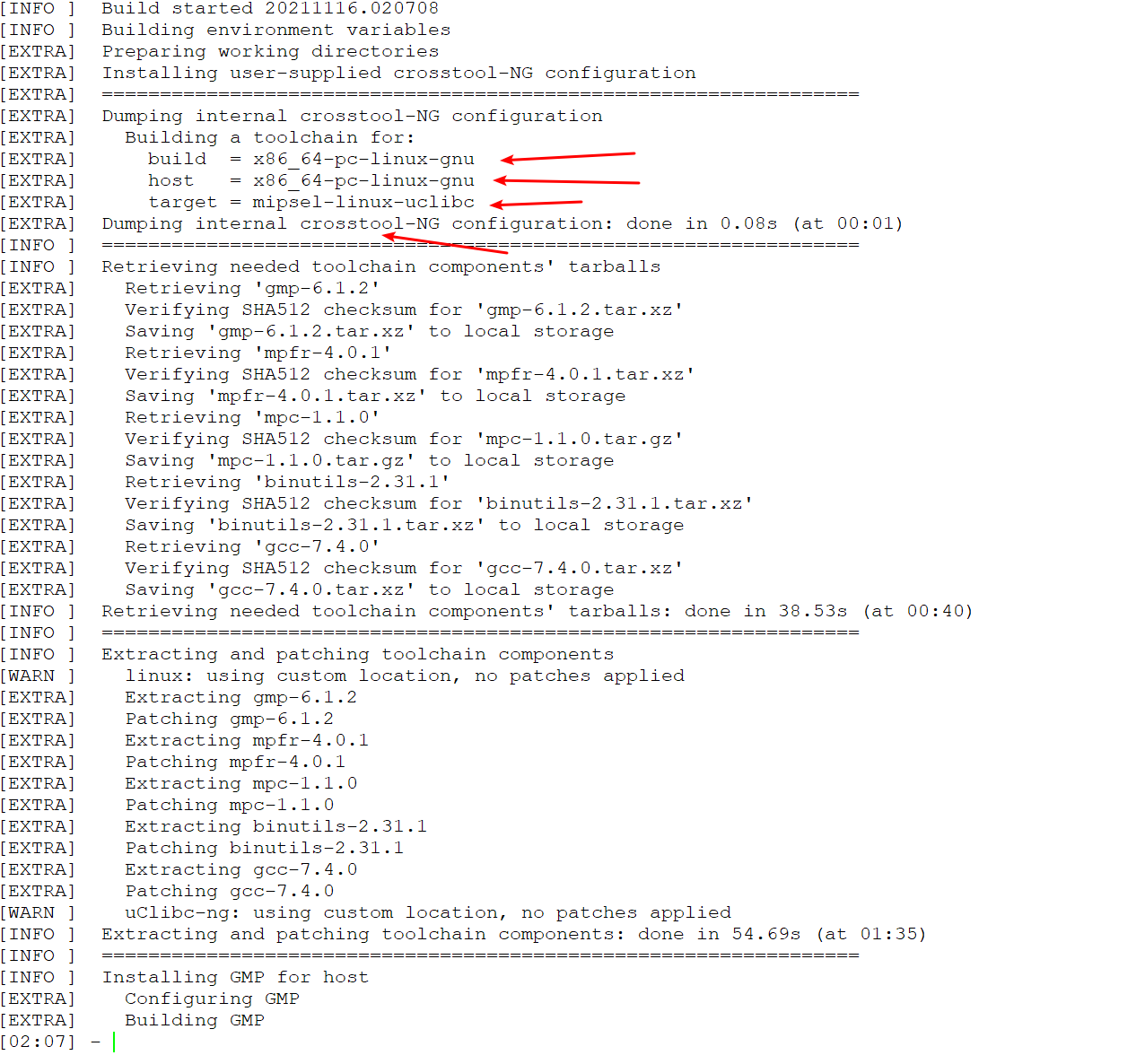
现在机器好了（虚拟机）、系统好了（UBUNTU），代码补丁好了（上面拖下来的），一切具备。我们要开始第一步，编译交叉工具链，如果有些比较强的同学，可以在网上找编译好的工具链，这会省很多时间。



这就开始编译工具链了。有些同学，可能有这方面知识，认为要找补丁，要对应内核版本等等。这个不用有怀疑，给大家找的是一切都OK的。大家按步骤操作即可。我们宗旨并不在于学习如果编译工具链。

这个过程，在一些比较差的机器上，可能要很久，大家有些心理准备。

尽量不要中间中止，可能会引起一些莫名的错误。



这是编译的中间过程。大家可以看下，大体了解下工具链所需要的包。而且上面环境信息说的很清楚了，开发机PC，目标机MIPS（EL是小端的意思），CROSSTOOL则是一个很有名的交叉工具链编译系统。

我机器编译用时18分24秒，大家要有心理准备，这个过程的确时间比较长。

下面我们编译系统。建议同学们看一下编译过程。这个过程略快。



我编译系统大概也用了20分钟。生成系统位于

trunk/images/YK-L1\_3.4.3.9L-100.trx

1. 题目
2. 编译系统，下载并提交，验收的时候，我会找MIPS板子，让大家下载验证。
3. 编译一个内核模块，内容任选，随系统提交，要求内容可体现在系统日志中。
4. 编译一个应用模块，内容任选，以固件形式随系统提交，要求可在自指定文件中有结果显示。

4）

trunk/linux-3.4.x/kernel/sched/core.c 是进程相关的一个核心文件，同学们在必要函数前加注释，画出大体调度流程。

trunk/linux-3.4.x/mm/page\_alloc.c 是内存管理相关的一个核心文件，同学们在必要函数前加注释，画出大体分配流程。

由于文件内容较多，不建议每行加，可以逐函数加注释，最后要生成一个流程图。以上两个文件任选其一。

如果有同学尚有余力，可以对以上文件内容进行修改。

提交或演示内容：  
  
1.编译好的系统，我会烧到指定系统里，可以运行算通过。  
2.内核模块，提交源代码，在运行系统中展示日志行  
3. 应用模块，提交源代码，在运行系统中自行指定展示方式  
4.CORE.C或PAGE\_ALLOC.C带注释提交