

个人资料

个人在知识星球中的ID编号：1536

常用名：Finch

联系方式：QQ： 微信：

目前职业：在职（19应届）

所在地区：上海

熟悉的编程语言：java/python

自我介绍：19应届毕业生，现在上海做javaweb后端，毕业设计题目与网络安全有关，导师也谈论了关于网络安全方向的事情，自己也挺想往web安全方向靠拢，遂加入本群，与大家共同进步，安全相关的技术只了解一点点。希望一年的时间能达到自己的目标。

LNMP环境搭建以及测试的详细过程

总体大的安装步骤：

1. 安装VM虚拟机
 2. 下载centos的镜像文件
 3. 为VM虚拟机安装CentOS操作系统
 4. 为安装好的CentOS操作系统安装图形化界面
 5. 安装nginx等软件前关闭防火墙和SELinux，防止对后面安装的软件有影响
 6. 安装nginx
 7. 安装mysql
 8. 安装php
 9. 配置nginx
 10. 配置mysql【密码，安全策略】
 11. 配置php【写php文件】
 12. 测试能否通过nginx在浏览器中访问到php文件
-

第一步：安装VM虚拟机，由于虚拟机很早就已经安装在本地，而且安装过程也挺容易，就不在此赘述了，找到VM官网，下载稳定版本的VM安装包，然后安装，一步一步默认安装就可以。

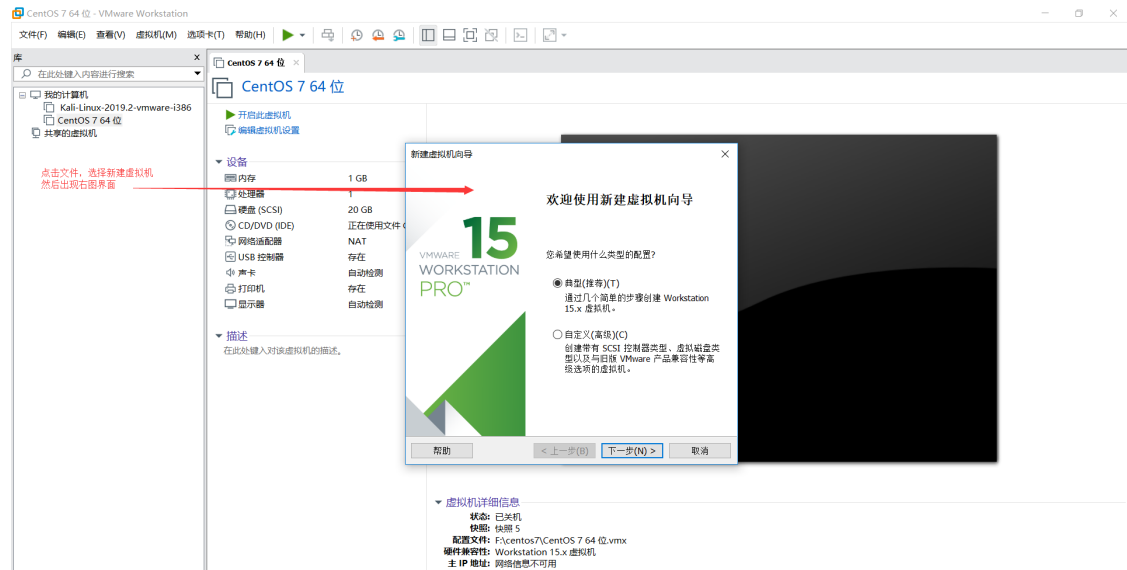
第二步：下载centos7的镜像文件

http://mirrors.aliyun.com/centos/7/isos/x86_64/CentOS-7-x86_64-DVD-1810.iso

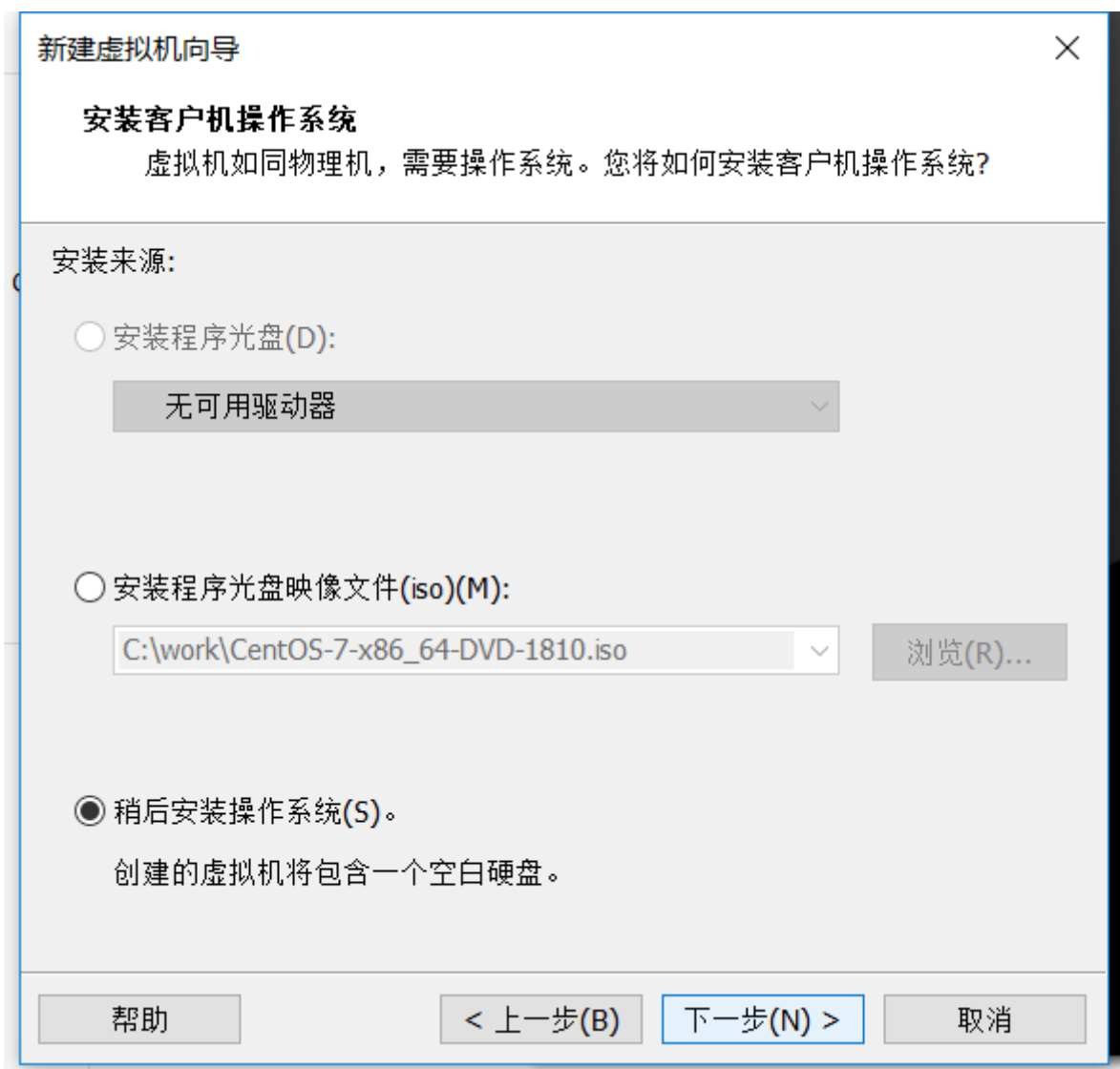
链接地址如上。

第三步：为VM虚拟机安装centos操作系统

1. 打开虚拟机，点击文件，选择**新建虚拟机**，然后出现如图所示的界面



2. 点击下一步，选择稍后安装操作系统，点击下一步



3. 在这一步**选择要安装的Linux操作系统**，版本为**centos 7 64位**

选择客户机操作系统

此虚拟机中将安装哪种操作系统？

客户机操作系统

☐ Microsoft Windows(W)

☒ Linux(L)

☐ VMware ESX(X)

☐ 其他(O)

版本(V)

CentOS 7 64 位

帮助

< 上一步(B)

下一步(N) >

取消

4. 选择好后点击下一步，填写好虚拟机名称（**这个随便填**，主要是你看到这个名字能知道自己当初安装的是什么操作系统就行，不是很重要），**位置是安装这个虚拟机存放的位置，最好选择C盘以外，同时保证这个盘至少有超过20G的磁盘空间**。点击下一步

新建虚拟机向导

×

命名虚拟机

您希望该虚拟机使用什么名称?

虚拟机名称(V):

CentOS 7 new

位置(L):

D:\centos7

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

5. 这一步默认点击**下一步**

新建虚拟机向导

×

指定磁盘容量

磁盘大小为多少?

虚拟机的硬盘作为一个或多个文件存储在主机的物理磁盘中。这些文件最初很小，随着您向虚拟机中添加应用程序、文件和数据而逐渐变大。

最大磁盘大小 (GB)(S):

针对 CentOS 7 64 位 的建议大小: 20 GB

☐ 将虚拟磁盘存储为单个文件(O)

☒ 将虚拟磁盘拆分成多个文件(M)

拆分磁盘后，可以更轻松地在计算机之间移动虚拟机，但可能会降低大容量磁盘的性能。

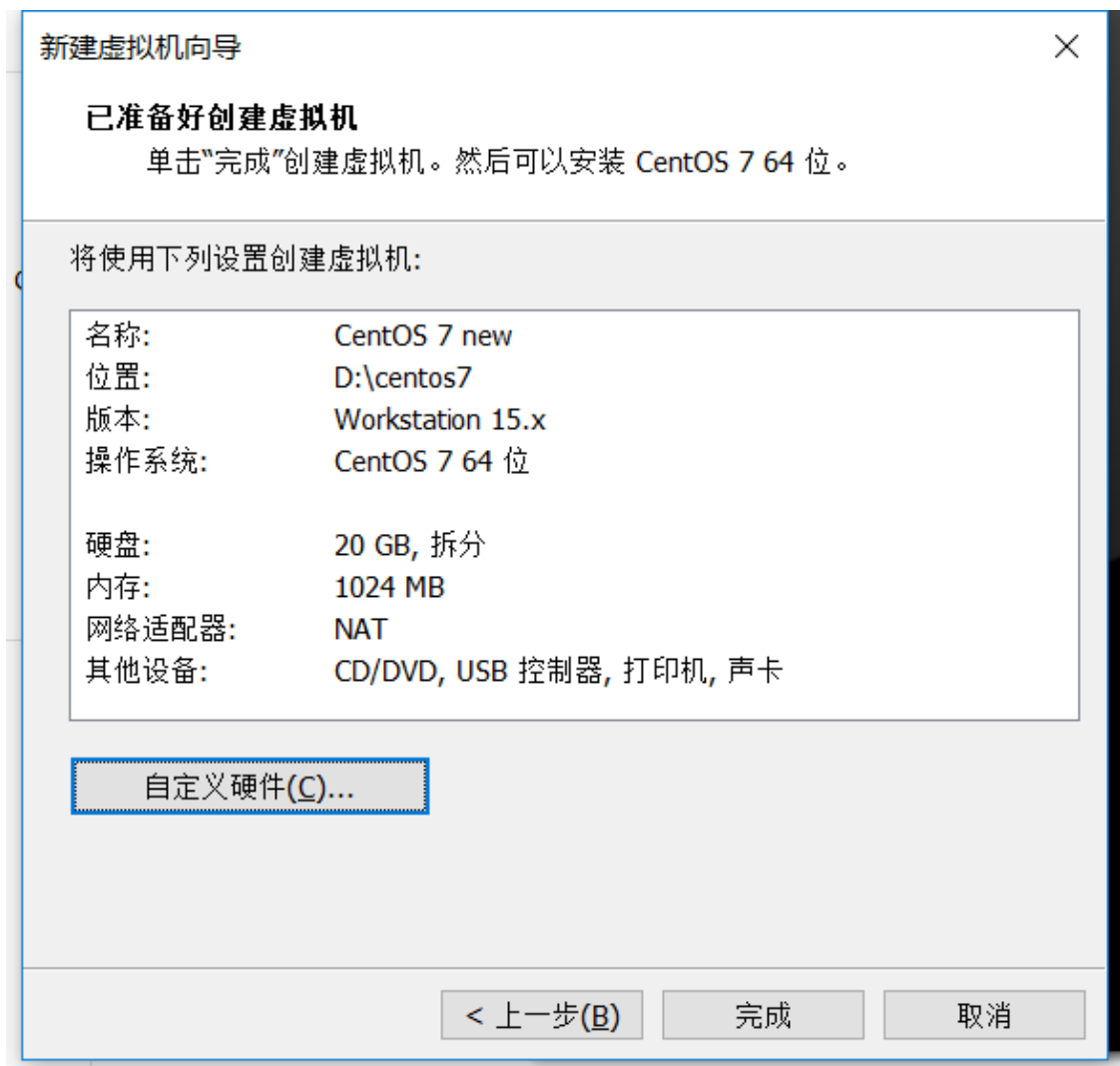
帮助

< 上一步(B)

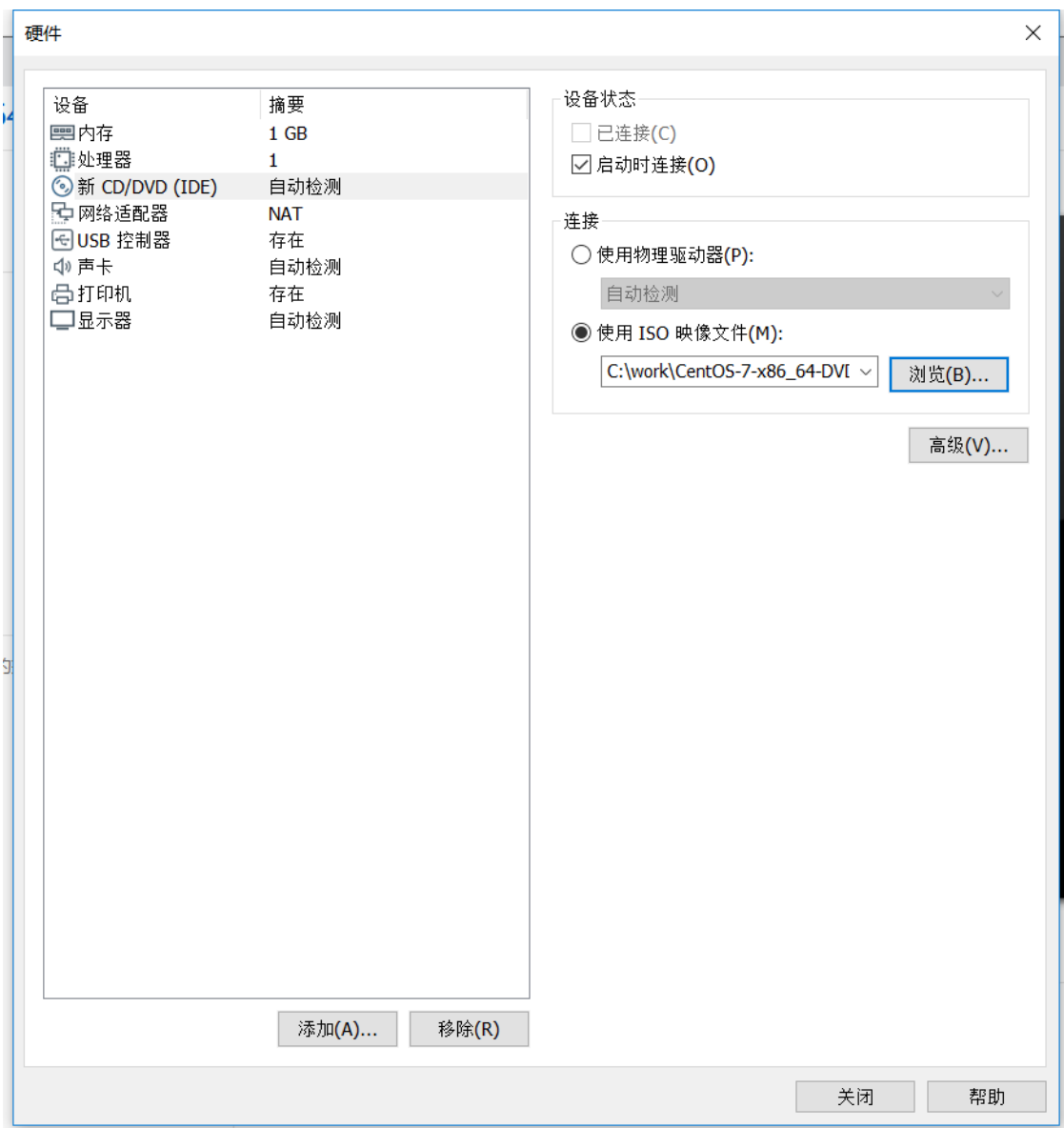
下一步(N) >

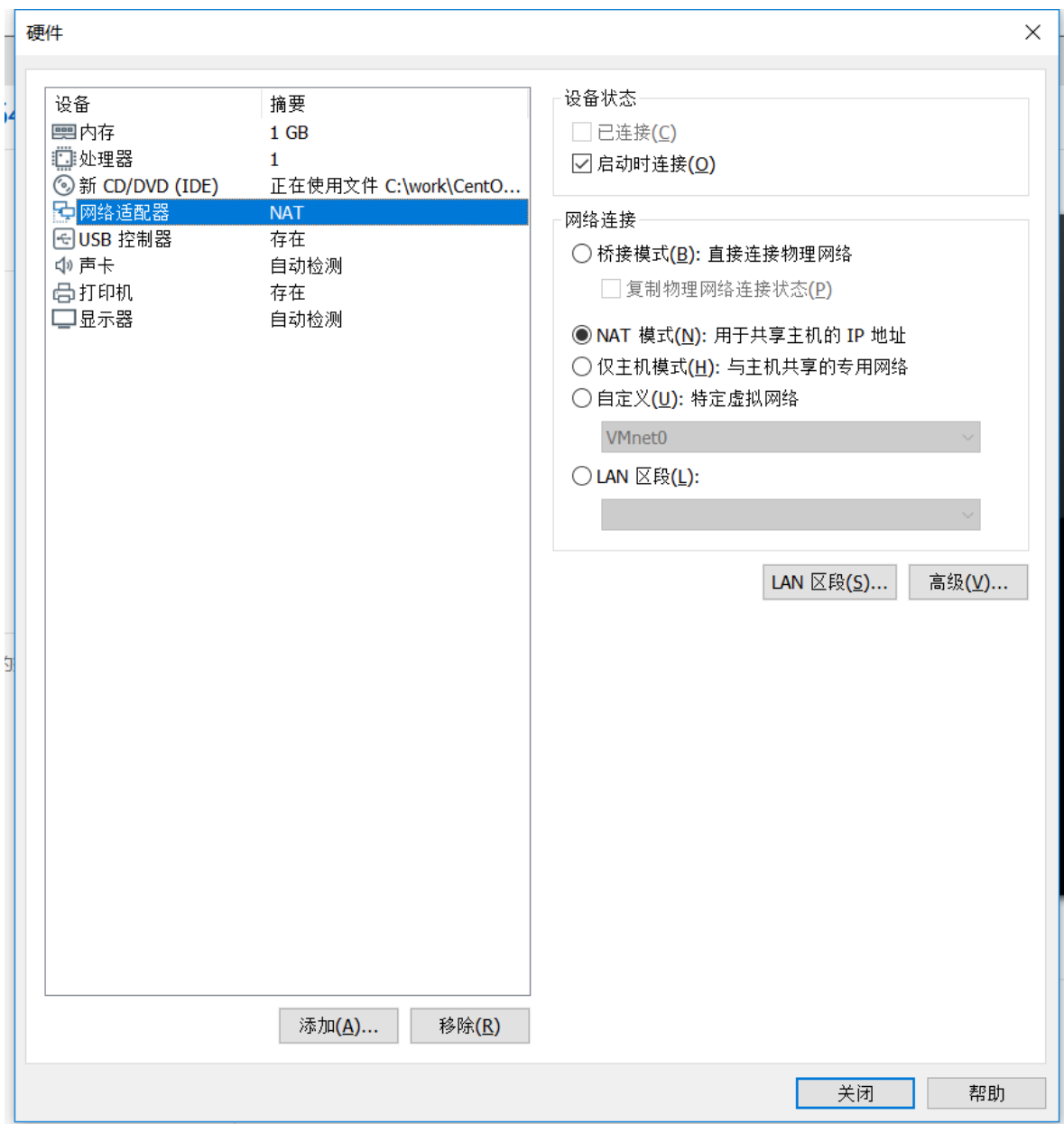
取消

6. 这一步[点击自定义硬件](#)

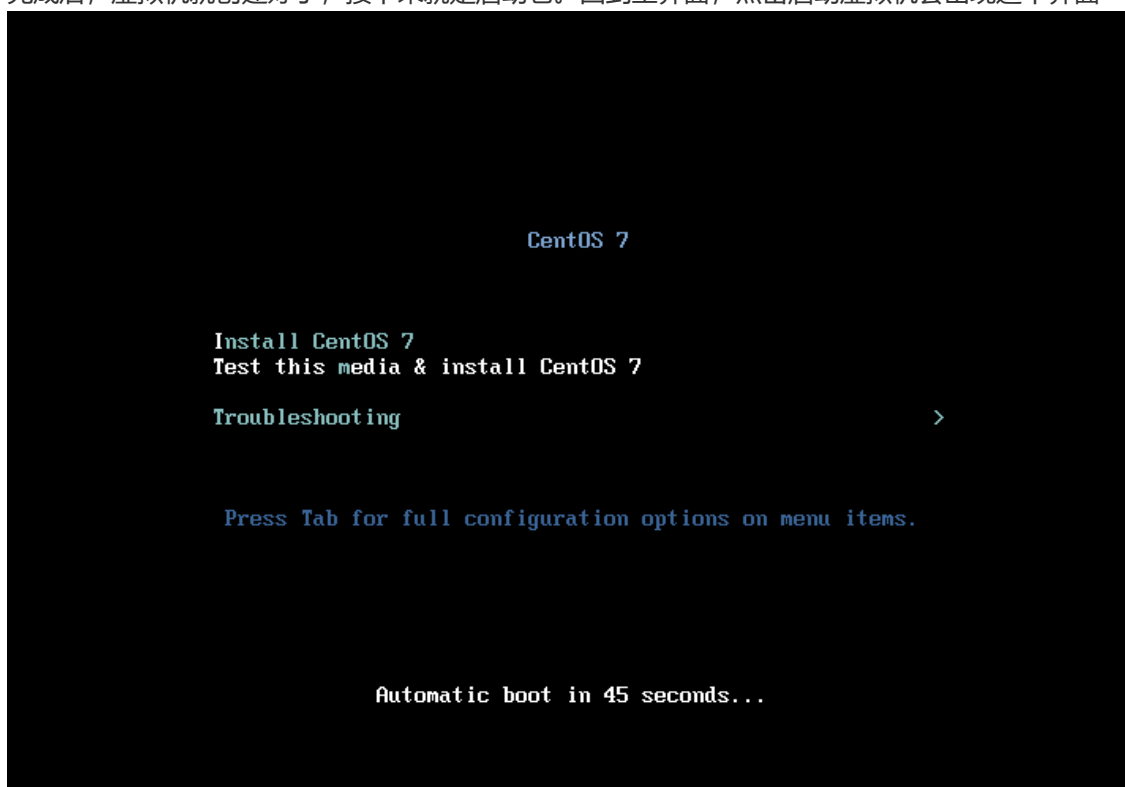


7. 首先点击 **新CD/DVD**，使用ISO映像文件（就是刚才下载的centos镜像文件），然后**选择网络适配器**，这里我选择的是NAT模式，有其他教程用的是桥接模式，但是安装到我的电脑上，虚拟机最后无法访问网络，可能是其他什么地方没有配置好，**新手就选择NAT模式**。这些都修改好了就可以点击关闭，然后点击完成了。





8. 完成后，虚拟机就创建好了，接下来就是启动它。回到主界面，点击启动虚拟机会出现这个界面



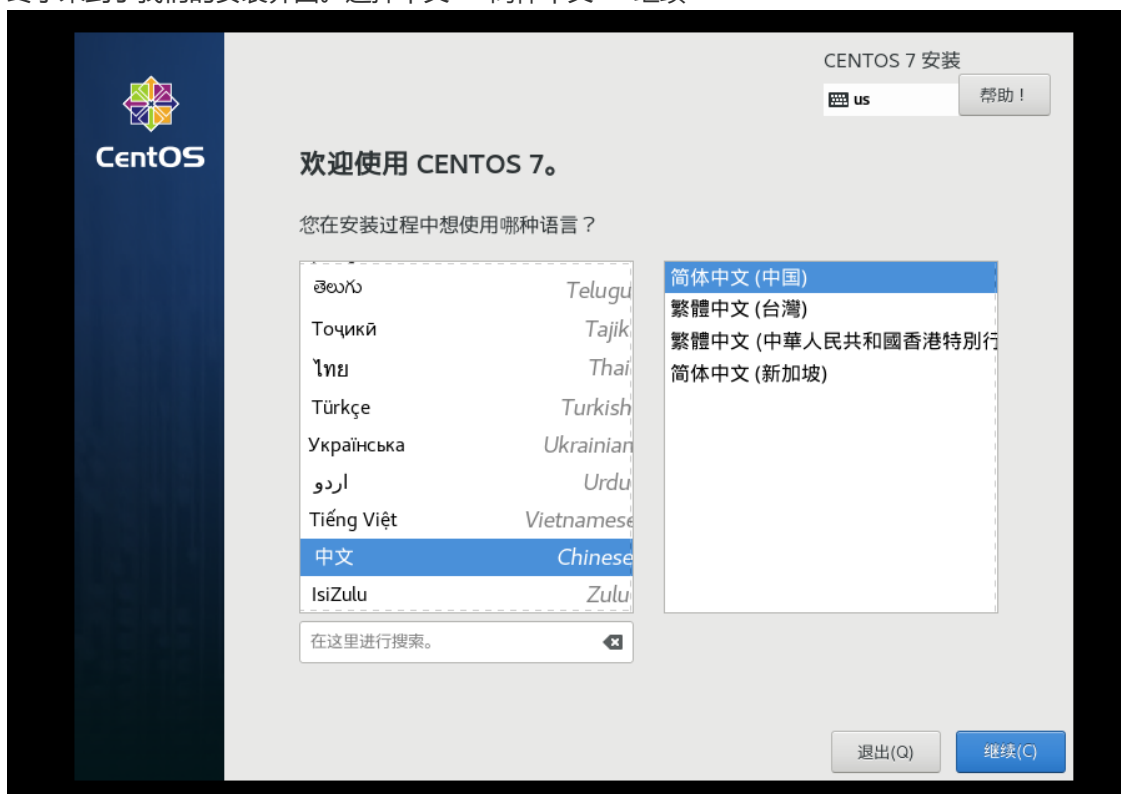
9. 直接按回车，它就会开始安装系统，

```
- Press the <ENTER> key to begin the installation process.

[ 10.497499] dracut-pre-udev[329]: modprobe: ERROR: could not insert 'floppy':
No such device

[ OK ] Started Show Plymouth Boot Screen.
[ OK ] Reached target Paths.
[ OK ] Started Forward Password Requests to Plymouth Directory Watch.
[ OK ] Reached target Basic System.
Mounting Configuration File System...
[ OK ] Mounted Configuration File System.
[ OK ] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[ OK ] Started Open-iSCSI.
Starting dracut initqueue hook...
[ 18.672994] sd 2:0:0:0: [sdal Assuming drive cache: write through
[ 18.434643] dracut-initqueue[1110]: mount: /dev/sr0 is write-protected, mounting read-only
[ OK ] Started Show Plymouth Boot Screen.
[ OK ] Reached target Paths.
[ OK ] Started Forward Password Requests to Plymouth Directory Watch.
[ OK ] Reached target Basic System.
Mounting Configuration File System...
[ OK ] Mounted Configuration File System.
[ OK ] Started Device-Mapper Multipath Device Controller.
Starting Open-iSCSI...
[ OK ] Started Open-iSCSI.
Starting dracut initqueue hook...
[ 18.434643] dracut-initqueue[1110]: mount: /dev/sr0 is write-protected, mounting read-only
[ OK ] Created slice system-checkisomd5.slice.
Starting Media check on /dev/sr0...
/dev/sr0: 1d2e8463d1bcae8ad491f4e5a07eb79f
Fragment sums: 3b55db189eeea7a20be82b85b49542dcc5ee4bb41bce8ff2fd4d675af7e
Fragment count: 20
Press [Esc] to abort check.
Checking: 004.1%_
```

10. 终于来到了我们的安装界面。选择中文--->简体中文--->继续



11. 点击安装位置选择自动分区，然后点击网络和主机名，点击右边的按钮。然后点击完成。点击开始安装。



安装信息摘要

CENTOS 7 安装

cn

帮助!



日期和时间(T)
亚洲/上海 时区



键盘(N)
汉语



语言支持(L)
简体中文 (中国)

软件



安装源(I)
本地介质



软件选择(S)
最小安装

系统



安装位置(D)
已选择自动分区



KDUMP
已启用 Kdump



网络和主机名(N)
有线 (ens33) 已连接



SECURITY POLICY
No profile selected

退出(Q)

开始安装(B)

在点击“开始安装”按钮前我们并不会操作您的磁盘。

安装目标位置

完成(D)

CENTOS 7 安装

cn

帮助!

设备选择

选择要在其中安装系统的设备。点击主菜单中的“开始安装”按钮前不会对该设备进行任何操作。

本地标准磁盘

20 GiB



VMware, VMware Virtual S
sda / 992.5 KiB 空闲

不会对未在此处选择的磁盘进行任何操作。

专用磁盘 & 网络磁盘

添加磁盘(A)...

不会对未在此处选择的磁盘进行任何操作。

其它存储选项

分区

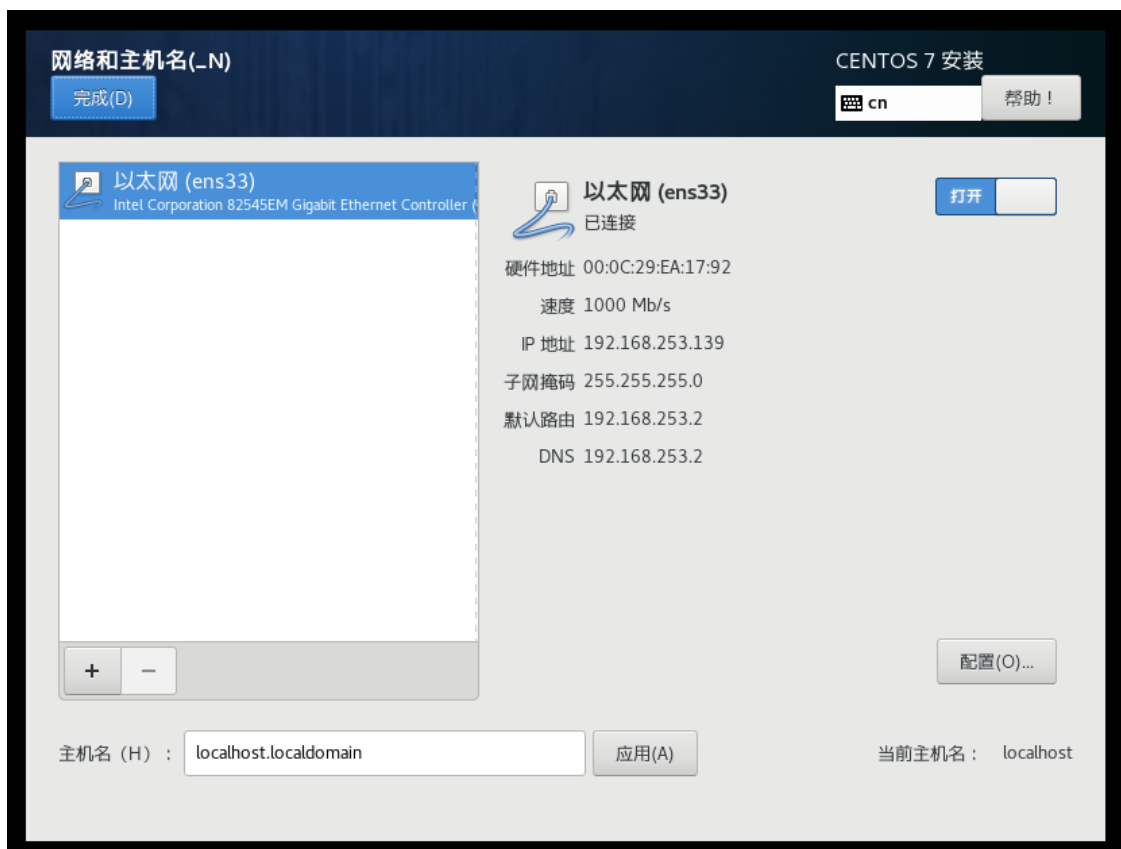
☒ 自动配置分区(U)。 ☐ 我要配置分区(I)。

☐ 我想让额外空间可用(M)。

加载

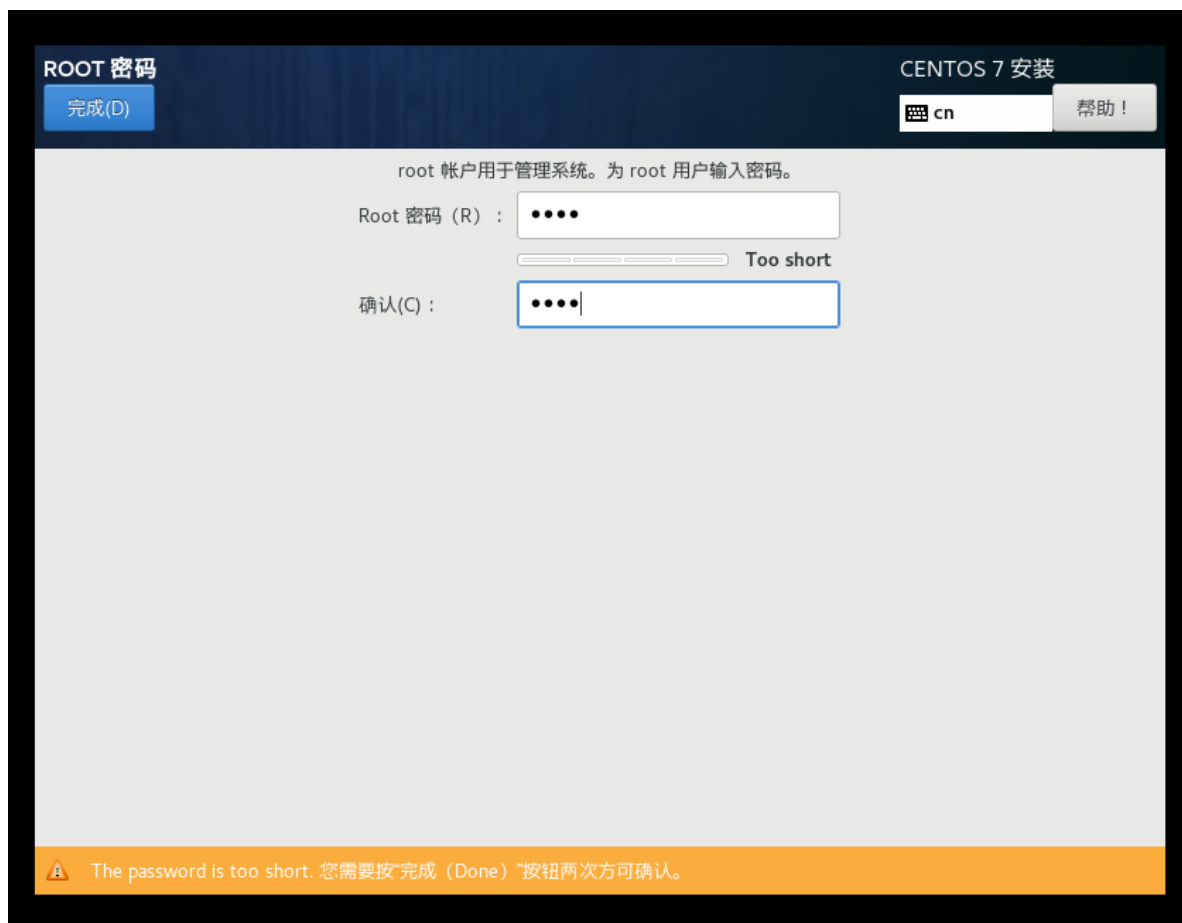
[完整磁盘摘要以及引导程序\(F\)...](#)

已选择 1 个磁盘；容量 20 GiB；992.5 KiB 空闲 [刷新\(R\)](#)



12. 点击**ROOT密码** 设置密码，点击完成。





13. 耐心等待，最后点击重启。



14. 重启之后选择第一个，键盘键入Enter即可。下面是它的登录界面，它没有图形化界面，我们先登录进去。

```
CentOS Linux 7 (Core)
Kernel 3.10.0-957.el7.x86_64 on an x86_64

localhost login: root_
```

15. 如此便登录进来了，输密码的时候不要以为自己没输入进去，只不过centos系统默认不显示你输入的密码而以，同时要注意的是，进入虚拟机，你的数字输入键可能没有开启，如果密码你设置的有数字，你可能会登录失败，只要点击键盘上的Num键就可以输入数字了。

```
CentOS Linux 7 (Core)
Kernel 3.10.0-957.el7.x86_64 on an x86_64

localhost login: root
Password:
[root@localhost ~]#
```

16. 接下来我们为centos安装图形化界面【对，它可以安装图形化界面，因为你最后测试nginx的时候需要用到浏览器输入地址，所以第一步先给它安装个图形化界面】我的安装步骤照搬的<https://www.linuxidc.com/Linux/2018-04/152000.htm>
17. 首先，**确保你自己是root模式**，我们的用户目前只有root，所以这一步不用担心。接下来输入下面的命令：`yum groupinstall "X Window System"`

```
CentOS Linux 7 (Core)
Kernel 3.10.0-957.el7.x86_64 on an x86_64

localhost login: root
Password:
[root@localhost ~]# yum groupinstall "X Window System"
```

18. 安装过程中会有让你输入y/d什么的 全部输入y即可，下图是这一步骤操作成功的结果图。

```
nm-connection-editor.x86_64 0:1.8.6-2.el7
odd.job.x86_64 0:0.31.5-4.el7
orc.x86_64 0:0.4.26-1.el7
pixmap.x86_64 0:0.34.0-1.el7
plymouth-plugin-label.x86_64 0:0.8.9-0.31.20140113.el7.centos
plymouth-theme-charge.x86_64 0:0.8.9-0.31.20140113.el7.centos
pulseaudio-libs.x86_64 0:10.0-5.el7
pykickstart.noarch 0:1.99.66.19-2.el7
python-lfy.noarch 0:0.75-6.el7
python-backports-ssl_match_hostname.noarch 0:3.5.0.1-1.el7
python-chardet.noarch 0:2.2.1-1.el7.1
python-deltarpm.x86_64 0:3.6-3.el7
python-gobject.x86_64 0:3.22.0-1.el7.4.1
python-kitchen.noarch 0:1.1.1-5.el7
python-meh-gui.noarch 0:0.25.2-1.el7
python-ntplib.noarch 0:0.3.2-1.el7
python-pyblock.x86_64 0:0.53-6.el7
python2-blockdev.x86_64 0:2.10-3.el7
putz.noarch 0:2016.10-2.el7
rest.x86_64 0:0.8.1-2.el7
satyr.x86_64 0:0.13-15.el7
sound-theme-freedesktop.noarch 0:0.8-3.el7
startup-notification.x86_64 0:0.12-0.el7
tigervnc-server-minimal.x86_64 0:1.8.0-13.el7
usermode.x86_64 0:1.111-5.el7
webkitgtk4-jsc.x86_64 0:2.20.5-1.el7
xcb-util.x86_64 0:0.4.0-2.el7
xml-common.noarch 0:0.6.3-39.el7
xmlrpc-c-client.x86_64 0:1.32.5-1995.svn2451.el7
xorg-x11-drv-dummy.x86_64 0:0.3.7-1.el7.1
xorg-x11-drv-fbdev.x86_64 0:0.5.0-1.el7
xorg-x11-drv-nouveau.x86_64 1:1.0.15-1.el7
xorg-x11-drv-synaptics.x86_64 0:1.9.0-2.el7
xorg-x11-drv-vesa.x86_64 0:2.4.0-1.el7
xorg-x11-drv-vmware.x86_64 0:13.2.1-1.el7.1
xorg-x11-drv-wacom.x86_64 0:0.36.1-1.el7
xorg-x11-server-utils.x86_64 0:7.7-20.el7
yelp.x86_64 2:3.28.1-1.el7
yelp-xsl.noarch 0:3.28.0-1.el7
zenity.x86_64 0:3.20.1-1.el7

Dependency Updated:
NetworkManager.x86_64 1:1.12.0-10.el7_6
NetworkManager-tui.x86_64 1:1.12.0-10.el7_6
systemd-sysv.x86_64 0:219-62.el7_6.9
NetworkManager-libnm.x86_64 1:1.12.0-10.el7_6
systemd.x86_64 0:219-62.el7_6.9
NetworkManager-team.x86_64 1:1.12.0-10.el7_6
systemd-libs.x86_64 0:219-62.el7_6.9

Complete!
[root@localhost ~]#
```

19. 接下来输入第二个命令检查一下我们可以安装的软件以及安装好的软件【我们这里什么都没安装，肯定没有安装好的软件】：`yum grouplist`

```
[root@localhost ~]# yum grouplist
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.aliyun.com
 * updates: ftp.jaist.ac.jp
Available Environment Groups:
Minimal Install
Compute Node
Infrastructure Server
File and Print Server
Basic Web Server
Virtualization Host
Server with GUI
GNOME Desktop
KDE Plasma Workspaces
Development and Creative Workstation
Available Groups:
Compatibility Libraries
Console Internet Tools
Development Tools
Graphical Administration Tools
Legacy UNIX Compatibility
Scientific Support
Security Tools
Smart Card Support
System Administration Tools
System Management
Done
[root@localhost ~]#
```

箭头标的这两个是需要我们去下载的：GNOME Desktop 和 Graphical Administration Tools

接下来我们去安装它们，安装命令为 `yum groupinstall "GNOME Desktop" "Graphical Administration Tools"`，千万不能把单词打错了。注意。

20. 运行完成的结果图如下【中途遇到的要输入y】：

xdg-desktop-portal	x86_64	1.0.2-1.el7	base	248 k
xdg-user-dirs	x86_64	0.15-5.el7	base	59 k
xdg-utils	noarch	1.1.0-0.17.20120809git.el7	base	70 k
xorg-x11-font-utils	x86_64	1:7.5-21.el7	base	194 k
xorg-x11-fonts-Type1	noarch	7.5-9.el7	base	521 k
xorg-x11-proto-devel	noarch	2018.4-1.el7	base	280 k
yajl	x86_64	2.0.4-4.el7	base	39 k
Updating for dependencies:				
bind-libs-lite	x86_64	32:9.9.4-74.el7_6.2	updates	742 k
bind-license	noarch	32:9.9.4-74.el7_6.2	updates	87 k
dbus	x86_64	1:1.10.24-13.el7_6	updates	245 k
dbus-libs	x86_64	1:1.10.24-13.el7_6	updates	169 k
krb5-libs	x86_64	1.15.1-37.el7_6	updates	883 k
nss	x86_64	3.36.0-7.1.el7_6	updates	835 k
nss-sysinit	x86_64	3.36.0-7.1.el7_6	updates	62 k
nss-tools	x86_64	3.36.0-7.1.el7_6	updates	515 k
nss-util	x86_64	3.36.0-1.1.el7_6	updates	78 k
policycoreutils	x86_64	2.5-29.el7_6.1	updates	916 k
Transaction Summary				
=====				
Install	252 Packages (+539 Dependent packages)			
Upgrade	(10 Dependent packages)			
Total download size: 598 M				
Is this ok [y/d/A]?				

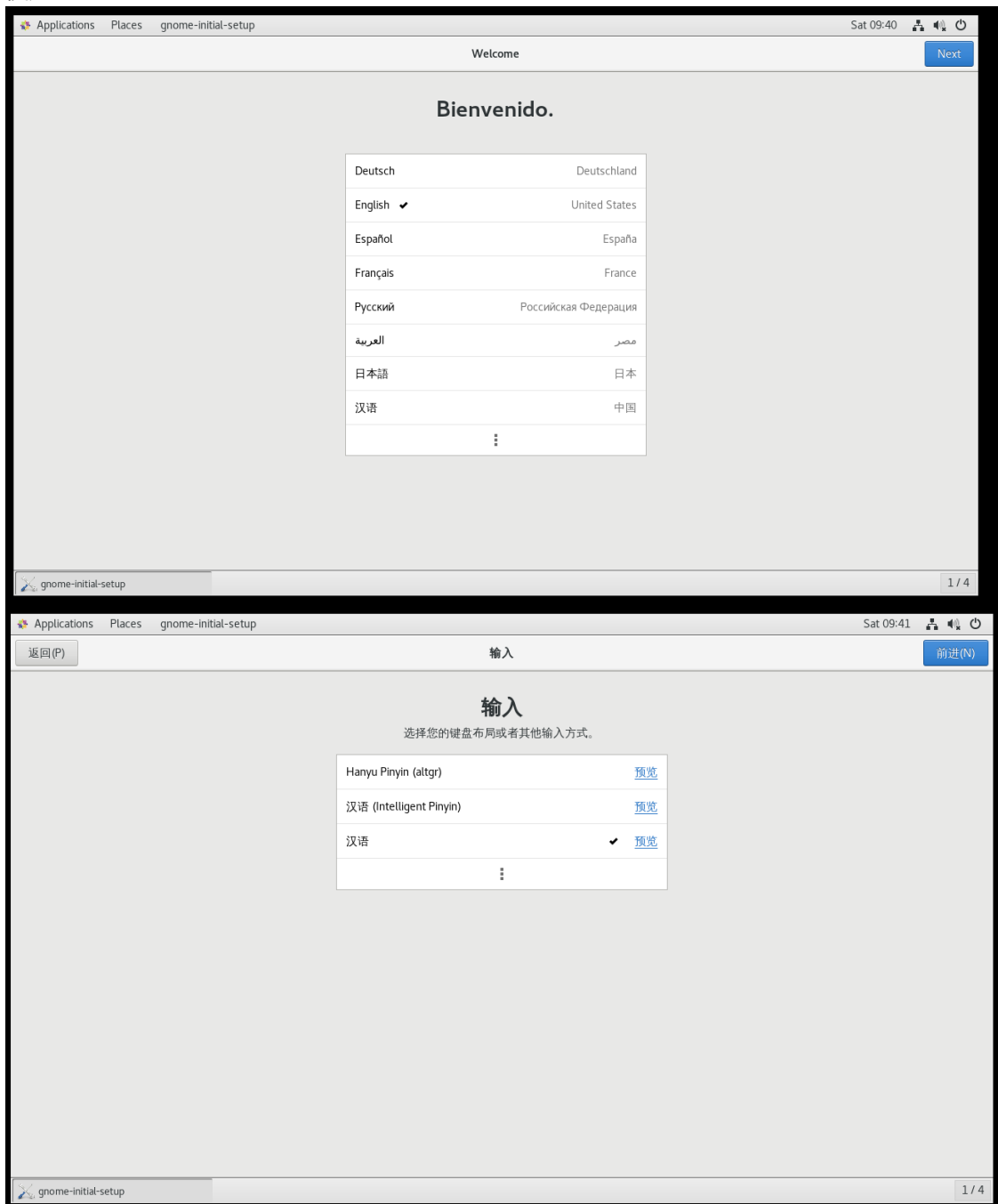
```
rdma-core.x86_64 0:0.17.2-3.el7
rhino.noarch 0:1.785-1.el7
rtkit.x86_64 0:0.11-10.el7
samba-common.noarch 0:4.8.3-6.el7_6
sane-backends.x86_64 0:1.0.24-12.el7
sbc.x86_64 0:1.0-5.el7
seawebios-bin.noarch 0:1.11.0-2.el7
setroubleshoot-plugins.noarch 0:3.0.67-3.el7
sgabios-bin.noarch 1:0.20110622eva-4.el7
skddic.noarch 0:20130104-6.T1435.el7
sox.x86_64 0:14.4.1-6.el7
speech-dispatcher-python.x86_64 0:0.7.1-15.el7
spice-glib.x86_64 0:0.35-2.el7
spice-server.x86_64 0:0.14.0-6.el7_6.1
sssd-common.x86_64 0:1.16.2-13.el7_6.8
sssd-ipa.x86_64 0:1.16.2-13.el7_6.8
sssd-krb5-common.x86_64 0:1.16.2-13.el7_6.8
sssd-proxy.x86_64 0:1.16.2-13.el7_6.8
systemd-python.x86_64 0:219-62.el7_6.9
tagsoup.noarch 0:1.2.1-0.el7
telepathy-farstream.x86_64 0:0.6.0-5.el7
telepathy-gabble.x86_64 0:0.10.1-4.el7
telepathy-haze.x86_64 0:0.0.0-1.el7
telepathy-mission-control.x86_64 1:5.16.3-3.el7
thai-scalable-fonts-common.noarch 0:0.5.0-7.el7
tracker.x86_64 0:1.10.5-6.el7
tzdata-java.noarch 0:2019b-1.el7
unbound-libs.x86_64 0:1.6.6-1.el7
urw-fonts.noarch 0:2.4-16.el7
usbredir.x86_64 0:0.7.1-3.el7
vfm-filesystem.x86_64 2:7.4.160-6.el7_6
vte-profile.x86_64 0:0.52.2-2.el7
wavpack.x86_64 0:4.60.1-9.el7
webkitgtk3.x86_64 0:2.4.11-2.el7
xcb-util-image.x86_64 0:0.4.0-2.el7
xcb-util-renderutil.x86_64 0:0.3.9-3.el7
xdg-desktop-portal.x86_64 0:1.0.2-1.el7
xdg-utils.noarch 0:1.1.0-0.17.20120809git.el7
xorg-x11-fonts-Type1.noarch 0:7.5-9.el7
yajl.x86_64 0:2.0.4-4.el7

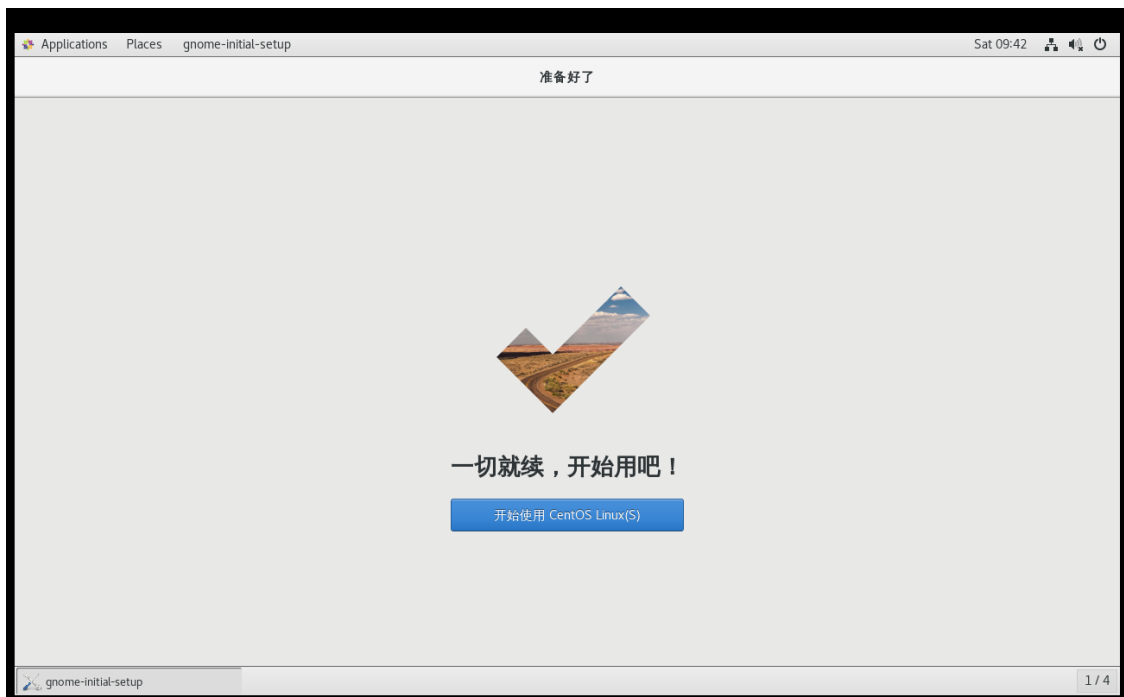
Dependency Updated:
bind-libs-lite.x86_64 32:9.9.4-74.el7_6.2 bind-license.noarch 32:9.9.4-74.el7_6.2 dbus.x86_64 1:1.10.24-13.el7_6 dbus-libs.x86_64 1:1.10.24-13.el7_6
krb5-libs.x86_64 0:1.15.1-37.el7_6 nss.x86_64 0:3.36.0-7.1.el7_6 nss-sysinit.x86_64 0:3.36.0-7.1.el7_6 nss-tools.x86_64 0:3.36.0-7.1.el7_6
nss-util.x86_64 0:3.36.0-1.1.el7_6 policycoreutils.x86_64 0:2.5-29.el7_6.1

Complete!
[root@localhost ~]#
```

完成！

21. 接下来输入 `startx` 命令进入图形化界面。选择汉语-->前进-->汉语-->前进-->前进-->跳过-->开始使用CentOS





22. 接下来就可以开始安装我们的N【nginx】M【mysql】P【php】了，在此之前先 `yum update` 一下，如果提示yum lock 说明有进程正在使用yum，因此重启下虚拟机。再执行这个命令。

安装nginx之前的准备工作

在安装之前，我们需要首先关闭下防火墙以及SELinux服务，避免这两个进程影响我们的安装。

查看防火墙状态的命令是 `systemctl status firewalld`

临时关闭防火墙的命令是 `systemctl stop firewalld`

永久关闭防火墙的命令是 `systemctl disable firewalld`

查看SELinux当前状态的命令是 `getenforce`

临时关闭SELinux的命令是 `setenforce 0`

永久关闭SELinux需要修改它的配置文件，它的配置文件在 `/etc/selinux/` 目录下的 `config` 里面，只需要将 `SELINUX=enforcing` 修改为 `SELINUX=disabled` 即可。命令行下需要修改文件需要使用到 `vi` 命令，具体是 `vi /etc/selinux/config` 进入该文件，然后按 `i` 这时你才可以修改文件内容，然后去修改，修改完成后要保存的话，首先按ESC键，然后输入 `:wq` 这样就保存好了。

首先我们关闭防火墙：

```
[root@localhost ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2019-08-03 09:46:38 CST; 10min ago
     Docs: man:firewalld(1)
    Main PID: 779 (firewalld)
       Tasks: 2
      CGroup: /system.slice/firewalld.service
              └─779 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

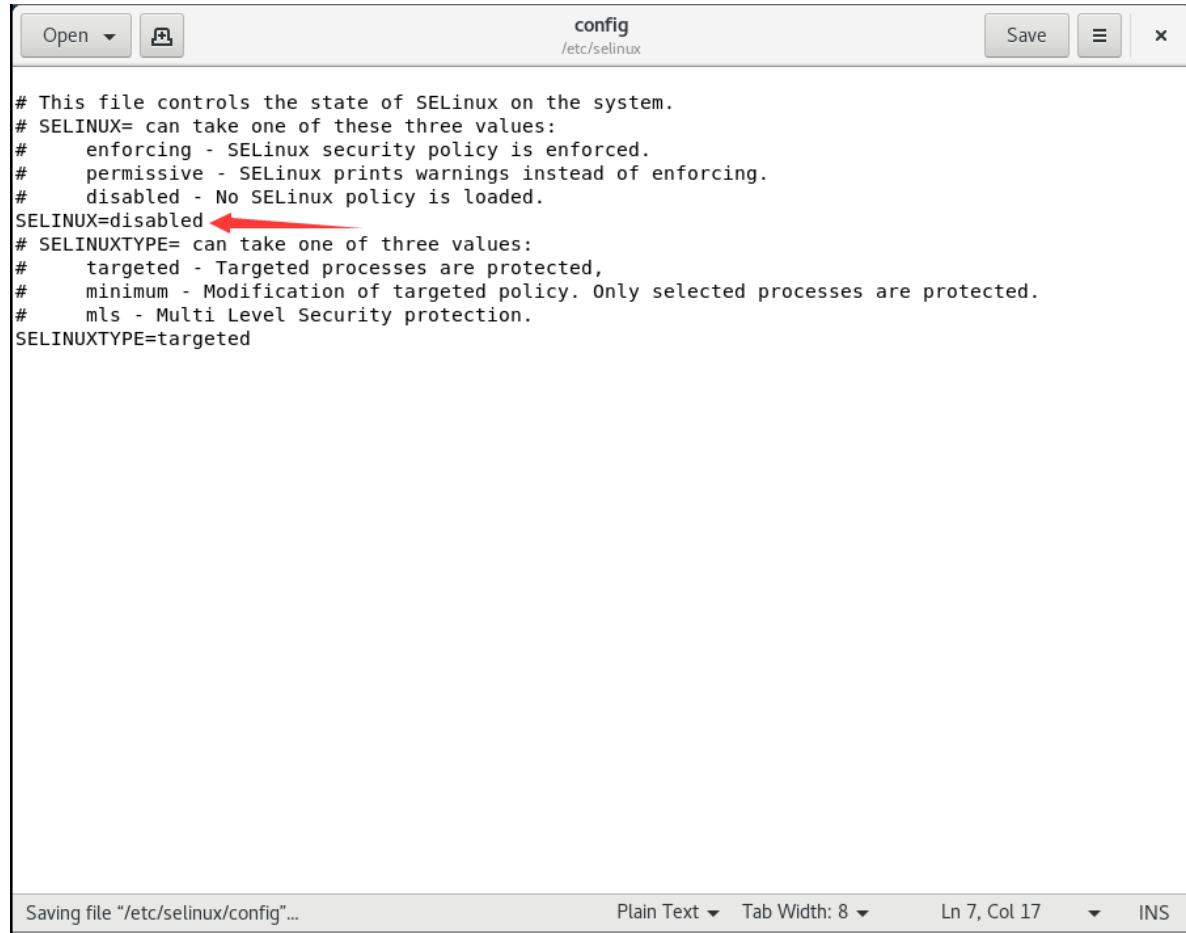
Aug 03 09:46:37 localhost.localdomain systemd[1]: Starting firewalld - dynamic...
Aug 03 09:46:38 localhost.localdomain systemd[1]: Started firewalld - dynamic...
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]# systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@localhost ~]#
```

然后我们关闭selinux:

```
[root@localhost ~]# getenforce
Enforcing
[root@localhost ~]#
```

这个结果说明SELinux服务开启了

我们需要找到它的配置文件，点击图形化界面上的places找到computer，然后来的etc/selinux/config这里，双击config，编辑它。点击保存。此时还未生效，我们可以使用 `setenforce 0` 先临时让它关闭，下次重启虚拟机，我们这次修改的配置文件就会生效。



```
config
/etc/selinux

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted

Saving file "/etc/selinux/config"...
Plain Text Tab Width: 8 Ln 7, Col 17 INS
```

```
root@localhost:~
File Edit View Search Terminal Help
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2019-08-03 09:46:38 CST; 10min ago
     Docs: man:firewalld(1)
  Main PID: 779 (firewalld)
    Tasks: 2
   CGroup: /system.slice/firewalld.service
           └─779 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Aug 03 09:46:37 localhost.localdomain systemd[1]: Starting firewalld - dynamic...
Aug 03 09:46:38 localhost.localdomain systemd[1]: Started firewalld - dynamic...
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost ~]# systemctl disable firewalld
Removed symlink /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed symlink /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@localhost ~]# getenforce
Enforcing
[root@localhost ~]# getenforce
Enforcing
[root@localhost ~]# setenforce 0
[root@localhost ~]# getenforce
Permissive
[root@localhost ~]#
```

开始正式安装nginx

1. 运行以下命令安装Nginx

`yum -y install nginx`,此时提示我nginx没有包可以用, 原因为nginx不是基础CentOS存储库的一部分。

但是我们可以安装EPEL repository来获取nginx, 因此第一步输入下面的命令 `yum install epel-release`

然后输入 `yum -y install nginx`, 大功告成。

```
[root@localhost ~]# yum -y install nginx
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.aliyun.com
 * updates: ftp.jaist.ac.jp
No package nginx available.
Error: Nothing to do
[root@localhost ~]# yum install epel-release
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.aliyun.com
 * extras: mirrors.aliyun.com
 * updates: ftp.jaist.ac.jp
Resolving Dependencies
--> Running transaction check
--> Package epel-release.noarch 0:7-11 will be installed
--> Finished Dependency Resolution
```

Dependencies Resolved

Package	Arch	Version	Repository	Size
Installing:				
epel-release	noarch	7-11	extras	15 k

```
root@localhost:~
```

```
Complete!
[root@localhost ~]# yum -y install nginx
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
epel/x86_64/metalink | 8.3 kB 00:00
* base: mirrors.aliyun.com
* epel: mirrors.yun-idc.com
* extras: mirrors.aliyun.com
* updates: ftp.jaist.ac.jp
epel | 5.3 kB 00:00
(1/3): epel/x86_64/group_gz | 88 kB 00:00
(2/3): epel/x86_64/updateinfo | 994 kB 00:00
(3/3): epel/x86_64/primary_db | 6.8 MB 00:01
Resolving Dependencies
--> Running transaction check
--> Package nginx.x86_64 1:1.12.2-3.el7 will be installed
--> Processing Dependency: nginx-all-modules = 1:1.12.2-3.el7 for package: 1:nginx-1.12.2-3.el7.x86_64
--> Processing Dependency: nginxfilesystem = 1:1.12.2-3.el7 for package: 1:nginx-1.12.2-3.el7.x86_64
--> Processing Dependency: nginxfilesystem for package: 1:nginx-1.12.2-3.el7.x86_64
--> Running transaction check
--> Package nginx-all-modules.noarch 1:1.12.2-3.el7 will be installed
--> Processing Dependency: nginx-mod-http-geoip = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Processing Dependency: nginx-mod-http-image-filter = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Processing Dependency: nginx-mod-http-perl = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Processing Dependency: nginx-mod-http-xslt-filter = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Processing Dependency: nginx-mod-mail = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Processing Dependency: nginx-mod-stream = 1:1.12.2-3.el7 for package: 1:nginx-all-modules-1.12.2-3.el7.noarch
--> Package nginxfilesystem.noarch 1:1.12.2-3.el7 will be installed
--> Running transaction check
--> Package nginx-mod-http-geoip.x86_64 1:1.12.2-3.el7 will be installed
--> Package nginx-mod-http-image-filter.x86_64 1:1.12.2-3.el7 will be installed
--> Package nginx-mod-http-perl.x86_64 1:1.12.2-3.el7 will be installed
--> Package nginx-mod-http-xslt-filter.x86_64 1:1.12.2-3.el7 will be installed
--> Package nginx-mod-mail.x86_64 1:1.12.2-3.el7 will be installed
--> Package nginx-mod-stream.x86_64 1:1.12.2-3.el7 will be installed
--> Finished Dependency Resolution

[root@localhost ~]#
```

接下来查看下nginx的版本 命令为 `nginx -v`

```
[root@localhost ~]# nginx -v
nginx version: nginx/1.12.2
[root@localhost ~]#
```

至此nginx安装完成

安装mysql

1. 首先更新YUM源

```
rpm -Uvh http://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm
```

2. 运行以下命令安装MySQL，这个过程会比较缓慢【看网速】。

```
yum -y install mysql-community-server
```

```
[root@localhost ~]# rpm -Uvh http://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm
Retrieving http://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm
warning: /var/tmp/rpm-tmp.5G008s: Header V3 DSA/SHA1 Signature, key ID 5072elf5: NOKEY
Preparing... ##### [100%]
Updating / installing...
 1:mysql57-community-release-el7-9 ##### [100%]
[root@localhost ~]# yum -y install mysql-community-server
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
```

3. 查看mysql版本号是否安装成功。命令为：

```
mysql -V V是大写的
[root@localhost ~]# mysql -V
mysql Ver 14.14 Distrib 5.7.27, for Linux (x86_64) using EditLine wrapper
[root@localhost ~]#
```

安装php

1. 依次运行下列命令更新YUM源

```
yum install -y http://dl.iuscommunity.org/pub/ius/stable/CentOS/7/x86_64/ius-release-1.0-15.ius.centos7.noarch.rpm
```

```
rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
```

```
[root@localhost ~]# yum install -y http://dl.iuscommunity.org/pub/ius/stable/CentOS/7/x86_64/ius-release-1.0-15.ius.centos7.noarch.rpm
Loaded plugins: fastestmirror, langpacks
ius-release-1.0-15.ius.centos7.noarch.rpm | 8.1 kB 00:00:00
Examining /var/tmp/yum-root-1Wuxiy/ius-release-1.0-15.ius.centos7.noarch.rpm: ius-release-1.0-15.ius.centos7.noarch
Marking /var/tmp/yum-root-1Wuxiy/ius-release-1.0-15.ius.centos7.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
--> Package ius-release.noarch 0:1.0-15.ius.centos7 will be installed
--> Finished Dependency Resolution
```

Dependencies Resolved

Package	Arch	Version	Repository	Size
Installing: ius-release	noarch	1.0-15.ius.centos7	/ius-release-1.0-15.ius.centos7.noarch	8.5
Transaction Summary				
Install 1 Package				

```
[root@localhost ~]# rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
Retrieving https://mirror.webtatic.com/yum/el7/webtatic-release.rpm
Warning: /var/tmp/rpm-tmp.6BuViZ: Header V4 RSA/SHA1 Signature, key ID 62e74ca5: NOKEY
Preparing... ##### [100%]
Updating / installing...
 1:webtatic-release-7-3 ##### [100%]
[root@localhost ~]#
```

2. 运行以下命令安装PHP

```
yum -y install php70w-devel php70w.x86_64 php70w-cli.x86_64 php70w-common.x86_64
php70w-gd.x86_64 php70w-ldap.x86_64 php70w-mbstring.x86_64 php70w-mcrypt.x86_64
php70w-pdo.x86_64 php70w-mysqlnd php70w-fpm php70w-opcache php70w-pear redis
php70w-pear-mongo
```

```
[root@localhost ~]# yum -y install php70w-devel php70w.x86_64 php70w-cli.x86_64 php70w-common.x86_64 php70w-gd.x86_64 php70w-ldap.x86_64 php
70w-mbstring.x86_64 php70w-mcrypt.x86_64 php70w-pdo.x86_64 php70w-mysqlnd php70w-fpm php70w-opcache php70w-pear redis php70w-pear-mongo
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile

Installed:
php70w.x86_64 0:7.0.33-1.w7          php70w-cli.x86_64 0:7.0.33-1.w7          php70w-common.x86_64 0:7.0.33-1.w7
php70w-devel.x86_64 0:7.0.33-1.w7    php70w-fpm.x86_64 0:7.0.33-1.w7          php70w-gd.x86_64 0:7.0.33-1.w7
php70w-ldap.x86_64 0:7.0.33-1.w7      php70w-mbstring.x86_64 0:7.0.33-1.w7      php70w-mcrypt.x86_64 0:7.0.33-1.w7
php70w-mysqlnd.x86_64 0:7.0.33-1.w7   php70w-opcache.x86_64 0:7.0.33-1.w7      php70w-pdo.x86_64 0:7.0.33-1.w7
php70w-pear.x86_64 0:3.1.6-1.w7

Dependency Installed:
apr.x86_64 0:1.4.8-3.el7_4.1          apr-util.x86_64 0:1.5.2-6.el7          autoconf.noarch 0:2.69-11.el7
automake.noarch 0:1.13.4-3.el7          httpd.x86_64 0:2.4.6-89.el7.centos.1   httpd-tools.x86_64 0:2.4.6-89.el7.centos.1
libmcrypt.x86_64 0:2.5.8-13.el7         m4.x86_64 0:1.4.16-10.el7             mailcap.noarch 0:2.1.41-2.el7
pcre-devel.x86_64 0:8.32-17.el7         perl-Data-Dumper.x86_64 0:2.145-3.el7          perl-Test-Harness.noarch 0:3.28-3.el7
perl-Thread-Queue.noarch 0:3.02-2.el7   php70w-pear.noarch 1:1.10.4-1.w7       php70w-pear-igbinary.x86_64 0:2.0.5-1.w7
php70w-process.x86_64 0:7.0.33-1.w7     php70w-xml.x86_64 0:7.0.33-1.w7

Complete!
[root@localhost ~]#
```

3. 查看php的版本

```
Complete!
[root@localhost ~]# php -v
PHP 7.0.33 (cli) (built: Dec 6 2018 22:30:44) ( NTS )
Copyright (c) 1997-2017 The PHP Group
Zend Engine v3.0.0, Copyright (c) 1998-2017 Zend Technologies
    with Zend OPcache v7.0.33, Copyright (c) 1999-2017, by Zend Technologies
[root@localhost ~]#
```

至此三个软件安装完成了。接下来需要配置。

配置nginx

1. 运行以下命令备份Nginx配置文件。

```
cp /etc/nginx/nginx.conf /etc/nginx/nginx.conf.bak
```

2. 打开nginx配置文件并修改它

```
vim /etc/nginx/nginx.conf
```

VI 与 VIM 区别

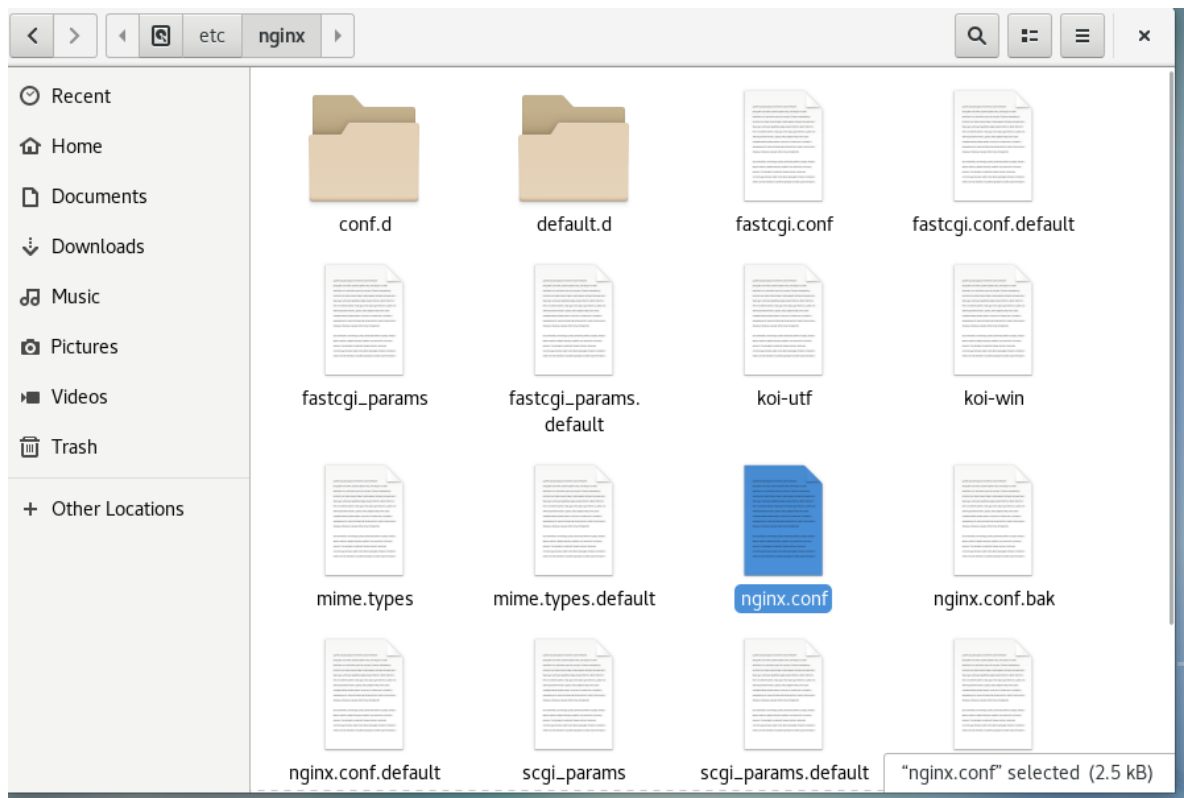
它们都是多模式编辑器，不同的是 `vim` 是 `vi` 的升级版本，它不仅兼容 `vi` 的所有指令，而且还有一些新的特性在里面。

按照上面 `vi` 的操作方式修改该文件。

3. 在 `server` 大括号内，添加下列配置信息，使 Nginx 支持 PHP 请求。

```
location / {
    index index.php index.html index.htm;
}
#配置Nginx通过fastcgi方式处理您的PHP请求
location ~ .php$ {
    root /usr/share/php;
    fastcgi_pass 127.0.0.1:9000; #Nginx通过本机的9000端口将PHP请求转发给PHP-FPM
    #进行处理。
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    include fastcgi_params; #Nginx调用fastcgi接口处理PHP请求
}
```

注意配置文件中的空格。



```

server {
    listen      80 default_server;
    listen      [::]:80 default_server;
    server_name _;
    root        /usr/share/nginx/html;

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;

    location / {
        index index.php index.html index.htm;
    }

    #配置Nginx通过fastcgi方式处理您的PHP请求
    location ~ .php$ {
        root /usr/share/php;
        fastcgi_pass 127.0.0.1:9000; #Nginx通过本机的9000端口将PHP请求转发给PHP-FPM进行处理。
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params; #Nginx调用fastcgi接口处理PHP请求
    }

    error_page 404 /404.html;
        location = /404.html {
    }

    error_page 500 502 503 504 /50x.html;
        location = /50x.html {
    }
}

```

4. 运行以下命令启动Nginx服务

```
systemctl start nginx
```

5. 运行以下命令添加Nginx服务开机自启动

```
systemctl enable nginx
```

```

root@localhost:~
File Edit View Search Terminal Help
[root@localhost ~]# systemctl start nginx
[root@localhost ~]# systemctl enable nginx
Created symlink from /etc/systemd/system/multi-user.target.wants/nginx.service to /usr/lib/systemd/system/nginx.service.
[root@localhost ~]#

```

配置mysql

1. 启动mysql服务

```
systemctl start mysqld
```

2. 设置MySQL服务开机自启动

```
systemctl enable mysqld
```

3. 用下列命令查看/var/log/mysqld.log文件，获取并记录root用户的初始密码。记住这个密码，下来重置root密码要使用

```
grep 'temporary password' /var/log/mysqld.log
```

4. 运行下列命令配置MySQL的安全性

```
mysql_secure_installation
```



```
[root@localhost ~]# systemctl start mysqld
[root@localhost ~]# systemctl enable mysqld
[root@localhost ~]# grep 'temporary password' /var/log/mysqld.log
2019-08-03T03:20:34.843812Z 1 [Note] A temporary password is generated for root@localhost: K:&Emlamr982
[root@localhost ~]#
```

从该图可以看出我的初始mysql密码为K:&Emlamr982

可以试着登录下mysql 看能不能登录进去 输入命令 `mysql -u root -p`，下来输入上面哪个密码即可。如果能登录进去就说明密码没输错（注意：键盘可能又关闭了数字键，记得开启）

```
[root@localhost ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.27

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

退出mysql输入 `exit`

```
[root@localhost ~]# mysql_secure_installation
```

Securing the MySQL server deployment.

Enter password for user root:

```
[root@localhost ~]# mysql_secure_installation
```

Securing the MySQL server deployment.

Enter password for user root: 输入刚才mysql的初始密码

The existing password for the user account root has expired. Please set a new password.

New password: 输入你准备修改的密码，这里的密码要求至少8位，并且必须出现大小写字母和特殊符号，否则会报错

Re-enter new password:

The 'validate_password' plugin is installed on the server.
The subsequent steps will run with the existing configuration of the plugin.

重新输入刚才修改了的密码

Using existing password for root.

Estimated strength of the password: 100

Change the password for root ? ((Press y|Y for Yes, any other key for No) : y 按y

New password:

这里是我的失误，不用管

Sorry, you can't use an empty password here.

New password:

再次输入刚才为root修改后的密码

Re-enter new password:

重复输入上一步骤修改的密码

Estimated strength of the password: 100

Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : y

By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have

按y


```
By default, a MySQL installation has an anonymous user,
allowing anyone to log into MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : y
Success.

Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.

Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y
- Dropping test database...
Success.
- Removing privileges on test database...
Success.
Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y
Success.

All done!
[root@localhost ~]#
```

删除匿名用户吗？选择y，删除

不允许root远程登录数据库，按y

删除存在于mysql中的test数据库，因为这个数据库任何人都有限，这样是很危险的，所以按y

删除在test数据库上的所有权限

现在重新加载权限表吗？我们现在要重新加载，所以按y

这样mysql就配置成功了。

配置php

刚在nginx的配置文件中 server里面写的 `root /usr/share/php` 所以我们需要到这个目录下新建我们的php文本，然后编写php程序，为了测试php脚本能否被nginx执行，所以写个简单点的php文件。

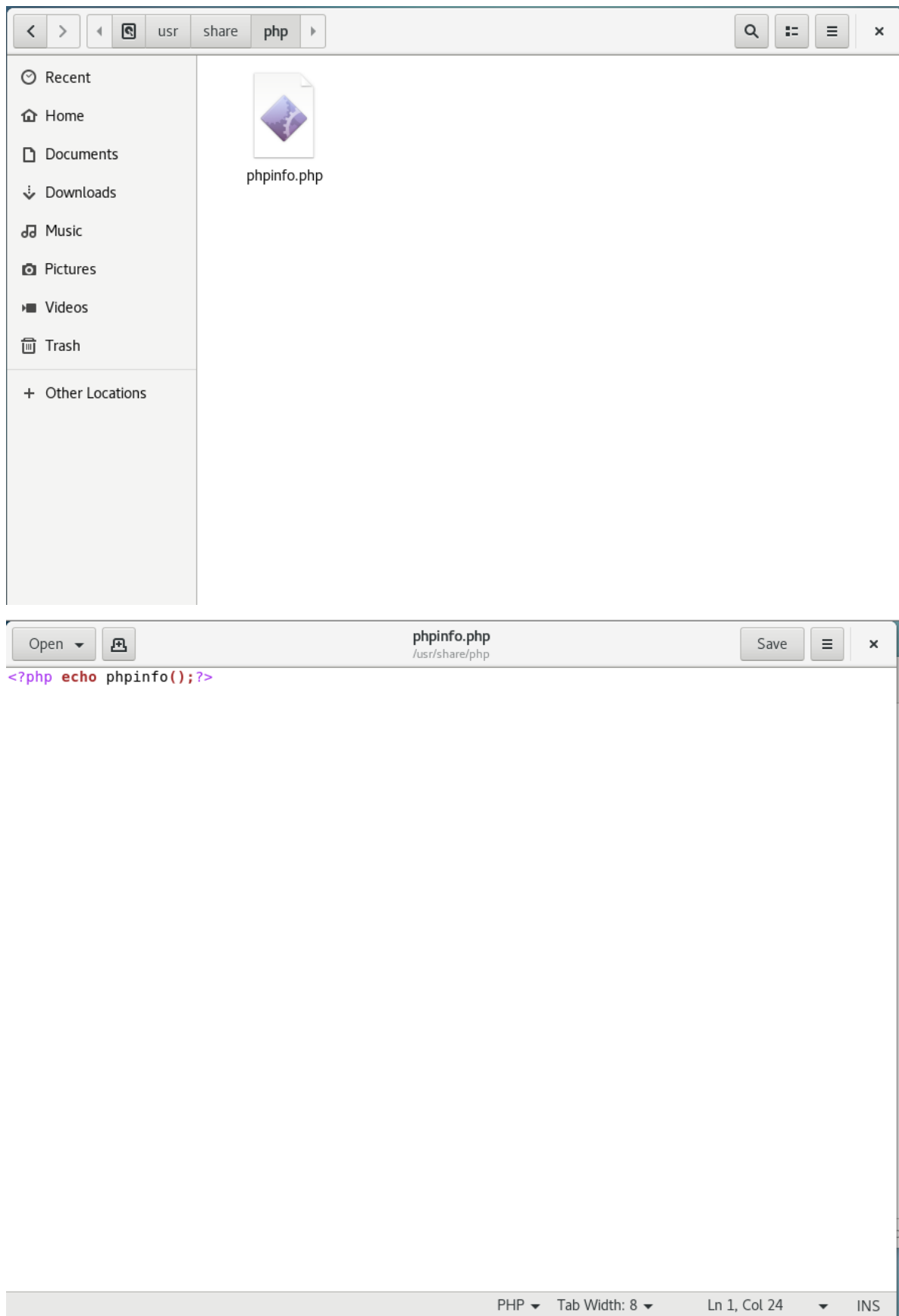
在centos中，好像不像windows那样可以右键 新建文件夹，因此我们需要使用到命令行。打开终端 然后输入

```
vim /usr/share/php/phpinfo.php
```

然后在phpinfo.php文件输入 `<?php echo phpinfo(); ?>` 的内容。

保存

```
[root@localhost ~]# vim /usr/share/php/phpinfo.php
[root@localhost ~]#
```



然后输入 `systemctl start php-fpm` 命令启动php-fpm服务

【php-fpm】服务：Nginx本身不能处理PHP，它只是个web服务器，当接收到请求后，如果是php请求，则发给php解释器处理，并把结果返回给客户端。那么php-fpm就是做的php和nginx的桥梁。

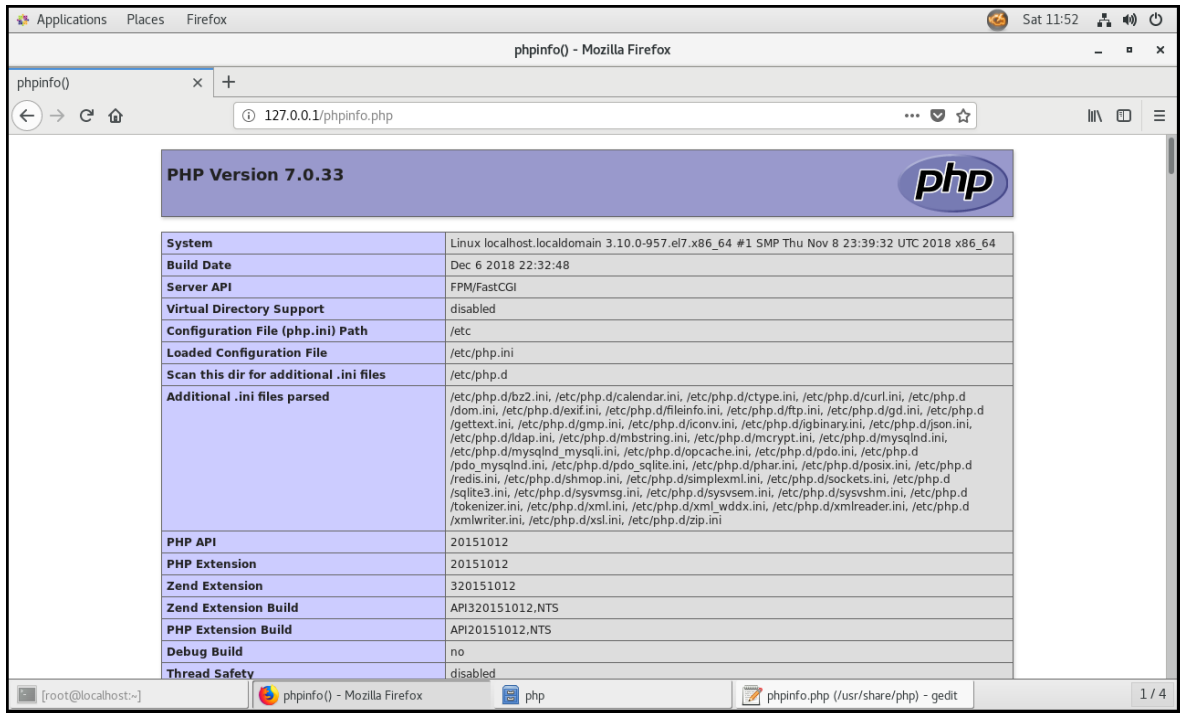
再输入 `systemctl enable php-fpm` 命令设置PHP-FPM开机自启动

```
All done!  
[root@localhost ~]# vim /usr/share/php/phpinfo.php  
[root@localhost ~]# systemctl start php-fpm  
[root@localhost ~]# systemctl enable php-fpm  
Created symlink from /etc/systemd/system/multi-user.target.wants/php-fpm.service to /usr/lib/systemd/system/php-fpm.service.  
[root@localhost ~]#
```

然后打开浏览器输入127.0.0.1/phpinfo.php

可以成功访问到这个页面。即测试成功了。

说明我们可以通过nginx服务器来解析php文件，并且显示出来。



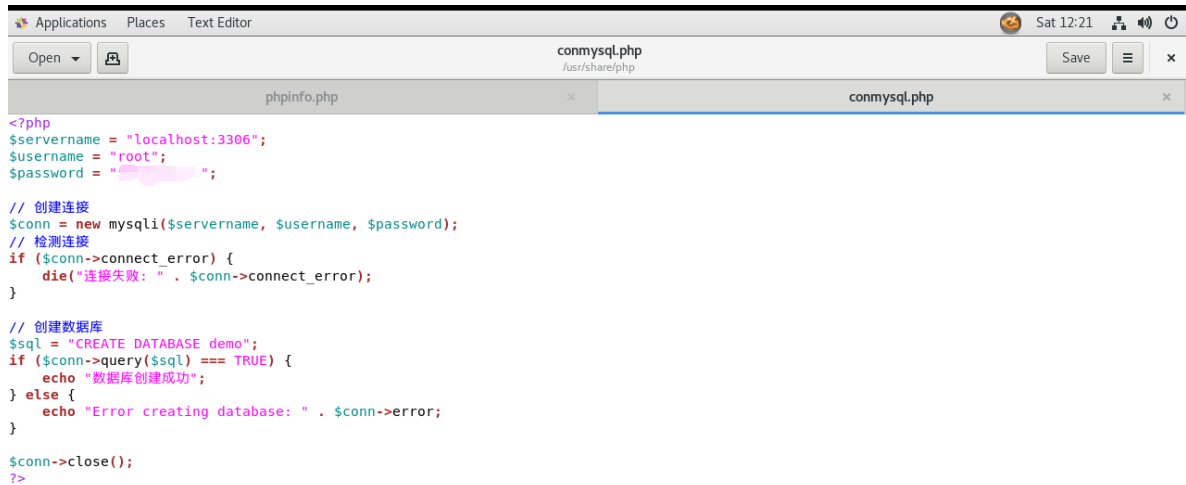
接下来我们需要连接数据库，其实连接数据库这种事情就比较简单了，我们已经有mysql数据库在我们的虚拟机上面，我们只需要在php代码里连接它，使用它就可以了。

我们在etc/share/php目录下创建一个conmysql.php的文件，然后输入关于数据库连接和创建的代码，看能否执行。

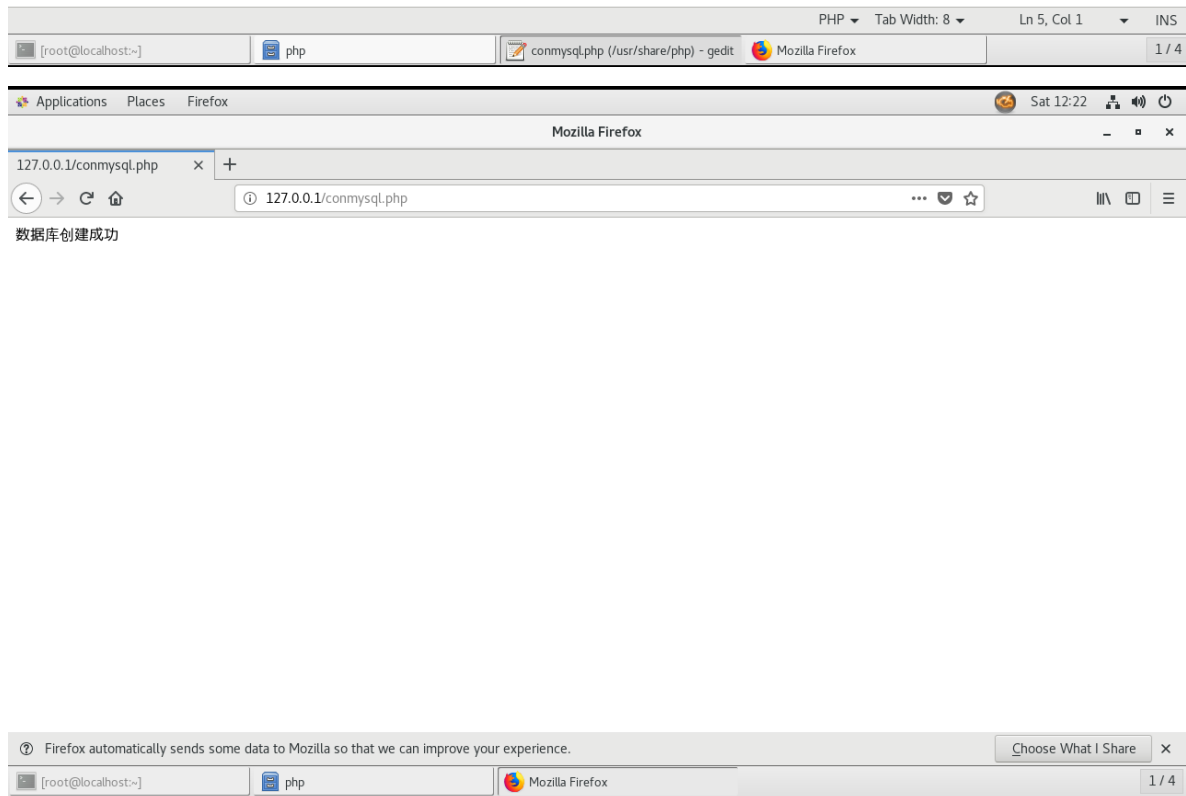
代码如下：

```
$servername = "localhost:3306";  
$username = "root";  
$password = "这里填你自己mysql的密码";  
  
// 创建连接  
$conn = new mysqli($servername, $username, $password);  
// 检测连接  
if ($conn->connect_error) {  
    die("连接失败: " . $conn->connect_error);  
}  
  
// 创建数据库  
$sql = "CREATE DATABASE demo";  
if ($conn->query($sql) === TRUE) {  
    echo "数据库创建成功";  
} else {  
    echo "Error creating database: " . $conn->error;  
}
```

```
$conn->close();  
?>
```



```
<?php  
$servername = "localhost:3306";  
$username = "root";  
$password = " ";  
  
// 创建连接  
$conn = new mysqli($servername, $username, $password);  
// 检测连接  
if ($conn->connect_error) {  
    die("连接失败: " . $conn->connect_error);  
}  
  
// 创建数据库  
$sql = "CREATE DATABASE demo";  
if ($conn->query($sql) === TRUE) {  
    echo "数据库创建成功";  
} else {  
    echo "Error creating database: " . $conn->error;  
}  
  
$conn->close();  
?>
```



访问127.0.0.1/conmysql.php后，成功连接mysql数据库，并且新创建了一个叫做demo的数据库。

下一步我们可以看看创建数据表的操作能否如预期一样执行。

我们新建一个createTable.php的文件

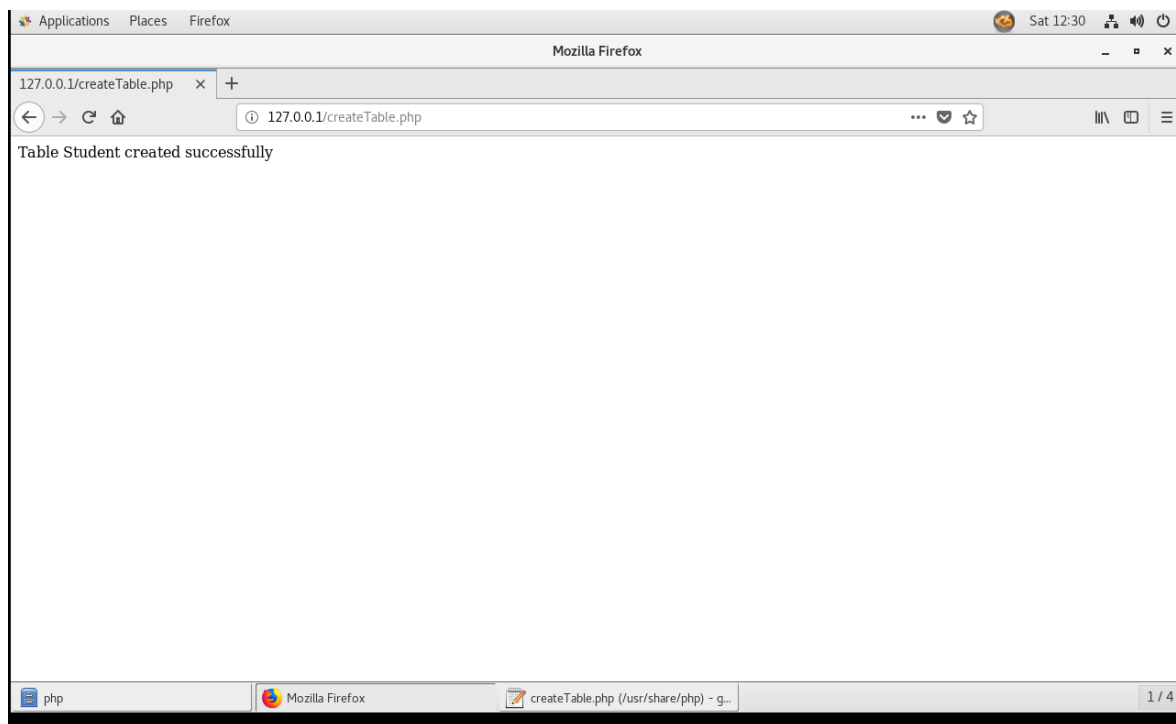
代码如下：

```
<?php  
$servername = "localhost:3306";  
$username = "root";  
$password = "你自己的mysql密码";  
$dbname = "demo"; //自己要连接的数据库名称
```

```
// 创建连接
$conn = new mysqli($servername, $username, $password, $dbname);
// 检测连接
if ($conn->connect_error) {
die("连接失败: " . $conn->connect_error);
}
// 使用 sql 创建数据表
$sql = "CREATE TABLE Student (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(30) NOT NULL,
address VARCHAR(30) NOT NULL,
email VARCHAR(50)
)";
if ($conn->query($sql) === TRUE) {
echo "Table MyGuests created successfully";
} else {
echo "创建数据表错误: " . $conn->error;
}
$conn->close();
?>
```

```
<?php
$servername = "localhost:3306";
$username = "root";
$password = "root:root";
$dbname = "demo"; //自己要连接的数据库名称
// 创建连接
$conn = new mysqli($servername, $username, $password, $dbname);
// 检测连接
if ($conn->connect_error) {
die("连接失败: " . $conn->connect_error);
}
// 使用 sql 创建数据表
$sql = "CREATE TABLE Student (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(30) NOT NULL,
address VARCHAR(30) NOT NULL,
email VARCHAR(50)
)";
if ($conn->query($sql) === TRUE) {
echo "Table Student created successfully";
} else {
echo "创建数据表错误: " . $conn->error;
}
$conn->close();
?>
```

然后测试这个php文件



成功创建了名叫Student的表。至此数据库的语句在php中是可以执行的。剩下的数据库的增删改查也一样。整个环境的搭建和测试至此结束。

服务器的加固

LNMP PHP环境的加固：

1.使用open_basedir限制虚拟主机跨目录访问

个人理解：对于黑客来说，要侵入一个网站，那么如果这个网站的一个目录被拿下，而且该目录可以访问web程序下的其他所有目录【开发人员没有去关注这个文件的权限】，那么黑客会轻易的将木马挂在该web程序下的所有文件中。因此在php环境中，使用这个open_basedir可以有效避免：**黑客拿到一个web程序的漏洞，就可以对所有web程序下的文件进行入侵。**但是同时这个open_basedir开启后会影I/O，因为每个调用的文件都需要判断是否在限制目录内。有网友进行过测试，最后得出的结论是使用open_basedir可以限制程序可操作的目录和文件，提高系统安全性。但会影响I/O性能导致系统执行变慢，**因此需要根据具体需求，在安全与性能上做平衡。**

2.禁用不安全的PHP函数

```
disable_functions =  
show_source,system,shell_exec,passthru,exec,popen,proc_open,proc_get_status,phpinfo
```

个人理解：由于这次实践只接触了phpinfo这个函数，所有就说说这个函数的危害性。测试LNMP的时候php文件中的代码就写的输出phpinfo();显示的页面信息十分丰富，包括php的版本号，以及mysql的版本号以及其他各种程序是否安装，每个软件肯定都有它的弱点，当黑客获取到了这些软件的版本信息后，很容易就能发现哪个软件的哪个版本容易破解，都有哪些漏洞？然后黑客就可以利用它所获得的这些信息来进行破坏。如果这些不安全的函数中还有能直接操作底层的，那就更加可怕，后果不堪设想。

3.关注软件安全资讯

个人理解：这一点不用多说，0day漏洞，技术的更新迭代速度，安全问题日益重要，只有及时获取安全相关的资讯才可以防患于未然。对于网络安全的从业者，必须要时刻关注网络安全方向的资讯，听一个朋友说可以经常去Twitter看那些网络安全界的大佬们发的推文，这些是很有必要的。

4.php用户只读

个人理解：以前接触过一点点安全方向的知识，sql注入攻击之类的，包括网络挂马，如果可以控制用户访问网站的时候只能读，而不能往网站里面写，那么即使黑客发现了这个网页的漏洞，它也无法上传恶意代码给服务器，这样就防止了被黑客攻击。如果是某些必须上传【例如上传身份证等图片信息之类的】，一定要在程序中做好安全校验。否则容易被黑客利用进而破坏服务器。

5.关闭php错误日志

个人理解：这个其实不是很清楚，javaweb的话我觉得错误日志可以记录下程序的错误给开发人员看，只要别把错误日志相关信息让用户看到就可以了，php的话如果关闭了错误日志，那程序如果出错了怎么去查找错误在哪儿？我觉得核心问题是不能让黑客拿到错误日志的信息，至于要不要关闭就另当别论了，可能有更好的解决办法。

6.php上传分离

个人理解：如果不进行上传分离，而让用户上传的数据就上传到动态服务器去解释执行，会被黑客利用执行恶意代码。上传分离可以使静态服务器存储文件，这样即使有恶意代码，静态服务器也不会去执行。

7.关闭PHP信息

个人理解：同禁止使用不安全的函数，思路是一样的 每个版本的每个东西肯定都有自身的弱点，要不然也不可能升级了，但是版本信息透露的不只是这些，通过版本信息，黑客可以知道这个版本的这个东西，哪些东西可以被利用，这对黑客来说是很重要的信息。

8.禁止动态加载链接库

个人理解：可能是因为动态加载链接库可以让黑客将它写的恶意代码在程序运行时偷偷的执行而不被察觉？这个有点不太清楚，个人感觉链接库里面可能有些不安全的函数，而黑客可以通过某些方式使用它们，因此禁止之后能够有效避免一些安全漏洞。

参考资料：<https://www.imooc.com/article/33496>

nginx的安全加固

这些并不是真正意义上nginx的加固，它主要是利用nginx加固web应用程序

1.屏蔽IP

个人理解：互联网时代，任何地方的任何人都可以在网络上访问信息，有心者也可以黑掉网站，为了避免这种状况，可以屏蔽掉除了美国和中国的IP，这样至少减少了一批国外的恶意扫描。如果不屏蔽这些IP，一个是服务器压力大，另一个无效的访问和恶意的扫描都会对服务器造成不利的影响。

2.封杀各种user-agent

个人理解：user-agent是浏览器的标识，当初在做爬虫的时候也了解过一点，可以通过伪装请求头，来达到爬取网页的效果，这里封杀各种user-agent主要是指那些恶意工具扫描留下来的特征值，这样就可以通过提取这些特征值来达到过滤的目的，如果黑客连某些基本信息都获取不到，那么他们也就无从继续下一步的工作了。

3.封杀特定的url

个人理解：这个不太清楚，可能是因为某些特定的url带有不安全的东西？或许是某些特征的url会对服务器造成影响。

4.强制网站使用域名

个人理解：网站渗透好像有包含扫描这个过程，通常扫描应该都是会扫描IP的，如果强制使用域名的话，就可以逃过IP扫描。因为你使用IP扫描什么也得不到。

5.url参数过滤敏感词

个人理解：应该和sql注入有关，例如select union等等，sql注入一直是互联网上比较多的一个漏洞，如果不进行url参数过滤的话，黑客可以很容易拿到数据库中的用户信息等等敏感信息。

6.强制要求referer

个人理解：很多网站目前都要求登录后才可以访问，如果你不强制要求referer的话，引用泄露的默认行为会使网站面临隐私和安全漏洞的风险。

mysql的安全加固

前面的mysql配置中有提到相关安全策略，也可以看做是加固的一种

1.账号安全

个人理解：禁止mysql以管理员权限运行，这个可理解为管理员权限太大，万一不小心删除了数据库那就造成了不可挽回的损失，

2.禁止远程登录

个人理解：这样可以防止黑客通过网络来达到mysql的最大权限，然后为所欲为。。。。。

3.口令

个人理解：口令好像mysql从某个版本开始要求设置密码至少8位，并且必须包含数字大小写字母以及特殊符号，目的是使暴力破解不那么容易拿下mysql的密码，加固了mysql的安全性。

4.授权

个人理解：根据用户需要，配置最小的权限，毕竟现在是数据的时代，如果不小心删除或者弄脏数据库，是会带来意想不到的损失的，又比如特定用户才可查看特定的表，这样可以防止越权获取敏感数据。

5.安装最新补丁

个人理解：每个版本都有安全性的问题，尽早发现尽早更新可以使软件更安全。有利于防止黑客入侵。

参考资料：<https://www.alibabacloud.com/help/zh/faq-detail/49568.htm>

https://blog.csdn.net/jy_he/article/details/52299884

<https://blog.csdn.net/fdipzone/article/details/54562656>

https://help.aliyun.com/document_detail/97251.html

<https://blog.csdn.net/lzm198707/article/details/50130615>

<https://segmentfault.com/a/1190000014610688>

<https://www.linuxidc.com/Linux/2018-04/152000.htm>