北京大学

PEKING UNIVERSITY

课程：LFS 系统搭建流程总结

学号：2001210476

姓名：谢新宇（大锅锅-）

日期：2021.4.7

Codechina 昵称：大锅锅-

Linux From Scratch 10.0

一、宿主机

1. 虚拟机：VMware 或 VirtualBox；
2. openEuler 操作系统镜像下载、配置和安装。

   起初使用 VMware 无法连接网络，安装 openEuler 时无以太网网络配置选项，改用 VirtualBox 后选择桥接网络。

二、准备工作

1. 准备宿主系统

   1） 运行脚本检查软件版本，未出现问题；

```
cat > version-check.sh << "EOF"
#!/bin/bash
# Simple script to list version numbers of critical development tools
export LC_ALL=C
bash --version | head -n1 | cut -d" " -f2-4
MYSH=$(readlink -f /bin/sh)
echo "/bin/sh -> $MYSH"
echo $MYSH | grep -q bash || echo "ERROR: /bin/sh does not point to bash"
unset MYSH

echo -n "Binutils: "; ld --version | head -n1 | cut -d" " -f3-
bison --version | head -n1

if [ -h /usr/bin/yacc ]; then
  echo "/usr/bin/yacc -> `readlink -f /usr/bin/yacc`";
elif [ -x /usr/bin/yacc ]; then
  echo yacc is `/usr/bin/yacc --version | head -n1`
else
  echo "yacc not found"
fi

bzip2 --version 2>&1 < /dev/null | head -n1 | cut -d" " -f1,6-
echo -n "Coreutils: "; chown --version | head -n1 | cut -d")" -f2
diff --version | head -n1
find --version | head -n1
gawk --version | head -n1

if [ -h /usr/bin/awk ]; then
  echo "/usr/bin/awk -> `readlink -f /usr/bin/awk`";
elif [ -x /usr/bin/awk ]; then
  echo awk is `/usr/bin/awk --version | head -n1`
else
  echo "awk not found"
fi
```

   2） cfdisk 创建分区 sdb1 作为文件系统、sdb2 作为 swap 分区，并建立文件系统；

LFS 假设根文件系统 (/) 采用 ext4 文件系统。输入以下命令在 LFS 分区创建一个 ext4 文件系统：

```
mkfs -v -t ext4 /dev/<xxx>
```

命令中 <xxx> 应该替换成 LFS 分区的名称。

如果您拥有一个现成的 swap分区，就不需要格式化它。如果新创建了一个 swap分区，需要执行以初始化它：

```
mkswap /dev/<yyy>
```

命令中 <yyy> 应该替换成 swap 分区的名称。

3） 设置环境变量并挂载新分区；

```
export LFS=/mnt/lfs
```

设置该环境变量的好处是，我们可以直接输入书中的命令，例如 **mkdir -v $LFS/tools**。Shell 在解析时会自动将 “$LFS” 替换成 “/mnt/lfs” (或是您设置的其他值)。

小心

无论何时，如果您离开并重新进入了工作环境，一定要确认 LFS 的设定值和您离开工作环境同。(例如，使用 **su** 切换到 root 或者其他用户时。) 请执行以下命令，检查 LFS 的设置是否正

```
echo $LFS
```

确认该命令的输出是您构建 LFS 的位置，如果您使用本书提供的例子，那么输出应该是 /lfs。如果输出不正确，使用前文给出的命令，将 $LFS 设置成正确的目录名。

4） 挂载新分区并启用 swap 分区；

```
mkdir -pv $LFS
mount -v -t ext4 /dev/<xxx> $LFS
```

将 <xxx> 替换成 LFS 分区的代号

如果您使用了 swap 分区，使用 **swapon** 命令启用它：

```
/sbin/swapon -v /dev/<zzz>
```

将 <zzz> 替换成 swap 分区的名称。

2. 软件包和补丁

1） 以 root 身份创建目录，写入权限；

```
mkdir -v $LFS/sources
```

下面为该目录添加写入权限和 sticky 标志。“Sticky” 标志使得即使有多个用户对该目录有写只有文件所有者能够删除其中的文件。输入以下命令，启用写入权限和 sticky 标志：

```
chmod -v a+wt $LFS/sources
```

存在多种获取构建 LFS 必须的软件包和补丁的方法：

2） wget https://mirror.bjtu.edu.cn/lfs/lfs-packages/10.0/wget-list 下载 wget-list 文件后下载软件包，其中有一个软件包 file-5.39.tar.gz 损坏，需要单独下载进 sources 目录；

```
wget --input-file=wget-list --continue --directory-prefix=$LFS/sources
```

另外，自 LFS 7.0 以来，本书提供一个单独的文件 md5sums，用来检查所有软件包的正确

3. 最后的准备工作

1） 创建目录布局；

以 root 身份，执行以下命令创建所需的目录布局：

```
mkdir -pv $LFS/{bin,etc,lib,sbin,usr,var}
case $(uname -m) in
  x86_64) mkdir -pv $LFS/lib64 ;;
esac
```

在第 6 章中，会使用交叉编译器编译程序 (细节参见工具链技术说明一节)。为了将这
序分离，它会被安装在一个专门的目录。执行以下命令创建该目录：

```
mkdir -pv $LFS/tools
```

2） 添加 LFS 用户，设置密码和访问权限，然后切换 lfs 用户；

了创建新用户，以 root 身份执行以下命令：

```
groupadd lfs
useradd -s /bin/bash -g lfs -m -k /dev/null lfs
```

命令行各选项的含义：

```
passwd lfs
```

将 lfs 设为 $LFS 中所有目录的所有者，使 lfs 对它们拥有完全访问权：

```
chown -v lfs $LFS/{usr,lib,var,etc,bin,sbin,tools}
case $(uname -m) in
  x86_64) chown -v lfs $LFS/lib64 ;;
esac
```

如果您按照本书的建议，建立了一个单独的工作目录，那么将这个目录的所有者也设为 lfs：

```
chown -v lfs $LFS/sources
```

注意

在某些宿主系统上，下面的命令不会正确完成，而会将 lfs 用户的登录会话挂起到后台。
符 “lfs:~$” 没有很快出现，输入 **fg** 命令以修复这个问题。

下面以 lfs 的身份登录。可以通过虚拟控制台或者显示管理器登录，也可以使用下面的命令切换用

```
su - lfs
```

参数 “-” 使得 **su** 启动一个登录 shell，而不是非登录 shell。您可以阅读 bash(1) 和 **info bash**
它们的区别。

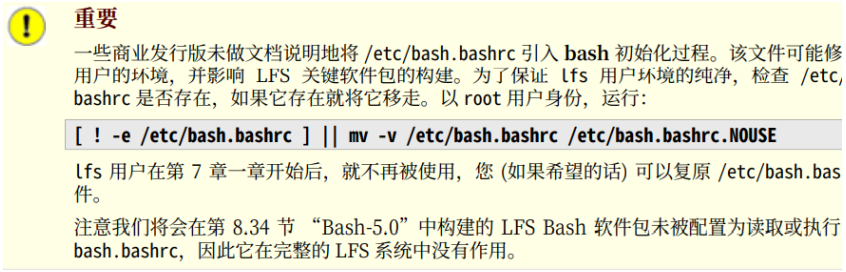
3） 配置环境，创建两个新的启动脚本；

```
cat > ~/.bash_profile << "EOF"
exec env -i HOME=$HOME TERM=$TERM PS1='\u:\w\$ ' /bin/bash
EOF
```

在以 lfs 用户登录时，初始的 shell 一般是一个登录 shell。它读取宿主系统的 /etc/profil
一些设置和环境变量)，然后读取 .bash_profile。我们在 .bash_profile 中使用 **exec env**
命令，新建一个除了 HOME, TERM 以及 PS1 外没有任何环境变量的 shell，替换当前 shell，防
必要和有潜在风险的环境变量进入编译环境。通过使用以上技巧，我们创建了一个干净环境

新的 shell 实例是 非登录 shell，它不会读取和执行 /etc/profile 或者 .bash_profile 的内
执行 .bashrc 文件。现在我们就创建一个 .bashrc 文件：

```
cat > ~/.bashrc << "EOF"
set +h
umask 022
LFS=/mnt/lfs
LC_ALL=POSIX
LFS_TGT=$(uname -m)-lfs-linux-gnu
PATH=/usr/bin
if [ ! -L /bin ]; then PATH=/bin:$PATH; fi
PATH=$LFS/tools/bin:$PATH
export LFS LC_ALL LFS_TGT PATH
EOF
```



重要

一些商业发行版未做文档说明地将 /etc/bash.bashrc 引入 bash 初始化过程。该文件可能修用户的环境，并影响 LFS 关键软件包的构建。为了保证 lfs 用户环境的纯净，检查 /etc/bashrc 是否存在，如果它存在就将它移走。以 root 用户身份，运行：

```
[ ! -e /etc/bash.bashrc ] || mv -v /etc/bash.bashrc /etc/bash.bashrc.NOUSE
```

lfs 用户在第 7 章一章开始后，就不再被使用，您 (如果希望的话) 可以复原 /etc/bash.bas件。

注意我们将会在第 8.34 节 “Bash-5.0”中构建的 LFS Bash 软件包未被配置为读取或执行 bash.bashrc，因此它在完整的 LFS 系统中没有作用。

最后，为了完全准备好编译临时工具的环境，指示 shell 读取刚才创建的配置文件：

```
source ~/.bash_profile
```

三、构建 LFS 交叉工具链和临时工具

1. 编译交叉工具链

   1） Binutils-2.35 - 第一遍

   2） GCC-10.2.0 - 第一遍

   进行到此时出现 C 无法编译的问题，需要检查最初的软件和版本是否正确，并安装缺少的软件，重试之后成功；

   3） Linux-5.8.5 API 头文件

   4） Glibc-2.32

   5） GCC-10.2.0 中的 Libstdc++，第一遍

2. 交叉编译临时工具

   1） M4-1.4.18

   2） Ncurses-6.2

   ……

   按照手册顺序安装软件包，进入第七章之前发现所有软件包安装载 root 用户下，原因是重启后未切换用户，后重新配置环境再次按手册顺序安装。这次惨痛的教训告诉我要仔细看手册，这个错误正是手册中强调的，而第一次没有仔细看导致无法继续进行，第二次仔细看手册后顺利通过。

3. 进入Chroot 并构建其他临时工具

   1） 改变所有者；

为了避免这样的问题，执行以下命令，将 $LFS/* 目录的所有者改变为 root：

```
chown -R root:root $LFS/{usr,lib,var,etc,bin,sbin,tools}
case $(uname -m) in
  x86_64) chown -R root:root $LFS/lib64 ;;
esac
```

2） 准备虚拟内核文件系统，创建初始设备节点，挂载和填充/dev，挂载虚拟内核文件系统；

首先创建这些文件系统的挂载点：

```
mkdir -pv $LFS/{dev,proc,sys,run}
```

7.3.1. 创建初始设备节点

在内核引导系统时，它需要一些设备节点，特别是 cons
样在内核填充 /dev 前，或者 Linux 使用 init=/bin/bas
建它们：

```
mknod -m 600 $LFS/dev/console c 5 1
mknod -m 666 $LFS/dev/null c 1 3
```

73

Linux

7.3.2. 挂载和填充 /dev

用设备文件填充 /dev 目录的推荐方法是挂载一个虚
设备被发现或访问时动态地创建设备文件。这个工作通常
统还没有 Udev，也没有被引导过，因此必须手工挂载
dev 目录就实现。绑定挂载是一种特殊挂载类型，它允
以下命令进行绑定挂载：

```
mount -v --bind /dev $LFS/dev
```

7.3.3. 挂载虚拟内核文件系统

现在挂载其余的虚拟内核文件系统：

```
mount -v --bind /dev/pts $LFS/dev/pts
mount -vt proc proc $LFS/proc
mount -vt sysfs sysfs $LFS/sys
mount -vt tmpfs tmpfs $LFS/run
```

```
if [ -h $LFS/dev/shm ]; then
  mkdir -pv $LFS/$(readlink $LFS/dev/shm)
fi
```

3） 进入 Chroot 环境；

```
chroot "$LFS" /usr/bin/env -i   \
    HOME=/root                  \
    TERM="$TERM"                \
    PS1='(lfs chroot) \u:\w\$ ' \
    PATH=/bin:/usr/bin:/sbin:/usr/sbin \
    /bin/bash --login +h
```

通过传递 -i 选项给 env 命令，可以清除 chroot 环境中的所有环

4） 创建目录；

```
mkdir -pv /{boot,home,mnt,opt,srv}
```

执行以下命令，为这些直接位于根目录中的目录创建次级目录结构：

```
mkdir -pv /etc/{opt,sysconfig}
mkdir -pv /lib/firmware
mkdir -pv /media/{floppy,cdrom}
mkdir -pv /usr/{,local/}{bin,include,lib,sbin,src}
mkdir -pv /usr/{,local/}share/{color,dict,doc,info,locale,man}
mkdir -pv /usr/{,local/}share/{misc,terminfo,zoneinfo}
mkdir -pv /usr/{,local/}share/man/man{1..8}
mkdir -pv /var/{cache,local,log,mail,opt,spool}
mkdir -pv /var/lib/{color,misc,locate}

ln -sfv /run /var/run
ln -sfv /run/lock /var/lock

install -dv -m 0750 /root
install -dv -m 1777 /tmp /var/tmp
```

默认情况下，新创建的目录具有权限码 755，但这并不适合所有目录。在以

5） 创建必要的文件和符号链接；

```
ln -sv /proc/self/mounts /etc/mtab
```

创建一个基本的 /etc/hosts 文件，一些测试套件，以及 Perl 的一个配置文件将会使

```
echo "127.0.0.1 localhost $(hostname)" > /etc/hosts
```

为了使得 root 能正常登录，而且用户名 “root” 能被正常识别，必须在文件 /etc/p
中写入相关的条目。

执行以下命令创建 /etc/passwd 文件：

```
cat > /etc/passwd << "EOF"
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/dev/null:/bin/false
daemon:x:6:6:Daemon User:/dev/null:/bin/false
messagebus:x:18:18:D-Bus Message Daemon User:/var/run/dbus:/bin/false
nobody:x:99:99:Unprivileged User:/dev/null:/bin/false
EOF
```

我们以后再设置 root 用户的实际密码。

```
cat > /etc/group << "EOF"
root:x:0:
bin:x:1:daemon
sys:x:2:
kmem:x:3:
tape:x:4:
tty:x:5:
daemon:x:6:
floppy:x:7:
disk:x:8:
lp:x:9:
dialout:x:10:
audio:x:11:
video:x:12:
utmp:x:13:
usb:x:14:
cdrom:x:15:
adm:x:16:
messagebus:x:18:
input:x:24:
mail:x:34:
kvm:x:61:
wheel:x:97:
nogroup:x:99:
users:x:999:
EOF
```

6） GCC-10.2.0 中的 Libstdc++，第二遍

……

按照顺序安装软件包，无问题。

7） 清理和备份临时系统（我没有备份，直接在第九章之前复制了虚拟机）

四、构建 LFS 系统

1. 安装基本系统软件

1） Man-pages-5.08

2） Tcl-8.6.10

……

几十个安装包用了很长时间，file-5.39 安装包损坏，去一开始的网址单独下载进 sources 目录重新安装解决。

3） 移除调试符号

```
save_lib="ld-2.32.so libc-2.32.so libpthread-2.32.so libthread_db-1.0.so"

cd /lib

for LIB in $save_lib; do
    objcopy --only-keep-debug $LIB $LIB.dbg
    strip --strip-unneeded $LIB
    objcopy --add-gnu-debuglink=$LIB.dbg $LIB
done

save_usrlib="libquadmath.so.0.0.0 libstdc++.so.6.0.28
             libitm.so.1.0.0 libatomic.so.1.2.0"

cd /usr/lib

for LIB in $save_usrlib; do
    objcopy --only-keep-debug $LIB $LIB.dbg
    strip --strip-unneeded $LIB
    objcopy --add-gnu-debuglink=$LIB.dbg $LIB
done

unset LIB save_lib save_usrlib
```

221

```
find /usr/lib -type f -name \*.a \
   -exec strip --strip-debug {} ';'

find /lib /usr/lib -type f -name \*.so* ! -name \*dbg \
   -exec strip --strip-unneeded {} ';'

find /{bin,sbin} /usr/{bin,sbin,libexec} -type f \
    -exec strip --strip-all {} ';'
```

4） 清理系统

```
rm -rf /tmp/*
```

现在需要登出，并使用新的 chroot 命令行重新进入 chroot 环境。从现在起，在退出并
境时，要使用下面的修改过的 chroot 命令：

```
logout

chroot "$LFS" /usr/bin/env -i          \
    HOME=/root TERM="$TERM"            \
    PS1='(lfs chroot) \u:\w\$ '        \
    PATH=/bin:/usr/bin:/sbin:/usr/sbin \
    /bin/bash --login
```

这里不再使用 +h 选项，因为所有之前安装的程序都已经替换成了最终版本，可以进行删

```
rm -f /usr/lib/lib{bfd,opcodes}.a
rm -f /usr/lib/libctf{,-nobfd}.a
rm -f /usr/lib/libbz2.a
rm -f /usr/lib/lib{com_err,e2p,ext2fs,ss}.a
rm -f /usr/lib/libltdl.a
rm -f /usr/lib/libfl.a
rm -f /usr/lib/libz.a
```

在 /usr/lib 和 /usr/libexec 目录中还有一些扩展名为 .la 的文件
经讨论过的，它们在链接到共享库，特别是使用 autotools 以外
行以下命令删除它们：

```
find /usr/lib /usr/libexec -name \*.la -delete
```

```
find /usr -depth -name $(uname -m)-lfs-linux-gnu\* | xargs rm -rf
```

/tools 也可以被删除，从而获得更多可用空间：

```
rm -rf /tools
```

最后，移除上一章开始时创建的临时 'tester' 用户账户。

```
userdel -r tester
```

2. 系统配置

1） 安装 LFS-Bootscripts-20200818；

2） 创建自定义 Udev 规则；

规则：

```
bash /lib/udev/init-net-rules.sh
```

现在检查文件 /etc/udev/rules.d/70-persistent-net.rules，确认网络设备与命名的

```
cat /etc/udev/rules.d/70-persistent-net.rules
```

3） CD-ROM 符号链接；

```
udevadm test /sys/block/hdd
```

观察包含一些 *_id 程序输出的行。“by-id” 模式在 ID_SERIAL 存在且非空
ID_MODEL 和 ID_REVISION 的组合。“by-path” 模式会使用 ID_PATH 的值。

如果默认模式不适合您的情况，可以像下面这样修改 /etc/udev/rules.d/83-cdro
(将 mode 替换成 “by-id” 或 “by-path” 中的一个)：

```
sed -e 's/"write_cd_rules"/"write_cd_rules mode"/' \
    -i /etc/udev/rules.d/83-cdrom-symlinks.rules
```

注意现在并不需要创建规则文件和符号链接，因为已经绑定挂载了宿主的 /dev 目

4） 处理重复设备（好像不需要）；

5） 一般网络配置，创建网络接口配置文件，/etc/resolv.conf 文件，配置系统主机名，.

自定义 /etc/hosts 文件；

例如，以下命令为 eth0 设备创建一个静态 IP 地址配置：

```
cd /etc/sysconfig/
cat > ifconfig.eth0 << "EOF"
ONBOOT=yes
IFACE=eth0
SERVICE=ipv4-static
IP=192.168.1.2
GATEWAY=192.168.1.1
PREFIX=24
BROADCAST=192.168.1.255
EOF
```

您必须修改每个文件中用斜体显示的设定值，使其与您的网络环

```
cat > /etc/resolv.conf << "EOF"
# Begin /etc/resolv.conf

domain <您的域名>
nameserver <您的主要域名服务器 IP 地址>
nameserver <您的次要域名服务器 IP 地址>

# End /etc/resolv.conf
EOF
```

```
echo "<lfs>" > /etc/hostname
```

<lfs> 需要被替换为赋予该计算机的名称。不要在这

```
cat > /etc/hosts << "EOF"
# Begin /etc/hosts

127.0.0.1 localhost.localdomain localhost
127.0.1.1 <FQDN> <HOSTNAME>
<192.168.1.1> <FQDN> <HOSTNAME> [alias1] [alias2 ...]
::1       localhost ip6-localhost ip6-loopback
ff02::1   ip6-allnodes
ff02::2   ip6-allrouters

# End /etc/hosts
EOF
```

6） System V 引导脚本使用与配置；

```
cat > /etc/inittab << "EOF"
# Begin /etc/inittab

id:3:initdefault:

si::sysinit:/etc/rc.d/init.d/rc S

l0:0:wait:/etc/rc.d/init.d/rc 0
l1:S1:wait:/etc/rc.d/init.d/rc 1
l2:2:wait:/etc/rc.d/init.d/rc 2
l3:3:wait:/etc/rc.d/init.d/rc 3
l4:4:wait:/etc/rc.d/init.d/rc 4
l5:5:wait:/etc/rc.d/init.d/rc 5
l6:6:wait:/etc/rc.d/init.d/rc 6

ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

su:S016:once:/sbin/sulogin

1:2345:respawn:/sbin/agetty --noclear tty1 9600
2:2345:respawn:/sbin/agetty tty2 9600
3:2345:respawn:/sbin/agetty tty3 9600
4:2345:respawn:/sbin/agetty tty4 9600
5:2345:respawn:/sbin/agetty tty5 9600
6:2345:respawn:/sbin/agetty tty6 9600

# End /etc/inittab
EOF
```

7） 配置系统时钟；

```
cat > /etc/sysconfig/clock << "EOF"
# Begin /etc/sysconfig/clock

UTC=1

# Set this to any options you might need to give to hwclock,
# such as machine hardware clock type for Alphas.
CLOCKPARAMS=

# End /etc/sysconfig/clock
EOF
```

8） Bash Shell 启动文件，创建/etc/profile文件；

Glibc 支持的所有 locale 可以用以下命令列出:

```
locale -a
```

字符映射可能有多个别名，例如 “ISO-8859-1” 也可以称为 “
序不能正确处理一些别名 (例如，“UTF-8”必须写作“UTF-8”
多数情况下，为了保险起见，最好使用 locale 的规范名称。为了
名> 替换成 **locale** -a 对于您希望的 locale 的输出 (以 “en_GB.i

```
LC_ALL=<locale 名> locale charmap
```

对于 “en_GB.iso88591” locale，以上命令输出:

```
ISO-8859-1
```

这样就最终确定 locale 应设置为 “en_GB.ISO-8859-1”。在将
动文件之前，一定要进行下列测试:

```
LC_ALL=<locale 名> locale language
LC_ALL=<locale 名> locale charmap
LC_ALL=<locale 名> locale int_curr_symbol
LC_ALL=<locale 名> locale int_prefix
```

以上命令应该输出语言名称，选定 locale 使用的字符编码，
际电话区号。如果以上某个命令失败并输出类似下面这样的消息

```
cat > /etc/profile << "EOF"
# Begin /etc/profile

export LANG=<ll>_<CC>.<charmap><@modifiers>

# End /etc/profile
EOF
```

9） 创建/etc/inputrc文件；

```
cat > /etc/inputrc << "EOF"
# Begin /etc/inputrc
# Modified by Chris Lynn <roryo@roryo.dynup.net>

# Allow the command prompt to wrap to the next line
set horizontal-scroll-mode Off

# Enable 8bit input
set meta-flag On
set input-meta On

# Turns off 8th bit stripping
set convert-meta Off

# Keep the 8th bit for display
set output-meta On

# none, visible or audible
set bell-style none

# All of the following map the escape sequence of the value
# contained in the 1st argument to the readline specific functions
"\eOd": backward-word
"\eOc": forward-word

# for linux console
"\e[1~": beginning-of-line
"\e[4~": end-of-line
"\e[5~": beginning-of-history
"\e[6~": end-of-history
"\e[3~": delete-char
"\e[2~": quoted-insert

# for xterm
"\eOH": beginning-of-line
"\eOF": end-of-line

# for Konsole
"\e[H": beginning-of-line
"\e[F": end-of-line

# End /etc/inputrc
EOF
```

10） 创建 /etc/shells 文件；

```
cat > /etc/shells << "EOF"
# Begin /etc/shells

/bin/sh
/bin/bash

# End /etc/shells
EOF
```

3. 使 LFS 系统可引导

此内容没有按照手册进行，创建/etc/fstab 文件，根据设置内容添加文件系统按顺序挂载；

1） Linux-5.8.5 安装；

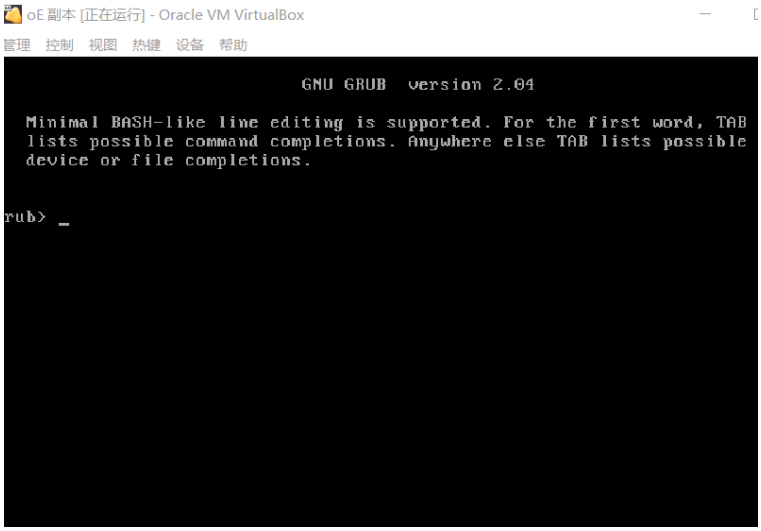
进入配置内核选项界面，根据 http://www.linuxfromscratch.org/~krejzi/basic-kernel.txt 中的内容选择内核选项，然后按照手册编译安装；

2） 配置 Linux 内核模块加载顺序；

```
install -v -m755 -d /etc/modprobe.d
cat > /etc/modprobe.d/usb.conf << "EOF"
# Begin /etc/modprobe.d/usb.conf

install ohci_hcd /sbin/modprobe ehci_hcd ; /sbin/modprobe -i ohci_hcd ; true
install uhci_hcd /sbin/modprobe ehci_hcd ; /sbin/modprobe -i uhci_hcd ; true

# End /etc/modprobe.d/usb.conf
EOF
```

3） 使用 GRUB 设定引导过程；

第一次按照手册 grub-install /dev/sda 将 grub 安装到了 sda 上，重启后发现出错，



发现必须安装到 sdb 才能启动，删除虚拟机副本后重新复制虚拟机，再次编译安装 linux 安装包，重新安装 grub 发现无法安装，加上 force 后 grub-install --force /dev/sdb 安装成功；grub-mkconfig 生成配置文件并将其中的

menuentry 'GNU/Linux' --class gnu-linux --class gnu --class os $menuentry_id_option 'gnulinux-simple-f6c7f95b-75a8-4d44-8abb-1039011fdaa6' {

load_video

insmod gzio

insmod part_gpt

insmod ext2

set root='hd1,gpt1'

if [x$feature_platform_search_hint = xy]; then

search --no-floppy --fs-uuid --set=root --hint-bios=hd1,gpt1 --hint-efi=hd1,gpt1 --hint-baremetal=ahci1,gpt1 f6c7f95b-75a8-4d44-8abb-1039011fdaa6

else

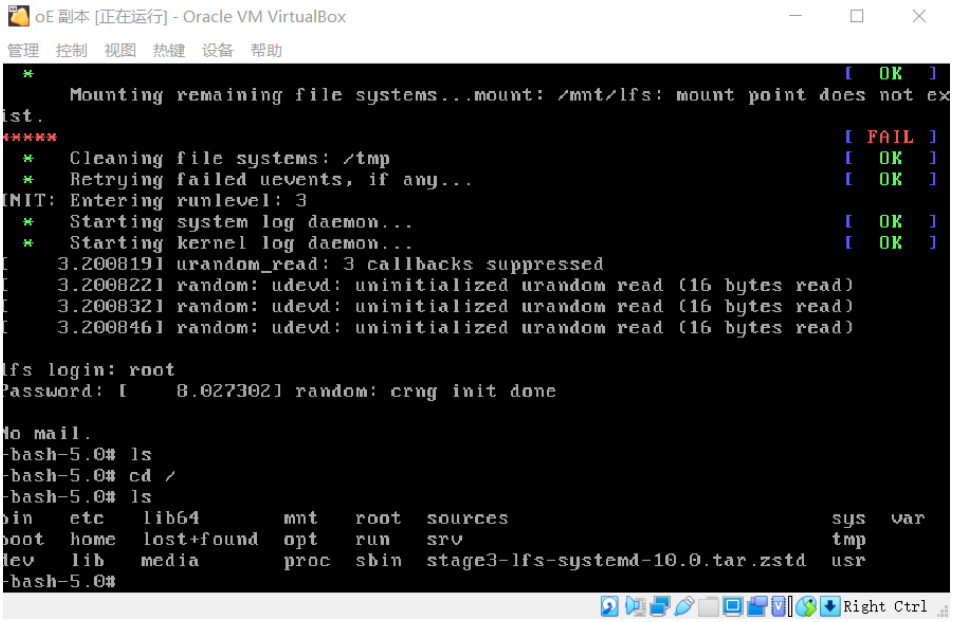
search --no-floppy --fs-uuid --set=root f6c7f95b-75a8-4d44-8abb-1039011fdaa6

```
        fi

        echo    'Loading Linux 5.8.5-lfs-SVN-20200901 ...'

        linux   /boot/vmlinuz-5.8.5-lfs-SVN-20200901 root=/dev/sdb1 ro
}
```

以及 advance 和 recovery 部分复制到/boot/grub2/grub.cfg 中对应的位置。

4. 重启

重启后进入 LFS 载入界面，发现 LFS 系统无法成功启动，错误是/mnt/lfs 挂载点不存在，



然后在宿主系统中找到对应路径发现/mnt/lfs 已存在，后来在老师提醒下发现是 lfs 系统中 lfs 目录不存在，进入 lfs 系统后在 mnt 目录下新建 lfs 目录，重启后问题解决，完成 lfs 搭建。