Wonderful Server

L.Y.　编

Created by XeLaTeX

2016 年 6 月 17 日

目 录

第 1 章

搭建本地镜像

§1.1 查看源使用帮助

对于 Debian 8.x 版本来说，163 源使用帮助如下。

```
deb http://mirrors.163.com/debian/ jessie main non-free contrib
deb http://mirrors.163.com/debian/ jessie-updates main non-free contrib
deb http://mirrors.163.com/debian/ jessie-backports main non-free contrib
deb-src http://mirrors.163.com/debian/ jessie main non-free contrib
deb-src http://mirrors.163.com/debian/ jessie-updates main non-free contrib
deb-src http://mirrors.163.com/debian/ jessie-backports main non-free contrib
deb http://mirrors.163.com/debian-security/ jessie/updates main non-free contrib
deb-src http://mirrors.163.com/debian-security/ jessie/updates main non-free contrib
```

其中，jessie-backports 部分可以不用下载，关于各个部分的意义见“有道笔记”中《Debian、Ubuntu 源列表说明》。

§1.2 下载源

安装好 apt-mirror 工具之后，配置/etc/apt/mirror.list 文件：

```
set base_path    /media/sf_share/Debian8.x_amd64_mirror

set mirror_path  $base_path/mirror
set skel_path    $base_path/skel
set var_path     $base_path/var
set cleanscript  $var_path/clean.sh
set defaultarch  <running host architecture>
# set postmirror_script $var_path/postmirror.sh
# set run_postmirror 0
set nthreads     20
set _tilde 0

deb-amd64 http://mirrors.163.com/debian/ jessie main non-free contrib
deb-amd64 http://mirrors.163.com/debian/ jessie-updates main non-free contrib
```

```
deb-amd64 http://mirrors.163.com/debian/ jessie-backports main non-free contrib
deb-amd64 http://mirrors.163.com/debian-security/ jessie/updates main non-free contr
ib

#clean http://ftp.cn.debian.org/debian
```

之后运行：apt-mirror 命令即可。对 Debian 8.x amd64 来说，大概有 70G 以上的内容，可用 tar 工具将其打包以便拷贝。

§1.3 配置源

修改/etc/apt/sources.list 文件：

```
deb file:/xxx/Debian8.x_amd64_mirror/mirror/mirrors.163.com/debian/          \
    jessie main contrib non-free
deb file:/xxx/Debian8.x_amd64_mirror/mirror/mirrors.163.com/debian/          \
    jessie-updates main non-free contrib
deb file:/xxx/Debian8.x_amd64_mirror/mirror/mirrors.163.com/debian/          \
    jessie-backports main non-free contrib
deb file:/xxx/Debian8.x_amd64_mirror/mirror/mirrors.163.com/debian-security/  \
    jessie/updates main non-free contrib
```

然后执行命令：apt-get update 即可。

Tips: 使用本地源执行 apt-get update 时，如出现“Release file expired”错误，执行如下命令进行更新操作：apt-get -o Acquire::Check-Valid-Until=false update。或者在/etc/apt/apt.conf.d/建立文件例如取名：10no–check-valid-until，然后写入内容：

```
Acquire::Check-Valid-Until "0";
```

将上述内容保存到文件：sources.list.dat 之中，配置过程可以使用脚本完成。

```
文件: 1.setSourcesList.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

echo -e "\nYong's 脚本: 备份 sources.list, 设置本地源\n"

/bin/bash ./backupFile.sh /etc/apt/sources.list
cp sources.list.dat /etc/apt/sources.list

apt-get -o Acquire::Check-Valid-Until=false update && apt-get upgrade
```

该脚本调用的 backupFile.sh 用来备份文件，

```
文件: backupFile.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

fileName=$1
```

```
cp -p $1{,.$(date +%F-%H-%M-%S).backup}
```

第 2 章

PureFTP

§2.1 安装配置

在 Debian 中，PureFTP 通过/etc/pure-ftpd/conf/目录中的配置文件来确定启动参数。也可以通过查询启动参数的意义，直接采用命令启动，不过最好还是配置这些文件，通过 PureFTP 的内置转换工具，自动将配置文件转换为启动参数。

参数的意义参见官方手册，这里通过脚本来完成配置。

> **Tips:** 官方手册中除了包含编译、使用等内容以外，还有一个常见问题列表，这个列表非常值得参考，例如如何设置给用户设置共享文件夹等。

文件: 5.ftpSet.sh

```
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

echo -e "\n 安装 pureFTP"

apt-get install pure-ftpd pure-ftpd-common -y

# 配置 pureFTP
echo -e "\n 配置 pureFTP"
confDir=/etc/pure-ftpd/conf

# 默认已禁止匿名登录
echo "yes" > $confDir/ChrootEveryone
echo "yes" > $confDir/UnixAuthentication
echo "yes" > $confDir/ProhibitDotFilesWrite
echo "yes" > $confDir/ProhibitDotFilesRead
echo "yes" > $confDir/DontResolve

echo "5"  > $confDir/MaxClientsPerIP
echo "20" > $confDir/MaxIdleTime
echo "98" > $confDir/MaxDiskUsage
```

```
echo "117 117" > $confDir/Umask

echo -e "\n 重启 pure-ftpd 服务"
systemctl restart pure-ftpd
```

§2.2 设置共享目录

在 chroot 开启的时候如何给 FTP 用户设置共享目录呢？这个问题官方的常见问题手册给出了回答。PureFTP 源码编译时，可以选择虚拟 chroot 选项，这种情形下，可以通过创建软链接：ln -s 来实现共享 (注意使用绝对路径)。但是，Debian 官方仓库中并未开启虚拟 chroot 选项。另一种更好的方法是，将共享目录挂载到用户目录下，命令为：mount --bind。可写到/etc/fstab 文件中，实现开机挂载。

§2.3 虚拟用户

PureFTP 的用户不一定非得是 Linux 用户，可以通过设定一个专门的 FTP Linux 账户，然后所有虚拟用户通过该专门的 Linux 账户访问 FTP，具体命令参见官方手册。

增加虚拟、修改虚拟账户的命令是：pure-pw，其后跟的命令和 Linux 账户配置是一样的。存储虚拟账户的文件格式为：

```
<account>:<password>:<uid>:<gid>:<gecos>:<home>:...
```

该文件包含了用户使用 FTP 的宽带等设置，其中，account、password、uid、gid、home 为非空字段。密码的加密方式和系统账户是一样的，home 字段若以“/./”结尾则表示 chroot。添加虚拟用户后，Pure-FTP 的命令可将该文件转化为二进制的 PureDB，该文件是二进制文件，经过了排序处理，在有大量虚拟用户的情况下，可以加快处理速度。也可以使用内置命令直接复制系统账户。

> **Tips:** 须查看 Pure-FTP 的配置文件,确认配置文件中默认设置的认证文件位置。例如,在 Debian 7.x 中,须做一下如下操作:`cd /etc/pure-ftpd/auth`,然后 `ln -s ../conf/PureDB`, 否则虚拟账户会无法进行认证，可以查看 log 确认问题所在。

第 3 章

文件服务器

文件服务器选定的系统版本是 Debian 8.x amd64。

§3.1 设置本地源

在离线环境下，为了安装软件方便，需要使用本地源，具体设置参见第 1 章。

§3.2 安装常用软件

安装编译环境、Git、Vim、ctags、sudo 等软件包。

```
文件: 2.installPackages.sh
#!/bin/bash

echo -e "\nYong's 脚本: 安装 编译环境、Git、Vim、ctags、sudo 等软件包\n"



PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

apt-get install build-essential vim -y
apt-get install ctags -y
apt-get install git git-gui -y
apt-get install sudo -y
```

§3.3 配置 root 使用环境

主要设置 bash 环境、Vim 环境等。

```
文件: 3.setRootEnv.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

echo -e "\nYong's 脚本: 设置 root 的用户环境\n"
```

```
homeDir=/root
chmod 750 $homeDir

# 创建 vim 配置文件目录
if [ ! -d "/root/.vim" ]
then
        mkdir $homeDir/.vim
fi

echo -e "Yong's 脚本: 复制 Vim 插件\n"

cp -av VimPlugin/* $homeDir/.vim/
cp -v Vim.dat $homeDir/.vimrc


echo -e "\nYong's 脚本: 在 /etc/profile 中设置 Intel 编译器的环境变量"

cat /etc/profile | grep -q "source /opt/intel"

if [ $? -ne 0 ]
then
        /bin/bash ./backupFile.sh /etc/profile
        cat profile.dat >> /etc/profile
fi

echo -e "\nYong's 脚本: 设置 .bashrc"

cat $homeDir/.bashrc | grep -q ".mybashset"

if [ $? -ne 0 ]
then
        cp -v rootBash.dat $homeDir/.mybashset
        /bin/bash ./backupFile.sh $homeDir/.bashrc
        echo "source $homeDir/.mybashset" >> $homeDir/.bashrc
fi
```

脚本中调用的其他文件是，

```
文件: Vim.dat
" 显示行号
set number

" 显示光标当前位置
set ruler

" 总是显示状态栏
set laststatus=2

" 高亮显示当前行/列
"set cursorline
```

```
"set cursorcolumn

" 高亮显示搜索结果
set hlsearch

" 查找前预览
set incsearch

" 搜索时大小写不敏感
set ignorecase

" 关闭兼容模式
set nocompatible

" 禁止光标闪烁,gVim 有效
"set gcr=a:block-blinkon0

" 缩进方式
set tabstop=4
set softtabstop=4
set shiftwidth=4

" 基于缩进或语法进行代码折叠
"set foldmethod=indent
set foldmethod=syntax
" 启动 vim 时关闭折叠代码
set nofoldenable

" 配色方案
colorscheme koehler

" 开启文件类型侦测
filetype on

" 根据侦测到的不同类型加载对应的插件
filetype plugin on


" 语法高亮
if &t_Co > 2 || has("gui_running")
        syntax on
        set hlsearch
endif

" 插件设置
map <F4> :NERDTreeToggle<CR>
map <F2> :TlistToggle<CR>
let Tlist_Use_Right_Window=1
```

```
nmap <C-H> <C-W>h " control+h 进入左边的窗口
nmap <C-J> <C-W>j " control+j 进入下边的窗口
nmap <C-K> <C-W>k " control+k 进入上边的窗口
nmap <C-L> <C-W>l " control+l 进入右边的窗口

" 对齐插件设置，不是特别灵敏
let mapleader=";"
nmap <Leader>t& :Tab /&<CR>
vmap <Leader>t& :Tab /&<CR>
nmap <Leader>t= :Tab /=<CR>
vmap <Leader>t= :Tab /=<CR>
nmap <Leader>t" :Tab /"<CR>
vmap <Leader>t" :Tab /"<CR>
nmap <Leader>t: :Tab /:<CR>
vmap <Leader>t: :Tab /:<CR>
nmap <Leader>t:: :Tab /:\zs<CR>
vmap <Leader>t:: :Tab /:\zs<CR>
nmap <Leader>t, :Tab /,<CR>
vmap <Leader>t, :Tab /,<CR>
nmap <Leader>t,, :Tab /,\zs<CR>
vmap <Leader>t,, :Tab /,\zs<CR>
nmap <Leader>t<Bar> :Tab /<Bar><CR>
vmap <Leader>t<Bar> :Tab /<Bar><CR>
```

包含 Intel 编译器的使用环境。

```
文件: profile.dat
source /opt/intel/bin/compilervars.sh intel64
source /opt/intel/impi/5.0.1.035/bin64/mpivars.sh
source /opt/intel/mkl/bin/mklvars.sh intel64 ilp64
```

root 的 Bash 环境。

```
文件: rootBash.dat
umask 0007

export LS_OPTIONS='--color=auto'
eval "`dircolors`"
alias ls='ls $LS_OPTIONS'
alias ll='ls $LS_OPTIONS -l'
alias l='ls $LS_OPTIONS -lA'

export GIT_EDITOR=vim
```

§3.4 增加用户

增加用户、设置用户使用环境等。

```
文件: 4.addUsers.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH
```

```
# 增加用户组
teamName=ZhangShuHaiTeam
groupadd $teamName

# 用户
accounts=$(cat accounts.dat)

# HOME 目录
homeDir=/home

# 创建共享目录
shareDir=/home/TeamShare
mkdir $shareDir
chown root:$teamName $shareDir
chmod 2770 $shareDir

/bin/bash ./backupFile.sh /etc/fstab

for data in $accounts
do
        userName=$(echo $data | cut -d ":" -f 1)
        initPasswd=$(echo $data | cut -d ":" -f 2)

        # 创建用户
        echo "Add User: " $userName
        useradd -G $teamName -m -s /bin/bash $userName
        echo $userName:$initPasswd | chpasswd

        # 修改主文件目录权限
        if [ -d $homeDir/$userName ]
        then
                echo "Set HOMEDIR 750: " $homeDir/$userName
                chmod 750 $homeDir/$userName
        fi

        /bin/bash ./4.1.setuserEnv.sh $userName
        /bin/bash ./4.2.shareDir.sh   $userName
done

mount -a

/bin/bash ./4.3.setACL.sh

echo -e "\nAll Done!"
```

其中，accounts.dat 每一行格式是：“用户名: 密码”。

设置用户的 Bash、Vim 环境等。

文件: 4.1.setuserEnv.sh

```
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

echo -e "\nYong's 脚本: 设置 $1 的用户环境\n"

userName=$1
homeDir=/home/$userName

# 创建 Vim 插件目录
if [ ! -d "$homeDir/.vim" ]
then
        mkdir $homeDir/.vim
fi

# 复制 Vim 插件
echo -e "Yong's 脚本: 复制 Vim 插件\n"

cp -av VimPlugin/* $homeDir/.vim/
chown -R $userName:$userName $homeDir/.vim

# 复制 Vim 配置文件
cp -v Vim.dat $homeDir/.vimrc
chown $userName:$userName $homeDir/.vimrc

# 复制 bash 配置文件
echo -e "\nYong's 脚本: 设置 .bashrc"

/bin/bash ./backupFile.sh $homeDir/.bashrc
cp -v Bash.dat $homeDir/.mybashset
chown $userName:$userName $homeDir/.mybashset

cat $homeDir/.bashrc | grep -q ".mybashset"

if [ $? -ne 0 ]
then
        echo "source $homeDir/.mybashset" >> $homeDir/.bashrc
fi

# 禁止图形界面登陆
/bin/bash ./backupFile.sh /etc/pam.d/gdm-password

echo "auth    required    pam_succeed_if.so user != $1 quiet_success" >> /etc/pam.d/gdm-password
```

文件: bash.dat

```
umask 0007

export LS_OPTIONS='--color=auto'
```

```
eval "`dircolors`"
alias ls='ls $LS_OPTIONS'
alias ll='ls $LS_OPTIONS -l'
alias l='ls $LS_OPTIONS -lA'

export GIT_EDITOR=vim
```

FTP 的共享文件夹设置。

```
文件: 4.2.shareDir.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

echo -e "\nYong's 脚本: 设置 $1 的用户环境\n"

useName=$1
homeDir=/home/$useName

shareDir=$homeDir/TeamShare

# 创建共享目录
if [ ! -d "$shareDir" ]
then
        mkdir $shareDir
        chown $useName:$useName $shareDir
else
        echo -e "\nYong's 脚本: $shareDir 目录已存在\n"
fi

# 开机挂载
echo "/home/TeamShare        $shareDir        none        defaults,bind        0        0" \
        >> /etc/fstab
```

某个特殊用户的 ACL 权限设置。

```
文件: 4.3.setACL.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

# 用户
accounts=$(cat accounts.dat)

# HOME 目录
homeDir=/home

superUser= 特殊用户名

for data in $accounts
do
        userName=$(echo $data | cut -d ":" -f 1)
```

```
        if [ $userName != $superUser ]
        then
                setfacl -m u:$superUser:rx        $homeDir/$userName
        fi

done

echo -e "\nYong's 脚本: ACL 权限设置完毕\n"
```

§3.5 FTP

在对比了几种 FTP 之后，选择 PureFTP，安装配置见第 2 章。

§3.6 防火墙

配置防火墙，并设置开机启动。

```
文件: 6.setFirewallService.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

basepath=/root/FirewallScript # 脚本放置目录

if [ -f "$basepath" ]; then
        echo $basepath " 是一个文件，与要创建的文件夹重名"
        exit 1
fi

if [ ! -d "$basepath" ]; then
        mkdir -p $basepath
fi

cp -p FirewallScript/* $basepath

cat $basepath/Firewall.service.dat > /etc/systemd/system/Firewall.service

systemctl enable Firewall
systemctl start Firewall
```

配置 iptables 的脚本。

```
文件: FirewallScript/Firewall.sh
#!/bin/bash

# 请先输入您的相关参数，不要输入错误了!
EXTIF="eth0" # 这个是可以连上 Public IP 的网络接口
INIF=""      # 内部 LAN 的连接接口；若无则写成 INIF=""
#INNET="192.168.100.0/24" # 若无内部网域接口，请填写成 INNET=""
```

```
INNET=""                                    # 若无内部网域接口，请填写成 INNET=""
export EXTIF INIF INNET


# 1. 清除规则、设定默认政策及开放 lo 与相关的设定值
PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH
iptables -F
iptables -X
iptables -Z
iptables -P INPUT   DROP        # 默认规则设为拒绝，清除设置的时候要小心
iptables -P OUTPUT  ACCEPT
iptables -P FORWARD ACCEPT
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

basepath=/root/FirewallScript # 脚本放置目录

# 2. 启动额外的防火墙 script 模块
if [ -f "$basepath/deny.sh" ]; then
        /bin/bash $basepath/deny.sh        # 阻止某些网络进入
fi
if [ -f "$basepath/allow.sh" ]; then
        /bin/bash $basepath/allow.sh  # 运行某些网络进入
fi

# 3. 允许某些类型的 ICMP 封包进入
# AICMP="0 3 3/4 4 11 12 14 16 18"
# for tyicmp in $AICMP
# do
#         iptables -A INPUT -i $EXTIF -p icmp --icmp-type $tyicmp -j ACCEPT
# done

# 4. 允许某些服务的进入，请依照你自己的环境开启
# iptables -A INPUT -p TCP -i $EXTIF --dport  21 --sport 1024:65534 -j ACCEPT # FTP
# iptables -A INPUT -p TCP -i $EXTIF --dport  22 --sport 1024:65534 -j ACCEPT # SSH
# iptables -A INPUT -p TCP -i $EXTIF --dport  25 --sport 1024:65534 -j ACCEPT # SMTP
# iptables -A INPUT -p UDP -i $EXTIF --dport  53 --sport 1024:65534 -j ACCEPT # DNS
# iptables -A INPUT -p TCP -i $EXTIF --dport  53 --sport 1024:65534 -j ACCEPT # DNS
# iptables -A INPUT -p TCP -i $EXTIF --dport  80 --sport 1024:65534 -j ACCEPT # WWW
# iptables -A INPUT -p TCP -i $EXTIF --dport 110 --sport 1024:65534 -j ACCEPT # POP3
# iptables -A INPUT -p TCP -i $EXTIF --dport 443 --sport 1024:65534 -j ACCEPT # HTTPS


# 5. 最终将这些功能储存下来吧!
iptables-save
```

设置允许通过防火墙的 IP。

```
文件: FirewallScript/allow.sh
#!/bin/bash
```

```
basepath=/root/FirewallScript # 脚本放置目录

ipAllow=$(cat $basepath/ipAllow.dat)

for userIP in $ipAllow
do
        iptables -A INPUT -i $EXTIF -s $userIP -j ACCEPT
        echo $userIP
done
```

设置阻止通过防火墙的 IP。

```
文件: FirewallScript/deny.sh
#!/bin/bash
basepath=/root/FirewallScript # 脚本放置目录

ipDeny=$(cat $basepath/ipDeny.dat)

for userIP in $ipDeny
do
        iptables -A INPUT -i $EXTIF -s $userIP -j DROP
done
```

其中，ipAllow.dat 和 ipDeny.dat 分别包含了允许通过的 IP 和阻止通过的 IP。

Tips: 在清空防火墙的时候，需要注意事先将默认策略改为 ACCEPT。命令为：iptables -P INPUT ACCEPT。

§3.7 限制使用 su 和 sudo

设置部分用户可以使用某些系统命令。

```
文件: 7.suAndsudo.sh
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

# 限制使用部分 sudo 命令
/bin/bash ./backupFile.sh /etc/sudoers
cat sudoers.dat >> /etc/sudoers

# 限制使用 su
addgroup wheel
usermod -a -G wheel yong        # 加入 wheel 组的用户可以使用 su 命令

/bin/bash ./backupFile.sh /etc/pam.d/su
cat su.dat >> /etc/pam.d/su
```

该文件各项意义参见《鸟哥 Linux 私房菜——基础篇》。

```
文件: sudoers.dat
% 用户组名 ALL=/sbin/poweroff,/sbin/reboot
```

/etc/pam.d/su 文件的配置可以参见该文件本身的注释说明，另外可参考《鸟哥 Linux 私房菜——基础篇》的“PAM 模块简介”部分。

文件：su.dat

```
auth        required        pam_wheel.so
```

§3.8 删除系统自带账户

作为文件服务器，有一些系统账户用不到，可以删除。

文件：8.delSysUserAndGroup.sh

```
#!/bin/bash

PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH

delUsers="lp sync news uucp games"
delGroups="lp news uucp games dip"

for userName in $delUsers
do
        echo "Delete User: " $userName
        userdel $userName
done

echo -e "Delete users done.\n"

for groupName in $delGroups
do
        echo "Delete Group: " $groupName
        groupdel $groupName
done

echo -e "Delete groups done.\n"
```

§3.9 温度监控

CPU 等温度监控，安装 lm-sensors 模块。

- 安装：apt-get install lm-sensors -y
- 探测：sensors-detect
- 查看：sensors

硬盘温度监控，安装 hddtemp 模块。详见“有道笔记”中《hddtemp》。

- 安装：apt-get install hddtemp
- 探测：hddtemp /dev/sda /dev/sdb ... ...

第 4 章

Intel Parallel XE Studio

§4.1 安装 XE

这里选择 Intel Parallel XE Studio Cluster Edtion，因为该版本包含了 MPI 库的内容。官网上可以申请一年使用的 License，为了方便这里选择 2015 版本，虽然不是最新版本，但是省去了一年一申的麻烦。

查看安装帮助可知，安装可以使用静默方式，事先用虚拟机安装一遍，可以得到配置文件，安装命令为：./install.sh -d ./autoInstall.cfg。该文件内容为：

```
文件: autoInstall.cfg
ACCEPT_EULA=accept
INSTALL_MODE=NONRPM
CONTINUE_WITH_OPTIONAL_ERROR=yes
PSET_INSTALL_DIR=/opt/intel
CONTINUE_WITH_INSTALLDIR_OVERWRITE=yes
PSET_MODE=install
ACTIVATION_LICENSE_FILE=/media/sf_share/XE2015/Crack/zwt.lic
ACTIVATION_TYPE=license_file
AMPLIFIER_SAMPLING_DRIVER_INSTALL_TYPE=build
AMPLIFIER_DRIVER_ACCESS_GROUP=vtune
AMPLIFIER_DRIVER_PERMISSIONS=666
AMPLIFIER_LOAD_DRIVER=yes
AMPLIFIER_C_COMPILER=/usr/bin//gcc
AMPLIFIER_KERNEL_SRC_DIR=/lib/modules/3.16.0-4-amd64/build
AMPLIFIER_MAKE_COMMAND=/usr/bin//make
AMPLIFIER_INSTALL_BOOT_SCRIPT=yes
AMPLIFIER_DRIVER_PER_USER_MODE=no
ENVIRONMENT_LD_SO_CONF=no
MPSS_RESTART_STACK=no
PHONEHOME_SEND_USAGE_DATA=no
COMPONENTS=;intel-mpi-rt__noarch;... ... 一系列组件
```

接下来就可以使用该文件进行静默安装，命令为：./install.sh -s autoInstall.cfg。这样就省去了以后安装需要交互操作的麻烦。

安装完毕后，还需要配置用户环境。将以下内容加入/etc/profile 文件中即可。

```
source /opt/intel/bin/compilervars.sh intel64
source /opt/intel/impi/5.0.1.035/bin64/mpivars.sh
source /opt/intel/mkl/bin/mklvars.sh intel64 ilp64
```

§4.2 Fortran 测试

接下来对一个简单的 Fortran 程序进行并行测试。

```
文件: hello.f90
program main
use mpi
implicit none

        integer ::  id, err, comm, psize

        call MPI_INIT( err )
        call MPI_COMM_DUP( MPI_COMM_WORLD, comm, err )
        call MPI_COMM_RANK( comm, id, err )
        call MPI_COMM_SIZE( comm, psize, err )

        write(*,*) id, psize,  " hello"

        call MPI_BARRIER( comm, err )
        call MPI_FINALIZE( err )
end program
```

编译：

```
mpiifort -o main hello.f90
```

执行：

```
mpirun -np 4 ./main
```

这里需要注意可执行文件的路径，即这里的“./”不要漏了。

运行结果：

```
           3           4  hello
           0           4  hello
           2           4  hello
           1           4  hello
```

第 5 章

邮件设置

这里使用的软件为 msmtp，该软件小巧且易于配置。邮箱客户端软件用的是 mutt，该软件同样小巧，但是功能强大。

> **注意：**一般系统提供了 mail、mutt、sendmail 等工具，有时候系统提供的 sendmail 工具指向了其它的 mail 服务软件，例如在 Debian 8.3 中，sendmail 是 exim4 这个软件的一个软链接，而 mail 命令来自于 bsd-mailx。

§5.1 mail 配置

mail 会读取/etc/目录下的 mail.rc 文件，或者读取用户家目录下的.mailrc 文件。但是 mail 似乎不能自定义配置文件的位置，在某些情况下会导致不便。可以在 mail.rc 文件中增加一行，使用 msmtp 发送邮件。

```
set sendmail="/usr/bin/msmtp"
```

§5.2 msmtp 配置文件

msmtp 会读取/etc/目录下的 msmtprc 文件，或者读取用户家目录下的.msmtprc 文件，也可以用-C 参数指定配置文件。一个示例如下：

```
account default
host smtp.126.com                       <-- smtp 服务器
port 25                                 <-- smtp 服务器端口
from xxx@xxx.com
auth login
user xxx@xxx.com                        <-- 邮箱
password xxxx                           <-- 密码
logfile /root/Mail/Log/msmtp.log        <-- 日志位置
```

§5.3 mutt 配置

mutt 会读取/etc/目录下的 Muttrc 文件，或读取用户家目录下的.muttrc 文件，也可以用-F 参数指定配置文件。一个示例如下：

```
set from="xxx@xxx.com"                    <-- 发件人
set sendmail="/usr/bin/msmtp"             <-- 使用 msmtp 发送邮件
set use_from=yes
set realname=" 张三"
set editor="vim"                          <-- 发送邮件使用的文本编辑器
```

在使用 nagios 的邮件警报功能时，可以将配置文件放在一个特定的目录下，稍微修改一下上述参数即可。

```
set from="xxx@xxx.com"
set sendmail="/usr/bin/msmtp -C /etc/nagios3/.msmtprc"
set use_from=yes
set realname="Nagios3 警报"
set editor="vim"
```

然后将发送邮件的命令改为：`/usr/bin/mutt -F /etc/nagios3/.muttrc`。不过需要注意的是，要保证两份配置文件对 nagios 进程的拥有者是可读取的，例如默认情况下，Debian 8.x 中 nagios 是以 nagios 用户和 nagios 组权限执行的。当然也可以将配置文件放置在 nagios 进程的拥有者的家目录底下，此时使用 mail 命令也可。

第 6 章

日志

系统日志详细的知识参考《鸟哥的 Linux 私房菜——基础篇》相应的部分。

§6.1 知识要点

§6.1.1 日志文件意义

- **/var/log/boot.log**：当前这次开机的启动信息。
- **/var/log/cron**：与 crontab 日程相关。
- **/var/log/dmesg**：开机核心信息。
- **/var/log/lastlog**：二进制文件，记录系统上面所有的账号最近一次登入系统时的相关信息。使用 lastlog 命令读取。
- **/var/log/maillog** 或 **/var/log/mail/***：邮件相关日志。
- **/var/log/messages**：几乎系统发生的错误讯息 (或者是重要的信息) 都会记录在这个档案中。
- **/var/log/secure**：基本上，只要牵涉到『需要输入账号密码』的软件，那么当登入时 (不管登入正确或错误) 都会被记录在此档案中。包括系统的 login 程序、图形接口登入所使用的 gdm 程序、su, sudo 等程序、还有网络联机的 ssh, telnet 等程序，登入信息都会被记载在这里。
- **/var/log/wtmp**, **/var/log/faillog**：二进制文件，这两个档案可以记录正确登入系统者的帐户信息 (wtmp) 与错误登入时所使用的帐户信息 (faillog) ！使用 last 命令读取。
- **/var/log/httpd/***, **/var/log/samba/***：不同的网络服务会使用它们自己的登录档案来记载它们自己产生的各项讯息！上述的目录内则是个别服务所制订的登录档。

§6.1.2 日志文件格式

- 事件发生的日期与时间；
- 发生此事件的主机名；
- 启动此事件的服务名称 (如 systemd, CROND 等) 或指令与函式名称 (如 su, login..)；
- 该讯息的实际数据内容。

§6.1.3 服务名称

相对序号	服务类别	说明
0	kern(kernel)	就是核心 (kernel) 产生的讯息，大部分都是硬件侦测以及核心功能的启用；
1	user	在用户层级所产生的信息，例如后续会介绍到的用户使用 logger 指令来记录登录文件的功能；
2	mail	只要与邮件收发有关的讯息记录都属于这个；
3	daemon	主要是系统的服务所产生的信息，例如 systemd 就是这个有关的讯息！
4	auth	主要与认证/授权有关的机制，例如 login, ssh, su 等需要账号/密码的咚咚；
5	syslog	就是由 syslog 相关协议产生的信息，其实就是 rsyslogd 这支程序本身产生的信息啊！
6	lpr	亦即是打印相关的讯息啊！
7	news	与新闻组服务器有关的东西；
8	uucp	全名为 Unix to Unix Copy Protocol，早期用于 unix 系统间的程序数据交换；
9	cron	就是例行性工作排程 cron/at 等产生讯息记录的地方；
10	authpriv	与 auth 类似，但记录较多账号私人的信息，包括 pam 模块的运作等！
11	ftp	与 FTP 通讯协议有关的讯息输出！
16 23	local0 local7	保留给本机用户使用的一些登录文件讯息，较常与终端机互动。

§6.1.4 信息等级

等级数值	等级名称	说明
7	debug	用来 debug (除错) 时产生的讯息数据；
6	info	仅是一些基本的讯息说明而已；
5	notice	虽然是正常信息，但比 info 还需要被注意到的一些信息内容；
4	warning (warn)	警示的讯息，可能有问题，但是还不至于影响到某个 daemon 运作的信息；基本上，info, notice, warn 这三个讯息都是在告知一些基本信息而已，应该还不至于造成一些系统运作困扰；
3	err (error)	一些重大的错误讯息，例如配置文件的某些设定值造成该服务服法启动的信息说明，通常藉由 err 的错误告知，应该可以了解到该服务无法启动的问题呢！

等级数值	等级名称	说明
2	crit	比 error 还要严重的错误信息，这个 crit 是临界点 (critical) 的缩写，这个错误已经很严重了喔！
1	alert	警告警告，已经很有问题的等级，比 crit 还要严重！
0	emerg (panic)	疼痛等级，意指系统已经几乎要当机的状态！很严重的错误信息了。通常大概只有硬件出问题，导致整个核心无法顺利运作，就会出现这样的等级的讯息吧！

§6.1.5 logrotate 配置文件

- /etc/logrotate.conf
- /etc/logrotate.d/

§6.2 Debian 8.x 日志

Debian 8.x 系统的/etc/rsyslog.conf 文件内容如下：

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022

$WorkDirectory /var/spool/rsyslog

$IncludeConfig /etc/rsyslog.d/*.conf

auth,authpriv.*                         /var/log/auth.log
*.*;auth,authpriv.none                  -/var/log/syslog
daemon.*                              -/var/log/daemon.log
kern.*                                  -/var/log/kern.log
lpr.*                                  -/var/log/lpr.log
mail.*                                  -/var/log/mail.log
user.*                                  -/var/log/user.log

mail.info                              -/var/log/mail.info
mail.warn                              -/var/log/mail.warn
mail.err                              /var/log/mail.err

news.crit                              /var/log/news/news.crit
news.err                              /var/log/news/news.err
news.notice                             -/var/log/news/news.notice

*.=debug;\
        auth,authpriv.none;\
```

```
        news.none;mail.none        -/var/log/debug
*.=info;*.=notice;*.=warn;\
        auth,authpriv.none;\
        cron,daemon.none;\
        mail,news.none                -/var/log/messages

*.emerg                                  :omusrmsg:*


daemon.*;mail.*;\
        news.err;\
        *.=debug;*.=info;\
        *.=notice;*.=warn        |/dev/xconsole
```

这里需要注意的是:**/var/log/auth.log** 记录了用户认证登录的敏感信息,而不是/var/log/secure 文件。

§6.3 保护日志文件

若系统被入侵，/var/log/目录可能会被恶意删除。可以采取将日志文件定时备份到远程系统等措施，防止查找不到攻击源。另外，可以将 bash 的 history 记录备份。

第 7 章

安全防护

§7.1 防暴力破解

fail2ban 是 Linux 上的一个著名的入侵保护的开源框架，它会监控多个系统的日志文件（例如：/var/log/auth.log 或者/var/log/secure）并根据检测到的任何可疑的行为自动触发不同的防御动作，可以用来防范如 ssh、ftp 等被暴力破解。

Debian 8.x 安装 fail2ban 后，配置文件位置在/etc/fail2ban 目录下：

```
drwxr-xr-x 2 root root  4096 5 月  30 21:54 action.d        <-- 定义动作
-rw-r--r-- 1 root root  1525 3 月  15  2014 fail2ban.conf
drwxr-xr-x 2 root root  4096 3 月  19  2014 fail2ban.d
drwxr-xr-x 2 root root  4096 5 月  30 21:23 filter.d        <-- 定义日志分析策略
-rw-r--r-- 1 root root 13883 5 月  30 21:48 jail.conf       <-- 设置需要保护的服务
drwxr-xr-x 2 root root  4096 3 月  19  2014 jail.d
```

一般情况下，我们只需要修改 jail.conf 即可，或者在 jail.d 自定义。jail.conf 文件最开头的 [DEFAULT] 部分是全局设置，可以在局部设置中覆盖全局变量。

```
[DEFAULT]
ignoreip = 127.0.0.1/8
ignorecommand =
bantime  = 600                          # 封锁时间，负数表示永久封锁
findtime = 600                          # 在多长时间以内达到条件则开始执行封锁
maxretry = 3                            # 如 600 秒达到 3 次则执行
backend = auto
usedns = warn
destemail = zhenyanxl@126.com           # 警报邮件发给该地址
sendername = Fail2Ban
sender = "Fail2Ban 警报"
banaction = iptables-multiport
mta = sendmail                          # 邮件参数，可改为 mail 等
protocol = tcp
chain = INPUT
action_ = %(banaction)s[略...]          # 仅封锁
action_mw = %(banaction)s[略...]        # 封锁 + 发送邮件
```

```
            %(mta)s-whois[略...]
action_mwl = %(banaction)s[略...]           # 封锁 + 发送邮件（邮件包含日志信息）
             %(mta)s-whois-lines[略...]
action = %(action_)s                        # 默认动作
```

例如，可以修改 ssh 和 pure-ftp 的默认设置，让封锁之后发送警报邮件。

```
[ssh]
enabled  = true                              <-- 开启监控
port     = ssh
filter   = sshd
mta      = mail                              <-- 使用 mail 命令发送邮件
action   = %(action_mwl)s
logpath  = /var/log/auth.log                 <-- 日志位置
maxretry = 6

[pure-ftpd]
enabled  = true
port     = ftp,ftp-data,ftps,ftps-data
filter   = pure-ftpd
mta      = mail
action   = %(action_mwl)s
logpath  = /var/log/syslog
maxretry = 6
```

§7.2 防端口扫描

在服务器没有使用硬件防火墙的情况下，为了防止有人恶意获取服务器用途等信息，防端口扫描十分必要。PortSentry 是一款非常容易使用的防恶意扫描工具，它可设置如下几种抵挡方式：

- 路由表：通过设置路由表，将信息流导向虚假位置。
- TCP_wrappers：将对方的 IP 写入/etc/hosts.deny 文件。
- iptables：通过修改 iptables，将对方数据包过滤。
- syslog()：给出日志消息，返回警告信息。
- 自定义脚本。

其中，路由表和 iptables 是两种较好的方法。

在 Debian 8.x 中，安装 PortSentry 以后其配置文件在/etc/portsentry 目录中，主要的配置文件是 portsentry.conf，默认内容如下：

```
TCP_PORTS="1,11,15,79,111,119,143,540,635,1080,1524,2000,5742,6667,12345,12346,20034
UDP_PORTS="1,7,9,69,161,162,513,635,640,641,700,37444,34555,31335,32770,32771,32772,
ADVANCED_PORTS_TCP="1024"
ADVANCED_PORTS_UDP="1024"
ADVANCED_EXCLUDE_TCP="113,139"
ADVANCED_EXCLUDE_UDP="520,138,137,67"
IGNORE_FILE="/etc/portsentry/portsentry.ignore"
HISTORY_FILE="/var/lib/portsentry/portsentry.history"
BLOCKED_FILE="/var/lib/portsentry/portsentry.blocked"
RESOLVE_HOST = "0"
BLOCK_UDP="0"        <-- 默认不阻挡 UDP 扫描，将值改为 1
```

```
BLOCK_TCP="0"        <-- 默认不阻挡 TCP 扫描，将值改为 1
KILL_ROUTE="/sbin/route add -host $TARGET$ reject"       <-- 利用路由阻挡
#KILL_ROUTE="/sbin/route add $TARGET$ 333.444.555.666"   <-- 将数据流导向不存在的主机
KILL_HOSTS_DENY="ALL: $TARGET$ : DENY"   <-- 利用 TCP_wrappers 阻挡
SCAN_TRIGGER="0"
```

对 UDP 和 TCP，PortSentry 分别有三种检测模式：

- portsentry-tcp：TCP 基本端口绑定模式；
- portsentry-udp：UDP 基本端口绑定模式；
- portsentry-stcp：TCP 秘密扫描检测模式；
- portsentry-sudp：UDP 秘密扫描检测模式；
- portsentry-atcp：TCP 高级秘密扫描检测模式；
- portsentry-audp：UDP 高级秘密扫描检测模式。

推荐使用高级秘密扫描检测模式，配置文件是/etc/default/portsentry，修改其默认值然后重启 portsentry 服务即可。

```
TCP_MODE="atcp"
UDP_MODE="audp"
```