Linux课程大纲

第一节 Linux基础

一、Linux简介

Linux 是一个开源、免费的操作系统，是一个多用户、多任务、支持多线程和多 CPU 的操作系统，其稳定性、安全性已经得到业界的认可，目前很多企业级的项目都会部署到Linux系统上。

Linux主要应用于服务器领域。 Linux免费、稳定、高效等特点在这里得到了很好的体现，近些年来Linux服务器市场得到了飞速的提升，尤其在一些高端领域尤为广泛。

Linux在嵌入式领域的应用也非常广泛，它运行稳定、对网络的良好支持性、低成本，且可以根据需要进行软件裁剪，内核最小可以达到几百KB等特点，使其近些年来在嵌入式领域的应用得到非常大的提高 。

二、Linux的内核版本和发行版本

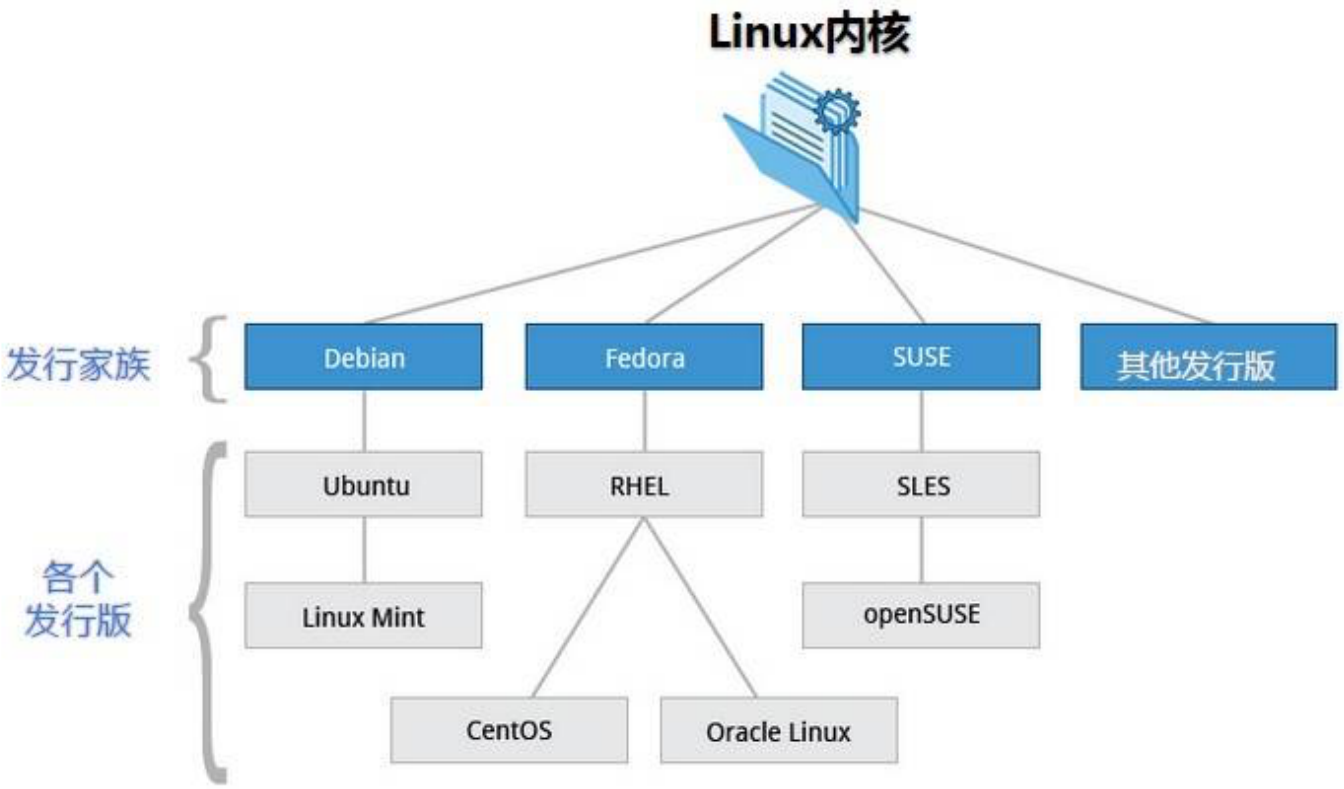
1.内核版本

内核是系统的心脏，是运行程序和管理像磁盘和打印机等硬件设备的核心程序，它提供了一个在裸设备与应用程序间的抽象层 。

> 内核版本号样式为
> **3.10.0-957.el7**
> 3：表示主版本号，有结构性变化时才变更
> 10：表示次版本号，新增功能时才发生变化；一般奇数表示测试版，偶数表示生产版
> 0：表示对次版本的修订次数或补丁包数
> 957：表示编译（或构建）的次数，每次编译可能对少量程序做优化或修改，但一般没有大的（可控的）功能变化
> e17：用来描述当前的版本特殊信息；其信息由编译时指定，具有较大的随意性

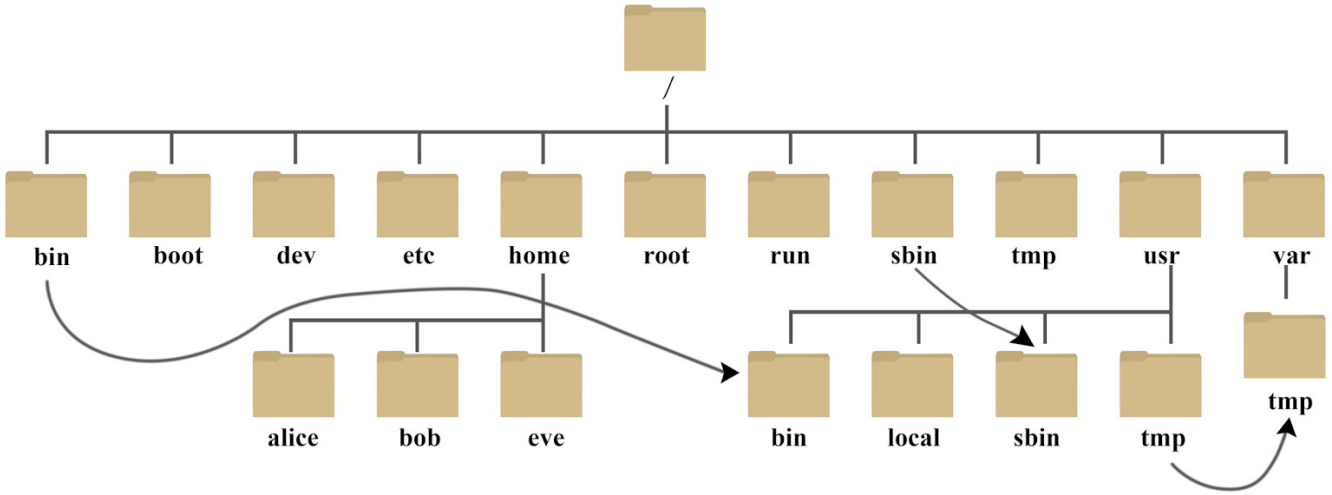
2.发行版本

就Linux的本质来说，它只是操作系统的核心，负责控制硬件、管理文件系统、程序进程等，并不给用户提供各种工具和应用软件 ，因此人们以Linux核心为中心，再集成搭配各种各样的系统管理软件或应用工具软件组成一套完整的操作系统，这样的组合便称为Linux发行版。



三、Linux目录结构

Linux系统中没有盘符的概念，所有的文件和目录都被组织成以一个根节点开始的倒置的树状结构， 文件系统的最顶层是由根目录开始的，系统使用 **/** 来表示根目录，呈现为一个树状结构。



FHS（文件系统层次结构标准）标准目录讲解

/ -----操作系统的根路径

/bin -----普通用户和管理员都可以执行的命令

/boot -----主引导目录，放置开机会使用到的文件，包括Linux核心文件以及开机菜单与开机所需配置文件等等

/dev -----设备文件目录，包含所有设备文件，如硬盘、USB设备等

/etc -----配置文件存放目录

/home -----普通用户家目录

/root -----管理员家目录

/lib -----共享库目录，包含用于启动系统和运行基本命令的共享库

/media -----可移除设备的挂载目录

/mnt -----临时设备建议的挂载目录

/proc -----存储的是当前内核运行状态的一系列特殊文件，这个目录是一个虚拟的目录，它是系统内存的映射，我们可以通过直接访问这个目录来获取系统信息

/sys -----这个目录跟/proc非常类似，也是一个虚拟的文件系统，主要也是记录核心与系统硬件信息较相关的信息

/srv -----srv可以视为“service”的缩写，用于存放本地服务所需的数据文件

/run -----临时文件系统，存储系统启动以来的信息。当系统重启时，这个目录下的文件应该被删掉或清除。

/tmp -----临时文件存放目录

/usr -----usr是Unix Software Resource的缩写， 也就是“Unix操作系统软件资源”所放置的目录，FHS建议所有软件开发者将他们的数据放置到这个目录下的子目录

```
/var  -----常变文件存放目录
```

四、绝对路径与相对路径

绝对路径：由根目录（/）开始写起的文件名或目录名称， 例如 /home/tom/

相对路径：相对于目前路径的文件名写法。 例如 ./home/tom或 ../../home/tom/ 等等。反正开头不是 / 就属于相对路径的写法

```
相对路径就是以  “你当前所在路径的相对位置”  来表示的。举例来说，如果目前在 /home 这个目录下， 如果想要进入 /var/log 这个目录时，可以采用下列两种写法：
```

- cd /var/log （绝对路径）
- cd ../var/log （相对路径）

  因为你在 /home 下面，所以要回到上一层 （../） 之后，才能继续往 /var 来移动，特别注意下列两个特殊的目录：

```
.  代表当前的目录，也可以使用 ./ 来表示
..  代表上一层目录，也可以 ../ 来代表
```

五、shell命令介绍

1.简介

通常来讲，计算机硬件是由运算器、控制器、存储器、输入/输出设备等共同组成的，而让各种硬件设备各司其职且又能协同运行的东西就是系统内核。Linux系统的内核负责完成对硬件资源的分配、调度等管理任务。由此可见，系统内核对计算机的正常运行来讲是太重要了，因此一般不建议直接去编辑内核中的参数，而是让用户通过基于系统调用接口开发出的程序或服务来管理计算机，以满足日常工作的需要。

Shell（也称为终端或者是壳）是一种命令行解释器，它是用户与操作系统内核之间的接口。充当的是人与内核之间的翻译官，用户把一些命令“告诉”终端，它就会调用相应的程序服务去完成某些工作。它提供了一种交互式的方式，让用户可以通过命令来操作计算机，执行各种任务和管理系统资源。

常见的 Shell 解释器包括 bash、zsh、csh、ksh等。bash 是许多 Linux 系统的默认 Shell，它提供了丰富的功能，包括命令历史、命令别名、管道、重定向等。zsh 是另一个流行的 Shell，它提供了比 bash 更多的功能，包括更强大的自动补全、更丰富的提示符配置等。csh 是 UNIX 系统上的一种 Shell，它的语法类似于C语言，因此得名。ksh 是 Korn Shell 的缩写，它提供了强大的编程能力和交互性。

2.命令格式

```
命令名称 [参数] [对象]
```

注意，命令名称、命令参数、命令对象之间请用空格键分隔。

命令对象一般是指要处理的文件、目录、用户等资源，而命令参数可以用长格式（完整的选项名称），也可以用短格式（单个字母的缩写），两者分别用 -- 与 - 作为前缀。

3.常用命令

ls

```
显示指定工作目录下的内容,如果不提供参数，ls将在当前目录上运行。
-a：列出指定目录下的所有文件和子目录（包括以“.”开头的隐含文件）。
-d：如果是目录，则显示目录的属性而不是目录下的内容。
-l：使用长格式显示文件或目录的详细属性信息。
-R：列出所有子目录下的文件
```

pwd

当前目录绝对路径

cd

```
改变当前工作目录,当不指定目标目录时，将被带到主目录。
cd      //进入主目录
cd ~    //进入主目录
cd /    //回到根目录
cd ..   //回到上一级目录
cd -    //返回到前一个工作目录
```

cat

```
查看文件的内容，将文件的内容输出到标准输出
格式：cat [参数] 文件名
-b：列出行号，仅针对非空白行号显示，空白行不标行号；
-n：打印出行号，连同空白行也会有行号，与-b的选项不同
```

tac

和cat命令相反，从最后一行开始显示
格式：tac [参数] 文件名

nl

```
计算文件中行号，将输出的文件内容自动的加上行号
格式：nl [选项] 文件名
-b ：指定行号指定的方式，主要有两种：
  -b a ：表示不论是否为空行，也同样列出行号（类似 cat -n）
  -b t ：如果有空行，空的那一行不要列出行号（默认值）
```

more

```
分页查看文件的内容
格式：more [选项] 文件名
空格键（space）:代表向下翻一页
回车键（Enter）:代表向下翻一行
b 键：往前查看一页，不过这动作只对文件有用，对管道无用
q 键：退出。
```

less

```
less命令 与 more命令 非常类似，但less命令 可以更加随意地浏览文件，而且 less 在查看之前不会加载整个文件
格式：less [选项] 文件名
空格键：向下翻动一页
Pagedown:向下翻动一页
Pageup:向上翻动一页
/字符串：向下搜寻[字符串]功能
? 字符串：向上搜寻[字符串]功能
n:重复前一个搜寻（与/或? 有关）
N：反向重复前一个搜寻（与/或? 有关）
q：离开less这个程序
```

head

```
用于查看文件的开头部分的内容，默认显示 10 行的内容
格式：head [选项] 文件名
-n：指定显示的行数，如果为负数则表示不显示后面的多少行
```

tail

```
用于查看文件的结尾部分的内容，默认显示 10 行的内容
格式：tail [选项] 文件名
-n：指定显示的行数，如果数字为+5表示从第5行开始显示内容
-f：实时查看被添加到一个文件中的内容
```

grep

```
在指定文本文件中匹配字符串，输出匹配字符串所在行的全部内容。
格式：grep [选项] 要查找的内容 目标文件名
-i：忽略大小写
-n：在文件中查找指定内容并显示匹配行的行号
-v：对匹配的结果进行反选
```

locate

```
用于查找符合条件的文件
格式：locate [选项] 要查找的文件
-i：忽略大小写
-r：使用基本正则表达式进行匹配
updatedb：更新locate命令的数据库
```

find

格式：find 路径 选项 要查找的文件
- -name 通过文件名查找文件
  示例：find / -name ??? 查找文件名长度为三个字符的文件
  示例：find . -name "[a-z]*[4-9].log" 在当前目录查找文件名以小写字母开头，最后是4到9加上.log结束的文件
- -user 通过用户名查找文件
- -exec 该参数后面跟命令，能够进行额外的动作
  示例： find /etc/ -name shadow -exec ls -l {} \;

该示例中特殊的地方有 {} 以及 \; 还有 -exec 这个参数，这些东西的意义为：
- {} 代表的是『由 find 找到的内容』，find 的结果会被放置到 {} 位置中；
- -exec 一直到 \; 是关键词，代表 find 额外动作的开始 (-exec) 到结束 (\;) ，在这中间的就是 find 指令内的额外动作。 在本例中就是『 ls -l {} 』
- 因为『 ; 』在 bash 环境下是有特殊意义的，因此利用反斜杠来转义。

mkdir

创建一个新目录
格式：mkdir [选项] 要创建的目录
-p：创建目录结构中指定的每一个目录，如果目录不存在则创建目录，如果目录已存在也不会被覆盖

touch

创建一个新的空文本文件
格式：touch [选项] 要创建的文件名
- 例：touch 1.txt

echo

创建文件并向文件中写入数据
- cat > 文件名：建立一文件，然后把接下来的键盘输入写入文件，直到按Ctrl+D为止
- echo "国家网络空间安全人才培养基地" > 123.txt //输出国家网络空间安全人才培养基地到123.txt文件
- echo "国家网络空间安全人才培养基地" >> 123.txt //追加输出国家网络空间安全人才培养基地到123.txt文件

cp

用于复制文件或目录，目录不能直接复制，需要加上-r参数
格式：cp [选项] 源文件 目标文件
- -b：若需覆盖文件，则覆盖前先行备份
- -r：复制目录，实现将源目录下的文件和子目录一起复制到目标目录中
- -f：如果目标文件或目录已经存在，则将其覆盖，并不作提示
- -p：连同文件的属性（权限、用户、时间）一起复制过去

mv

用于移动或重命名目录与文件
- -b：若需覆盖文件，则覆盖前先行备份
- -f：如果目标文件已经存在，不会询问而直接覆盖；

rm

用于删除一个文件或者目录
- `-f`：强制删除文件
- `-r`：递归删除目录

六、vim编辑器

1.介绍

vi是 Linux操作系统中最通用的文本编辑器,VIM 编辑器是从 vi 发展出来的一个性能更强大的文本编辑器。可以主动的以字体颜色辨别语法的正确性，方便程序设计，vim 与 vi 编辑器完全兼容。

vim 编辑器有 3 种基本工作模式，分别是命令模式、文本输入模式和末行模式

- 命令模式
  该模式是进入 vim 编辑器后的默认模式。在这个模式中可以使用上下左右按键来移动光标，可以使用删除字符或删除整行来处理文件内容，也可使用复制、粘贴来处理文件数据。
- 编辑模式
  在命令行模式中按下i、a或o即可进入编辑模式，在该模式下，用户输入的字符都会被保存起来，并将其显示在屏幕上。在文本输入过程中，若想回到命令模式下，按下Esc键即可。
- 末行模式
  在命令模式下，用户按:键即可进入末行模式下，此时会在显示窗口的最后一行（通常也是屏幕的最后一行）显示一个:作为末行模式的说明符，等待用户输入命令。

工作模式之间的切换方法：

- 如果要从命令模式转换到编辑模式，可以输入命令a、i或者o
- 如果需要从编辑模式返回命令模式，按下 Esc 键即可
- 在命令模式下输入 : 即可切换到末行模式，然后输入命令。

2.常用快捷命令

```
编辑模式：
i 当前光标前                    I光标所在行最前
a 当前光标后                    A光标所在行最后
o 当前光标的下一行              O当前光标的上一行
Esc 退出编辑模式

复制粘贴：
  yy            复制光标所在行
  2yy           复制光标所在行开始的2行
  p             粘贴
  dd            删除光标所在行（实际是剪切）
  3dd           删除光标所在行开始的3行（剪切）
  x             删除当前光标所在的字符
  X（大写）      删除当前光标前一个字符

光标移动：
  h 左移   j 下移 k 上移  l 右（四个箭头也可以上下左右移动）
```

```
  ctrl+f  往下翻页
  ctrl+b  向上翻页
  G       定位到最后一行（整个文档最后一行）
  5G      定位到第5行
  gg      定位到第一行（整个文档第一行）
  w       跳到下一个单词
  b       跳到上一个单词

其他命令:
  zz     保存并退出
  u             撤销
  ctrl+r        反撤销（撤销u操作）
  /             搜索，先输入/,再输入搜索内容+回车。n、N表示上（下）一个搜索结果

末行模式:
按 : 进入命令模式
  w    保存
  q    退出
  wq 或 x    保存退出
  q! 不保存强制退出
  wq! 强制保存并退出，管理员才有权限
  :set nu 显示行号
  :set nonu 关闭行号
```

七、文件打包与压缩

打包：将一大堆文件或目录什么的变成一个总的文件

压缩：将一个大的文件通过一些压缩算法变成一个小文件

为什么要区分这两个概念呢？其实这源于Linux中的很多压缩程序只能针对一个文件进行压缩，这样当你想要压缩一大堆文件时，你就得先借助另外的工具将这一大堆文件先打成一个包，然后再用压缩程序进行压缩。

tar

```
常用参数:
  -c: 建立压缩档案
  -x: 解包
  -t: 查看内容
  -f: 指定文件
  -v；显示执行过程
  -z: 使用gzip压缩或者解压
```

举例:

```
tar cf file.tar files                创建包含files的tar文件
tar xf files.tar                     从files.tar中提取文件
tar czf file.tar.gz files            使用Gzip压缩创建tar文件
tar xzf file.tar.gz                  使用Gzip提取tar文件
```

gzip

```
gzip file                                   压缩file并重命名为file.gz(原文件会清除)
gzip -d file.gz                             将file.gz解压缩为file(原文件会清除)
```

zip

```
zip -r xxx.zip  ./filename                  压缩filename的内容为xxx.zip文件
unzip filename.zip                          解压zip文件到当前目录
```

常见压缩文件扩展名
*.zip zip 程序压缩的文件；
*.gz gzip 程序压缩的文件；
*.tar tar 程序打包的数据，并没有压缩过；
*.tar.gz tar 程序打包的文件，其中并且经过 gzip 的压缩

第二节 Linux用户与权限管理

一、Linux用户介绍

Linux用户类型分为三类：超级管理员用户、系统用户和普通用户

- 超级管理员用户（root用户）：UID为 0，具有一切权限，可以操作系统中的所有资源。Root用户可以进行基础的文件操作及特殊的系统管理，可以修改系统中的任何文件。
- 系统用户：UID 为 1～999： Linux 系统为了避免因某个服务程序出现漏洞而被黑客提权至整台服务器，默认服务程序会有独立的系统用户负责运行，进而有效控制被破坏围。
- 普通用户： UID 从 1000 开始：是由管理员创建的用于日常工作的用户，能够使用Linux的大部分资源，一些特定的权限受到控制。用户只对自己的目录有写权限，读写权限受一定的限制，有效保证了系统安全性。

查看uid命令：

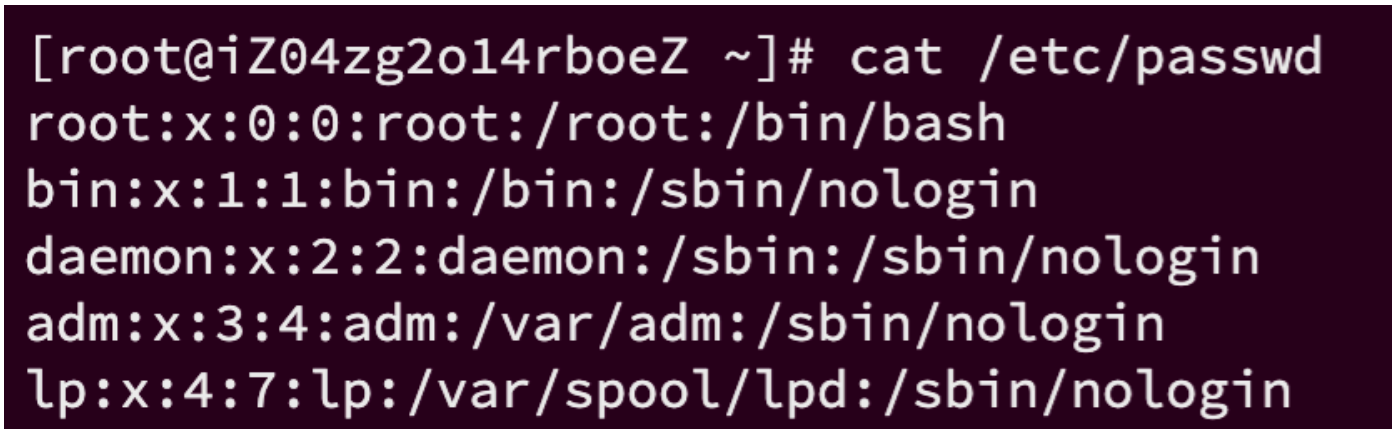
```
id
id -u username
```

二、用户和组相关文件

1./etc/passwd

/etc/passwd 文件，是用户配置文件，存储了系统中所有用户的基本信息，并且所有用户都可以读取此文件内容。
文件内数据格式如下

```
用户名：密码：UID（用户ID）：GID（组ID）：描述性信息：主目录：默认Shell
密码字段的"x" 表示此用户设有密码，但不是真正的密码，真正的密码保存在 /etc/shadow 文件中
```



2./etc/shadow

/etc/shadow 文件，用于存储 Linux 系统中用户的密码信息，又称为“影子文件”。由于/etc/passwd 文件允许所有用户读取，易导致用户密码泄露，因此 Linux 系统将用户的密码信息从 /etc/passwd 文件中分离出来，并单独放到了此文件中。/etc/shadow 文件只有 root 用户拥有读权限，其他用户没有任何权限，这样就保证了用户密码的安全性。文件内数据格式如下

> 用户名：加密密码：最近修改时间：最小修改时间间隔：密码有效期：密码变更前的警告天数：密码过期后的宽限时间：账号失效时间：保留字段

```
[root@iZ04zg2o14rboeZ ~]# cat /etc/shadow
root:$6$AZ3bHRkJ$fZBBQyWsbFzoUAJQvpEQussutZs.HeorSut2r7pXPAMWc/KAXkvwBcCcMcDUyemR9synVRs.0Q5Ke1smpkhPC/:19773:0:99999:7:::
bin:*:17834:0:99999:7:::
daemon:*:17834:0:99999:7:::
adm:*:17834:0:99999:7:::
lp:*:17834:0:99999:7:::
sync:*:17834:0:99999:7:::
shutdown:*:17834:0:99999:7:::
halt:*:17834:0:99999:7:::
```

> 其中密码字段分为三部分，每部分用 `$` 分隔
> `$6$`：这部分表示了密码散列算法的类型，6表示该密码使用了 SHA-512 算法，5：表示采用 SHA-256 加密，1：表示采用 MD5 加密
> `AZ3bHRkJ$fZBBQyWsbFzoUAJQvpEQussutZs`：这部分是“盐值”（Salt），用于增加密码破解的难度。这个盐值将与用户的明文密码一起作为 SHA-512 散列函数的输入，使得即使两个用户使用了相同的明文密码，他们的散列密码也会不同。
> `HeorSut2r7pXPAMWc/KAXkvwBcCcMcDUyemR9synVRs`：这部分是 SHA-512 加盐哈希算法处理过的密码散列值

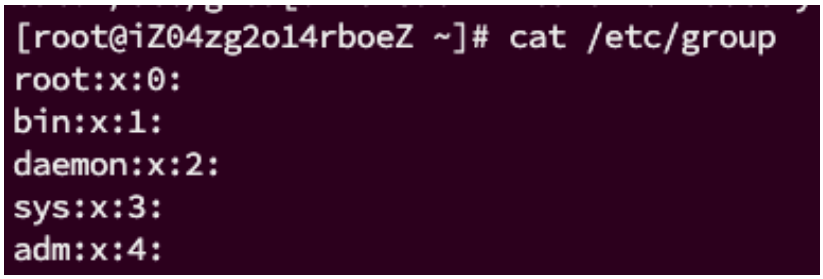
使用join破解密码：

1、将 /etc/shadow 文件内容导出到 shadow.tx 文件中
2、执行命令 john --wordlist=top500.txt shadow.txt 使用 top500.txt 字典对 shadow.txt 文件内容进行破解
3、执行命令 john --show shadow.txt 查看破解结果

3./etc/group

/etc/group文件，存放用户组的信息，此文件的格式也类似于/etc/passwd文件，由冒号(:)隔开若干个字段，这些字段有：

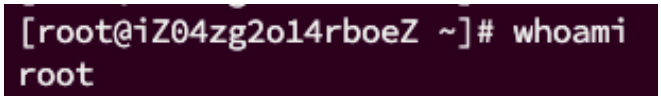
> 组名:口令:组标识号:组内用户列表



三、用户和组管理命令

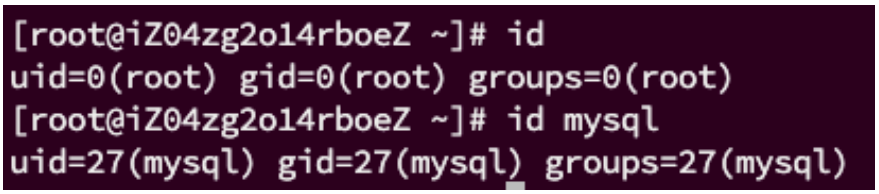
1.whoami

查看登录的用户



2.id

[用户名] 显示用户的ID，以及所属群组的ID



3.sudo

以管理员身份执行命令，普通用户无法使用sudo命令，需要将普通用户添加到/etc/sudoers文件中，才可以使用。

使用时可使用 -u 参数指定以特定用户身份执行命令。

因为在Linux系统中的/etc/sudoers文件第107行，存在这么一行内容：

```
106 ## Allows people in group wheel to run all commands
107 %wheel  ALL=(ALL)       ALL
```

上面的注释信息提示：允许用户在这个组里去执行任意命令。

- %wheel: 这是一个用户组名，前面的%符号指示该名称是一个用户组名而不是用户名。
- ALL=: 这是host list部分，表示此规则适用于所有主机。
- (ALL): 表示wheel用户组中的用户可以以任何用户的身份执行命令。
- ALL: 表示wheel用户组中的用户可以执行任何命令。

4.su

切换用户

```
格式：su [选项] 用户名
-c：向Shell传递单个命令
su - 用户名   #改变当前用户，比如用了su - root之后，下面所有的命令就可以不用打sudo了，因为当前用户已经是管理员root了
su - 用户名 -c "命令" #在指定用户下执行某个命令
注意：从 root用户切换到普通用户时不需要输入密码，而从普通用户切换成 root 用户需要进行验证
```

5.useradd

创建新的用户

```
格式：useradd [选项] 用户名
• -u：指定该用户的默认 UID
• -g：指定一个初始的用户基本组，useradd -g 用户组 新建的用户名
```

6.passwd

修改用户密码、过期时间、认证信息等

```
格式：passwd [选项] [用户名]
• -l：锁定用户，禁止其登录
• -u：解除锁定，允许用户登录
• -e：强制用户在下次登录时修改密码
```

7.userdel

删除用户

```
userdel [选项] 用户名
-r：同时删除用户及用户家目录
```

8.usermod

修改用户的属性

```
格式：usermod [选项] 用户名
• -g：修改用户的基本组，usermod -g 新基本组 用户名
• -G：修改用户的附加组，usermod -G 新附加组 用户名
• -a：将用户追加至上边 -G 中提到的附加组中，并不从其它组中删除此用户
• -u：修改用户的 UID
例：usermod -a -G root xiyangyang #把xiyangyang用户加入到root组
```

9.chage

修改帐号和密码的有效期限

格式：chage [选项] 用户名
- -M：密码保持有效的最大天数
- -E：修改密码失效日期
- -W：用户密码到期前多少天开始提醒

10.groupadd

创建一个组

格式：groupadd [选项] 组名
- -g：指定该组的默认 GID

例如：创建组ID为1111的用户组test，可以执行：groupadd –g 1111 test

11.groupmod

修改用户组的属性

格式：groupmod [选项] 组名
- -g：为用户组指定新的GID
- -n：为用户组改名

例如：要将系统中已经存在的组test组名修改为test1、组ID为1112，可执行命令：groupmod –g 1112 –n test1 test

12.gpasswd

将用户加入到某个组或从某个组移除

格式：gpasswd [选项] 用户名 组名
- -a：将用户加入到组中
- -d：将用户从组中移除

例：gpasswd -a xiyangyang root #正在将用户“xiyangyang”加入到“root”组中

13.groupdel

删除用户组

格式：groupdel 用户组
例如要将系统中已经存在的组test1删除，可执行命令：groupdel test1

14.groups

查看某个用户属于哪些组

格式：groups 用户名

四、Linux权限管理

在Linux服务器上有严格的权限等级，如果权限过高导致误操作会增加服务器的风险。

```
每个文件的权限针对三类对象进行定义
  它的所有者（由u表示）
  它的所有组（由g代表），代表该组的所有成员
  其他（由o表示，除所有者和所有组的用户）
  所有用户（用a表示）
每个文件针对每类访问者定义了三种主要权限
  读（由r表示）
  写（或修改，用w表示）
  执行（由x表示）
  rwx r-x r-x
   7   5   5

  rwx  r--  r--
  421  400  400
   7    4    4
```

1.权限对于目录和文件的意义

代表字符	权限值	权限	对文件的含义	对目录的含义
r	4	读	可以查看文件内容	可以列出目录中的内容
w	2	写	可以修改文件内容	可以在目录中创建、删除文件
x	1	执行	可以执行文件	可以进入目录

以下三个命令控制与文件关联的权限：

(1)chmod

更改文件的权限

```
格式：chmod [选项] 要分配的权限 文件或目录名
  -R：改变目录及目录下的文件和目录的访问权限
示例：chmod o=w test 表示将其他人对test这个文件的权限设置为只可写入

在权限修改时我们也可以采用数字来代替 rwx
r ------------4
w ------------2
x ------------1
例如：chmod 765 filename 表示授予所有者 rwx 权限，授予所属组 rw 权限，授予其他人 rx 权限
所有者 = r+w+x = 4+2+1 = 7
所属组 = r+w- = 4+2 = 6
其他人 = r+x = 4+1 = 5
```

(2)chown

更改文件和目录的所有者

```
格式：chown 选项 用户名 文件或目录名
  -R：改变目录及目录下的文件和目录的所有者
示例：chown -R hyf /tmp/test 将/tmp/test目录及其子目录的所有者改为hyf用户
```

(3)chgrp

更改文件和目录的所属组

```
格式：chgrp 选项 组名 文件或目录名
  -R：改变目录及目录下的文件和目录的所属组
示例：chgrp -R hyf /tmp/test 将/tmp/test目录及其子目录的所属组改为hyf用户
```

2.特殊权限位

特殊权限位包含粘滞位（SBIT）、SGID、SUID；

粘滞位（stickybit）针对目录赋权，为目录添加粘滞位后该目录中创建的文件和目录只有创建者和超级管理员可以删除（数字权限1）

```
粘滞位赋予方法：chmod o+t 目录名称
```

SUID针对可执行文件赋权， 作用在普通文件中没有任何意义 。在默认情况下，用户发起一个进程，该进程的属主是发起者，而并非是文件的属主，但是如果给二进制程序文件添加了SUID权限后，用户发起一个进程，该进程的属主为程序文件所属的属主，而并非是发起者。 (数字权限4)

```
suid赋予方法：chmod u+s 可执行文件名称
```

sgid针对目录赋权时，为目录添加sgid后在该目录中建立的文件属组继承父目录的属组 (数字权限2)

sgid针对可执行文件赋权时，用户发起一个进程，该进程的属主为程序文件所属的属组，而并非是发起者。(数字权限2)

```
sgid赋予方法：chmod g+s 目录名称
```

3.特殊属性

(1)chattr

改变属性

```
格式：chattr 选项 文件或目录
• i：不能更改
• a：不能删除和修改，只能追加
```

(2)lsattr

查看属性

```
格式：lsattr 文件或目录
```

4.umask

umask值用于设置用户在创建文件时的默认权限，当我们在系统中创建目录或文件时，目录或文件所具有的默认权限就是由umask值决定的。

对于root用户，系统默认的umask值是0022；对于普通用户，系统默认的umask值是0002。执行umask命令可以查看当前用户的umask值。

默认情况下，对于目录，用户所能拥有的最大权限是777；对于文件，用户所能拥有的最大权限是目录的最大权限去掉执行权限，即666。因为x执行权限对于目录是必须的，没有执行权限就无法进入目录，而对于文件则不必默认赋予x执行权限。

```
计算方式
对于目录：777 – umask = 创建的目录的默认权限
对于文件：666 – umask = 创建的文件的默认权限

7 7 7
0 0 2
7 7 5

6 6 6
0 0 2
6 6 4

修改umask方法
编辑/etc/profile和/etc/bashrc的内容，然后执行 source /etc/profile /etc/bashrc 重新读取bash配置
```

5.权限细化（ACL）

ACL主要的目的是在提供传统的 owner,group,others 的 read,write,execute 权限之外的细部权限划分。ACL 可以针对单一使用者，单一文件或目录来进行 r,w,x 的权限规范，对于需要特殊权限的设定非常有帮助。

ACL 主要可以针对下列几个方面来控制权限：

- 使用者 (user)：可以针对使用者来设定权限；
- 群组 (group)：针对群组为对象来设定其权限；
- 默认属性 (mask)：还可以针对在该目录下在建立新文件/目录时，规范新数据的默认权限；

ACL的设定方法：

(1)setfacl

设定某个目录/文件的 ACL

```
格式：setfacl [-bkRd] [{-m|-x} acl 参数] 目标文件名
  选项与参数：
  -m ：设定后续的 acl 参数给文件使用，不可与 -x 合用；
  -x ：删除后续的 acl 参数，不可与 -m 合用；
  -d ：设定『预设 acl 参数』，只对目录有效，在该目录新建的数据会引用此默认值
  -b ：移除『所有的』 ACL 设定参数；
  -k ：移除『预设的』 ACL 参数
  -R ：递归设定 acl ，即包括子目录都会被设定；
注意：设定ACL权限的用户和组必须存在，否则执行命令失败
```

例如：针对xiyangyang用户设置 ACL 权限 [u:账号:权限]；如果是组的话格式为[g:组名:权限]

```
# touch acl_test
# ll acl_test
  -rw-r--r--. 1 root root 0 8月  15 23:02 acl_test
# setfacl -m u:xiyangyang:rx acl_test
# ll acl_test
  -rw-r-xr--+ 1 root root 0 8月  15 23:02 acl_test
```

权限部分多了个 +

```
# 设定值中的 u 后面未指定用户，代表设定该文件拥有者
# setfacl -m u::rwx acl_test
# ll acl_test
  -rwxr-xr--+ 1 root root 0 8月  15 23:02 acl_test
```

上面显示 root 的权限成为 rwx 了！

注：如果一个文件设定了 ACL 参数后，他的权限部分就会多出一个 + 号，但是此时你看到的权限与实际权限可能就会有点误差，可以通过 getfacl 来查看。

(2)getfacl

查看某个目录/文件的 ACL

```
格式：getfacl filename
# getfacl acl_test
# file: acl_test     <==文件名
# owner: root        <==文件所有者
# group: root        <==文件所属组
user::rwx            <==文件所有者的权限
user:xiyangyang:r-x  <==xiyangyang用户的权限
group::r--           <==文件所属组的权限
mask::r-x            <==文件预设的有效权限
other::r--           <==其他用户的权限
```

有效权限的设定，用户或群组所设定的权限必须要存在于 mask 的权限设定范围内才会生效，此即有效权限 [m:权限]

例如：xiyangyang用户的权限与 mask 的集合仅有 r 的存在，所以 xiyangyang 用户仅具有 r 的权限，并不存在 w 权限，这就是 mask 的功能。我们可以通过 mask 来规定允许的最大权限，这样就可以避免不小心开放某些权限给其他用户或组了。

```
# touch acl.txt
# setfacl -m u:xiyangyang:rw acl.txt
# setfacl -m m:r acl.txt
# getfacl acl.txt
# file: acl.txt
# owner: root
# group: root
user::rw-
user:xiyangyang:rw-    #effective:r--
group::r--
mask::r--
other::r--
```

使用默认权限设定目录中新建文件的 ACL 权限继承 [d:[u|g]:[user|group]:权限]
例如：创建 test 目录，为该目录设置默认 ACL 后在该目录下新建的目录与文件均会继承该目录的ACL

```
# mkdir test
# setfacl -m d:u:xiyangyang:rx test
# getfacl test
# file: test
# owner: root
# group: root
user::rwx
group::r-x
other::r-x
default:user::rwx
default:user:xiyangyang:r-x
default:group::r-x
default:mask::r-x
default:other::r-x

# cd test
# touch 111
# mkdir 222
# ll
总用量 0
-rw-r--r--+ 1 root root 0 8月  16 00:16 111
drwxr-xr-x+ 2 root root 6 8月  16 00:16 222
```

(3)取消 ACL

```
setfacl -b filename #取消所有的ACL
setfacl -x u:xiyangyang test #取消test目录的xiyangyang用户的
setfacl -k test #取消默认ACL
```

第三节 Linux日志管理

一、日志简介

日志是记录系统运行过程中各种重要信息的文件，在系统运行过程中由各进程创建并记录。

日志的作用是记录系统的运行过程及异常信息，这些日志可以帮助我们对系统进行审核以及对一些故障进行排查，一般这些日志存储在/var/log目录中。常见日志文件及记录信息如下：

- /var/log/messages
  - 系统日志信息，这个文件相当的重要，几乎系统发生的错误讯息 (或者是重要的信息) 都会记录在这个文件中；如果系统发生莫名的错误时，这个文件是一定要查阅的日志文件之一。
- /var/log/secure
  - 用户认证相关的安全事件信息，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) 都会被记录在此文件中。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/maillog
  - 与邮件相关的日志文件
- /var/log/cron
  - 与定时任务相关的日志
- /var/log/boot.log
  - 与系统启动有关的日志，开机的时候系统核心会去侦测与启动硬件，接下来开始各种核心支持的功能启动等。这些流程都会记录在/var/log/boot.log 里面！不过这个文件只会存在这次开机启动的信息，前次开机的信息并不会被保留下来！
- /var/log/wtmp，/var/log/btmp
  - 这两个文件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 (btmp)，该文件无法通过cat等命令查看，可通过last命令查看。
- /var/log/lastlog
  - 可以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。该文件无法通过cat等命令查看，可通过lastlog命令查看。

(1)last

查看正确登录系统的用户

```
[root@iZ04zg2o14rboeZ kevin]# last
root     pts/1        101.39.213.146   Tue Jun 25 13:59   still logged in
root     pts/2        101.39.213.146   Mon Jun 24 08:55 - 09:29  (00:34)
root     pts/1        101.39.213.146   Fri Jun 21 15:13 - 11:03 (2+19:49)
root     pts/1        101.39.213.146   Tue Jun 18 11:10 - 13:44  (02:34)
root     pts/1        101.39.213.146   Tue Jun 18 11:06 - 11:08  (00:01)
```

- 第一列：用户名
- 第二列：终端位置，pts/0 (伪终端) 意味着从 SSH 或 telnet 的远程连接的用，.tty (teletypewriter) 意味着直接连接到计算机或者本地连接的用户

- 第三列：登录的 IP 或终端名，用户通过本地终端连接则显示空，如果是重启活动，会显示内核版本
- 第四列：登录开始时间
- 第五列：结束时间
- 第六列：持续时间，still logged in 表示仍然在线

(2)lastb

查看登录失败的用户

每一列内容同last命令一样

```
[root@iZ04zg2o14rboeZ kevin]# lastb
root     ssh:notty    60.246.138.17    Tue Jun 25 16:44 - 16:44  (00:00)
root     ssh:notty    60.246.138.17    Tue Jun 25 16:44 - 16:44  (00:00)
root     ssh:notty    60.246.138.17    Tue Jun 25 16:44 - 16:44  (00:00)
root     ssh:notty    60.246.138.17    Tue Jun 25 16:44 - 16:44  (00:00)
root     ssh:notty    60.246.138.17    Tue Jun 25 16:44 - 16:44  (00:00)
```

二、rsyslog服务

rsyslog是一个开源工具，它被广泛应用于Linux系统，它除了可以将收集的日志信息保存在本地外还可以通过TCP、UDP等协议将日志消息转发到其他主机。

rsyslog的主配置文件为 /etc/rsyslog.conf，大概分为三个部分：MODULES、GLOBAL DIRECTIVES、RULES。

```
#MODULES
这个部分是针对接收配置的，主要是指定接收日志的协议和端口。若要配置日志服务器，则需要将相应的配置项去掉注释。
#GLOBAL DIRECTIVES
这个部分主要用来配置模板，模板的作用是指定你希望在日志文件中保存的日志格式。
默认配置为：
# Use default timestamp format
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
```

```
指定日志采集规则
• 格式：日志类型.日志级别    执行动作
    cron.*                                                  /var/log/cron
```

1.日志类型

常见日志类型分为：

- auth　　#主要与认证/授权有关的机制，例如 login, ssh, su
- authpriv　#与 auth 类似，但记录较多账号私人的信息
- cron　　#定时任务相关
- kern　#内核产生的日志
- mail　#邮件收发信息
- lpr　　#打印产生的日志
- user　#用户程序产生的日志信息

2.日志级别

日志级别用来指出消息的优先等级，即消息的重要程度。其优先级别如下：

（数字等级越小，优先级越高，消息越重要，但记录的信息越少）

- 0 emerg（紧急）：会导致系统不可用的情况，一般是硬件出现问题
- 1 alert（警告）：必须马上采取措施解决的问题。
- 2 crit （严重）：严重的错误。
- 3 err （错误）：运行出现错误。
- 4 warning（提醒）：可能影响系统功能，需要提醒用户的重要事件。
- 5 notice（注意）：不会影响正常功能，但是需要注意的事件。
- 6 info（信息）：一般信息。
- 7 debug（调试）：程序或系统调试信息等。

除此之外，"日志类型"与"日志级别"可以使用星号（*）代表所有，因此*.*就表示来自所有类型的所有级别的消息。

3.执行动作

"执行动作"字段则用来定义如何处理接收到的消息，可以指定如下几项内容：

- /PATH/FILENAME 将消息存储到指定的文件中，文件必须以斜线（/）开头的绝对路径命名；
- :omusrmsg:USERNAME 将消息发送给指定的已经登录的用户；
- @HOSTNAME 将消息转发到指定的日志服务器；一个@表示通过UDP发送，两个@表示通过TCP发送
- :omusrmsg:* 将消息发送给所有已经登录的用户。

4.预设规则

*.info；mail.none；authpriv.none；cron.none：由于 mail, authpriv, cron 等类别产生的讯息较多， 且已经写入以下的几个文件中，因此在 /var/log/messages 里面就不记录这些内容。除此之外的其他讯息都写入/var/log/messages 中。

- authpriv.*：认证方面的信息均写入 /var/log/secure 文件；

- mail.*：邮件方面的信息则均写入 /var/log/maillog 文件；
- cron.*：计划任务均写入 /var/log/cron 文件；
- *.emerg：当产生最严重的错误等级时，将该等级的讯息广播给所有在系统登录的账号，要这么做的原因是希望在线的用户能够赶紧通知系统管理员来处理错误。
- uucp,news.crit：uucp 是早期 Unix-like 系统进行数据传递的通讯协议，后来常用在新闻组的用途中。 news 则是新闻组。当新闻组方面的信息有严重错误时就写入 /var/log/spooler 文件中；
- local7.*：将本机开机时应该显示到屏幕的讯息写入到 /var/log/boot.log 文件中；

5.自定义日志采集格式

在日志客户机（发送端）上设置：

```
#新增规则，向日志中心服务器上发送日志
*.*    @[日志中心机真实IP地址]
```

在日志中心机（收集端）上设置：

```
$template 自定义格式名[example: MyFormatLog],"%timegenerated% %FROMHOST-IP% %syslogtag%
%msg%\n"

%timegenerated%    #显示日志时间
%FROMHOST-IP%      #显示主机IP
%syslogtag%        #日志记录目标
%msg%              #日志内容
\n                 #换行

#在指定的日志中采用自定义格式
日志类型.日志级别   执行动作;自定义格式名[example: MyFormatLog]

#修改系统默认日志采集格式为自定义格式，将默认配置注释
$ActionFileDefaultTemplate 自定义格式名[example: MyFormatLog]

#启用UDP(或TCP)服务端，将下面两行内容取消注释
$ModLoad imudp
$UDPServerRun 514

# 根据客户端的IP单独存放主机日志在不同目录，设置远程日志存放路径及文件名格式
$template Remote,"/var/log/syslog/%fromhost-ip%/%fromhost-ip%_%$YEAR%-%$MONTH%-%$DAY%.log"
# 排除本地主机IP日志记录，只记录远程主机日志
:fromhost-ip, !isequal, "127.0.0.1" ?Remote
# 注意此规则需要在其它规则之前，否则配置没有意义，远程主机的日志也会记录到Server的日志文件中
```

7.journalctl

日志查看工具

```
-n 3          #查看最近3条日志
-p err        #查看错误日志
-o verbose    #查看日志的详细参数
--since       #查看从什么时间开始的日志
--until       #查看到什么时间为止的日志
_PID=pid      #只输出指定 PID 号码的信息
_UID=uid      #只输出指定 UID 号码的信息
```

如何使用systemd-journald保存系统日志，默认systemd-journald是不保存系统日志到硬盘的。
那么关机后再次开机只能看到本次开机之后的日志，关机之前的日志时无法查看的，想要保存日志需要进行如下操作：

```
mkdir /var/log/journal
chgrp systemd-journal /var/log/journal
chmod g+s /var/log/journal
killall -1 systemd-journald
ls /var/log/journal
```

第四节 Linux系统管理

一、同步时间

CentOS可以使用NTP协议来同步时间服务器。

1.安装ntp服务

```
yum install ntp ntpdate -y
```

2.配置ntp服务器

将/etc/ntp.conf文件中原本的NTP服务器注释，并设置下方新的NTP服务器：

```
server ntp1.aliyun.com iburst
server ntp2.aliyun.com iburst
```

配置完毕后，要重启ntp服务

```
systemctl restart ntpd
```

3.设置ntp服务开机自启

```
systemctl enable ntpd
```

4.设置硬件时钟

使用当前系统时间设置硬件时钟

```
hwclock -w
```

二、特殊字符

1.数据重定向

标准输出（STDOUT，文件描述符为 1，使用 > 或 >>）：默认输出到屏幕。

错误输出（STDERR，文件描述符为 2，使用 2> 或 2>>）：默认输出到屏幕

```
标准输出重定向示例：
ll / #此时屏幕会显示出文件信息
ll / > ~/rootfile #将标准输出重定向到~/rootfile
cat ~/rootfile #~/rootfile文件中的内容是ll /命令的执行结果
```

- 1>：以覆盖的方法将『正确的数据』输出到指定的文件或装置上；
- 1>>：以累加的方法将『正确的数据』输出到指定的文件或装置上；
- 2>：以覆盖的方法将『错误的数据』输出到指定的文件或装置上；
- 2>>：以累加的方法将『错误的数据』输出到指定的文件或装置上；

/dev/null 垃圾桶黑洞文件与文件重定向特殊写法：
如果希望执行某个命令，但又不希望在屏幕上显示出输出的结果，那么可以将输出重定向到/dev/null。

如果要将正确与错误数据通通写入同一个文件去，这个时候就得要使用下面的特殊重定向写法：

```
案例：下列命令使用普通用户执行
find /home -name .bashrc > list 2>&1
find /home -name .bashrc &> list
```

2.多命令执行

多命令执行符	格式	作用
;	命令1 ； 命令2	多个命令顺序执行，命令之间没有任何逻辑关系
&&	命令1 && 命令2	逻辑与，当命令1正确执行，则命令2才会执行，当命令1执行不正确，则命令2不会执行
||	命令1 || 命令2	逻辑或，当命令1执行不正确，命令2才会执行，当命令执行正确，命令2不会执行

范例一：使用 ls 查阅目录 /tmp/abc 是否存在，若存在则用 touch 建立 /tmp/abc/hehe

```
[root@study ~]$ ls /tmp/abc && touch /tmp/abc/hehe
ls: cannot access /tmp/abc: No such file or directory
```

范例二：测试 /tmp/abc 是否存在，若不存在则予以建立，若存在就不作任何事情

```
[root@study ~]$ rm -r /tmp/abc #先删除此目录以方便测试
[root@study ~]$ ls /tmp/abc || mkdir /tmp/abc
ls: cannot access /tmp/abc: No such file or directory #提示不存在
[root@study ~]$ ll -d /tmp/abc
drwxrwxr-x. 2 root root 6 Jul 9 19:17 /tmp/abc #有结果了！说明有进行 mkdir
```

范例三：我不清楚 /tmp/abc 是否存在，但就是要建立 /tmp/abc/hehe 文件

```
[root@study ~]$ ls /tmp/abc || mkdir /tmp/abc && touch /tmp/abc/hehe
```

该范例总是会尝试建立 /tmp/abc/hehe ，不论 /tmp/abc 是否存在。由于 Linux 底下的指令都是由左往右执行的，所以范例三有几种结果我们来分析一下：

(1)若 /tmp/abc 不存在，则开始 mkdir /tmp/abc，由于 mkdir /tmp/abc 会成功进行，故会执行 touch /tmp/abc/hehe，最终 hehe 就被建立了；

(2)若 /tmp/abc 存在，所以 mkdir /tmp/abc 不会执行，此时继续向后运行，然后遇到 && 后就开始建立 /tmp/abc/hehe 了！最终 /tmp/abc/hehe 被建立起来。

三、网络和系统管理

1.ifconfig

显示或设置网络设备

```
[root@iZ04zg2o14rboeZ ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.59.35  netmask 255.255.240.0  broadcast 172.17.63.255
        ether 00:16:3e:34:6f:60  txqueuelen 1000  (Ethernet)
        RX packets 13459488  bytes 3550967142 (3.3 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11215442  bytes 2858119888 (2.6 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 142030  bytes 17500328 (16.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 142030  bytes 17500328 (16.6 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- 第一行：eth0表示网卡名字
  - UP：代表此网络接口为启用状态（down为关闭状态）
  - RUNNING：代表网卡设备已连接
  - MULTICAST：表示支持组播
  - MTU：为数据包最大传输单元

- 第二行：网卡的IP地址、子网掩码、广播地址
- 第三行：IP v6地址、掩码长度、作用域link表示仅该接口有效（有的系统可能没有ipv6地址）
- 第四行：表示为网卡的MAC地址 传输队列长度 接口类型
- 第五行：接受数据包个数、大小统计信息
- 第六行：异常接受包的个数、如丢包量、错误等
- 第七行：发送数据包个数、大小统计信息
- 第八行：发送包的个数、如丢包量、错误等

```
ifconfig 网卡名字 up       #启动指定网卡
ifconfig 网卡名字 down     #关闭指定网卡
ifconfig -a               #显示所有网卡接口的信息（包括未激活的网卡接口）
```

2.ping

检测网络连通性

格式：ping [选项] host
- -c：指定ping的次数
- -s：设置ping包长度
- -i：设置发送包间隔时间
- -f：快速的连续ping一台主机，ping的速度达到100次每秒

3.netstat

显示网络相关信息

格式：netstat [选项]
- -a：显示所有连接
- -t：仅显示tcp相关选项
- -u：仅显示udp相关选项
- -n：拒绝显示别名，能显示数字的全部转化成数字。
- -l：仅列出有在 Listen （监听）的端口
- -p：显示建立相关链接的程序名

```
[root@iZ04zg2o14rboeZ ~]# netstat -lntup
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      16891/nginx: master
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      985/pure-ftpd (SERV
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1428/sshd
```

4.wget

用于在终端下载网络文件

格式：wget [参数] 下载地址
- -b：后台下载模式
- -c：启用断点续传
- -P：下载到指定目录

5.ps

格式：ps [选项]
- -a：显示所有用户的进程
- -u：显示用户名和启动时间
- -x：显示没有控制终端地进程
- -e：显示没有控制终端地进程，相较于x选项更为简略
- -f：全格式显示

```
[root@iZ04zg2o14rboeZ ~]# ps -ef|head
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 Feb20 ?        00:09:49 /usr/lib/systemd/systemd --system --deserialize 17
root         2     0  0 Feb20 ?        00:00:00 [kthreadd]
root         3     2  0 Feb20 ?        00:00:04 [ksoftirqd/0]
root         5     2  0 Feb20 ?        00:00:00 [kworker/0:0H]
root         7     2  0 Feb20 ?        00:00:01 [migration/0]
root         8     2  0 Feb20 ?        00:00:00 [rcu_bh]
root         9     2  0 Feb20 ?        00:08:09 [rcu_sched]
root        10     2  0 Feb20 ?        00:00:00 [lru-add-drain]
root        11     2  0 Feb20 ?        00:00:48 [watchdog/0]
[root@iZ04zg2o14rboeZ ~]# ps -aux|head
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  51712  3708 ?        Ss   Feb20   9:49 /usr/lib/systemd/systemd --system --deserialize 17
root         2  0.0  0.0      0     0 ?        S    Feb20   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Feb20   0:04 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Feb20   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    Feb20   0:01 [migration/0]
root         8  0.0  0.0      0     0 ?        S    Feb20   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    Feb20   8:09 [rcu_sched]
root        10  0.0  0.0      0     0 ?        S<   Feb20   0:00 [lru-add-drain]
root        11  0.0  0.0      0     0 ?        S    Feb20   0:48 [watchdog/0]
```

显示结果说明：

- USER：进程所有者的用户名。
- PID：进程ID。
- %CPU：进程占用的CPU百分比。
- %MEM：进程占用的物理内存百分比。
- VSZ：进程的虚拟内存大小（以KB为单位）。
- RSS：进程占用的物理内存大小（以KB为单位）。
- TTY：进程所在的控制终端（如果没有，则显示?）。
- STAT：进程状态，其中 R表示（运行）：进程正在运行或在运行队列中等待，S表示（中断）：进程处于休眠中，当某个条件形成后或者接收到信号时，则脱离该状态，D表示（不可中断）：进程不响应系统异步信号，即便用kill命令也不能将其中断，Z表示（僵死）：进程已经终止，但进程描述符依然存在， 直到父进程调用wait4()系统函数后将进程释放，T（停止）：进程收到停止信号后停止运行。
- START：进程启动时间。
- TIME：进程消耗的累积CPU时间。
- COMMAND：启动进程的命令。

6.top

用于动态监视进程活动与系统负载等信息

格式：top [选项]

- -d ：后面可以接秒数，指定更新时间
- -p ：指定某些个 PID 来进行观察监测
- -E ：切换显示单位

  在 top 执行过程当中可以使用的按键指令：
  - P ：以 CPU 的使用资源排序显示；
  - M ：以 内存 的使用资源排序显示；
  - N ：以 PID 来排序
  - T ：由该 进程 使用的 CPU 时间累积 (TIME+) 排序

```
top - 17:11:53 up 127 days, 39 min,  1 user,  load average: 0.01, 0.02, 0.05
Tasks: 124 total,   1 running, 123 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.2 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  8009108 total,   163048 free,   815784 used,  7030276 buff/cache
KiB Swap:        0 total,        0 free,        0 used.  6861588 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
26055 mysql     20   0 2276672 333816   9044 S   0.3  4.2   6:18.67 mysqld
    1 root      20   0   51712   3708   2412 S   0.0  0.0   9:49.35 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.29 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:04.52 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    7 root      rt   0       0      0      0 S   0.0  0.0   0:01.30 migration/0
```

top命令执行结果的前5行为系统整体的统计信息，其所代表的含义如下：

- 第1行：系统时间、运行时间、登录终端数、系统负载（三个数值分别为1分钟、5分钟、15分钟内的平均值，数值越小意味着负载越低）。
- 第2行：进程总数、运行中的进程数、睡眠中的进程数、停止的进程数、僵死的进程数。
- 第3行：用户占用资源百分比、系统内核占用资源百分比、空闲的资源百分比等。
- 第4行：物理内存总量、内存空闲量、内存使用量、冲和缓存占用的内存。
- 第5行：虚拟内存总量、虚拟内存空闲量、虚拟内存使用量、可用内存。

top 下半部分的画面，则是每个 进程 使用的资源情况。比较需要注意的是：

- PID ： 每个进程的 ID
- USER：进程的所有者；
- PR ： 进程的执行顺序，越小越早被执行；
- NI ： 与优先级有关，也是越小越早被执行；
- VIRT：进程使用的虚拟内存总量
- RES：进程使用的物理内存大小
- SHR：进程共享内存的大小
- S：进程状态
- %CPU：CPU 的使用率；

- %MEM：内存的使用率；
- TIME+：进程消耗的累积CPU时间；
- COMMAND：启动进程的命令

7.kill

终止指定进程的运行，kill命令是通过向进程发送指定的信号来结束相应进程的。在默认情况下，采用kill命令编号为15的信号。该信号将终止所有不能捕获该信号的进程。对于那些可以捕获该信号的进程就要用编号为9的kill信号，强行“杀掉”该进程。

```
格式：kill [选项] 进程号
• kill -9 进程号     #强制杀掉进程
```

8.killall

终止特定的一类进程，killall 命令不再依靠 PID 来杀死单个进程，而是通过程序的进程名来杀死一类进程

```
格式：killall [选项]  进程名
• -i：交互式，询问是否要杀死某个进程；
• -I：忽略进程名的大小写
```

9.systemctl

管理系统中的服务

```
systemctl start 服务名        #启动服务
systemctl restart 服务名      #重启服务
systemctl stop 服务名         #停止服务
systemctl status 服务名       #查看服务
systemctl enable 服务名       #添加开机自启
systemctl disable 服务名      #取消开机自启
systemctl is-enabled 服务名   #查看服务是否开机启动
```

10.service

同systemctl，管理系统中的服务

11.free

系统内存的使用情况

```
• -h：以友好的方式显示内存使用情况
• -s：指定刷新的时间，持续观察内存使用状况
```

```
[root@iZ04zg2o14rboeZ ~]# free -h
              total        used        free      shared  buff/cache   available
Mem:           7.6G        796M        155M        860K        6.7G        6.5G
Swap:            0B          0B          0B
```

输出内容介绍

- Mem 行(第二行)是内存的使用情况。
- Swap 行(第三行)是交换空间的使用情况。
- total 列显示系统总的可用物理内存和交换空间大小。
- used 列显示已经被使用的物理内存和交换空间。
- free 列显示还有多少物理内存和交换空间可用使用。
- shared 列显示被共享使用的物理内存大小。
- buff/cache 列显示被 缓冲区 和 缓存区 使用的物理内存大小。
- available 列显示还可以被应用程序使用的物理内存大小。

12.reboot

重启系统

```
其他命令还有:
shutdown -h now: 立即关机
shutdown -h 10: 十分钟以后关机
shutdown -c: 取消定时关机
poweroff: 关闭系统
```

四、配置sshd服务

1.介绍

SSH（Secure Shell）是一种能够以安全的方式提供远程登录的协议，也是目前远程管理Linux系统的首选方式。在此之前，一般使用FTP或Telnet来进行远程登录。但是因为它们以明文的形式在网络中传输账户密码和数据信息，因此很不安全，很容易受到黑客发起的中间人攻击，这轻则篡改传输的数据信息，重则直接抓取服务器的账户密码。

sshd是基于SSH协议开发的一款远程管理服务程序，不仅使用起来方便快捷，而且能够提供两种安全验证的方法：

- 基于口令的验证——用账户和密码来验证登录；
- 基于密钥的验证——需要在本地生成密钥对，然后把密钥对中的公钥上传至服务器，并与服务器中的公钥进行比较；该方式相较来说更安全。

2.配置文件

sshd服务的配置信息保存在 /etc/ssh/sshd_config 文件中。我们一般会把保存着最主要配置信息的文件称为主配置文件，而配置文件中有许多以符号 # 开头的注释行，要想让这些配置参数生效，需要在修改参数后再去掉前面的符号 # 。

sshd_config文件中常见的重要参数：

- #Port 22 #默认的sshd服务端口，修改后可能无法启动ssh，执行 setenforce 0 临时关闭SELinux
- #ListenAddress 0.0.0.0 #表示侦听所有地址，如果主机不需要从公网ssh访问，可以把监听地址改为内网地址
- #PermitRootLogin yes #是否允许root登录

- #PasswordAuthentication yes #是否允许使用密码验证
- #PubkeyAuthentication yes #是否允许使用密钥验证
- #PermitEmptyPasswords no #是否允许空密码登录
- #MaxAuthTries 6 #最大密码尝试次数

3.远程连接

ssh命令连接远程主机：

```
ssh -p 目标主机ssh端口号 用户名@远程主机IP地址
#-p参数表示指定端口，如果目标主机没有自定义ssh使用的端口号可不使用该参数
```

4.免密登录

(1)生成密钥对

首先在服务器上制作密钥对。首先用密码登录到你打算使用密钥登录的账户，然后执行以下命令：

```
ssh-keygen
```

(2)安装公钥

在你需要登录的服务器上，将你的公钥内容复制到服务器登录用户家目录下的authorized_keys文件中

```
vi ~/.ssh/authorized_keys
chmod 600 authorized_keys
chmod 700 ~/.ssh
```

(3)开启密钥登录功能

编辑 /etc/ssh/sshd_config 文件，进行如下设置：

```
RSAAuthentication yes #启用RSA认证
PubkeyAuthentication yes  #开启公钥验证
```

如果是设置root用户的密钥登录功能，所以要查看root用户是否能通过ssh登陆

```
PermitRootLogin yes
```

重启ssh服务

```
systemctl restart sshd
```

如果想要通过单个私钥登录多台服务器，可以通过 ssh-copy-id IP地址 命令将公钥复制到其它主机。

五、远程传输命令

scp（secure copy）是一个基于SSH协议在网络之间进行安全传输的命令，其格式为：

- scp [参数] 本地文件 远程帐户@远程IP地址：远程目录
- -P(大写)：指定远程主机的sshd端口号
- -r：用于传输文件夹
- -v：显示详细的连接进度

scp不仅能够通过网络传送数据，而且所有的数据都将进行加密处理。例如，如果想把一些文件通过网络从一台主机传递到其他主机，这两台主机又恰巧是Linux系统，这时使用scp命令就可以轻松完成文件的传递了。还可以使用scp命令把远程主机上的文件下载到本地主机，其命令格式为：

- scp [参数] 远程用户@远程IP地址：远程文件 本地目录 #从远处复制到本地
- scp [参数] 本地文件 远程用户@远程IP地址:远程目录 #从本地复制到远处

六、软件安装

rpm命令是RPM软件包的管理工具。rpm原本是Red Hat Linux发行版专门用来管理Linux各项套件的程序，由于它遵循GPL规则且功能强大方便，因而广受欢迎。逐渐受到其他发行版的采用 。

1.rpm的用途

- 安装、删除、升级和管理软件
- 通过RPM包管理能知道软件包包含哪些文件，也能知道系统中的某个文件属于哪个软件包；
- 可以在查询系统中的软件包是否安装以及其版本；

2.rpm优点和缺点

- 优点
  由于已经编译完成并且打包，所以安装很方便
  由于套件信息已经记录在Linux主机中，方便查询、升级与卸载
- 缺点
  安装环境必须与打包时的环境一致
  需要满足软件的依赖属性需求
  卸载时需要特别小心，最底层的软件不可以先删除，否则可能造成整个系统出问题

3.rpm命令基本使用

```
rpm -qa                  #列出所有已经安装过的rpm软件包
rpm -qpi rpm软件包名      #查看该软件包的信息
rpm -qpl rpm软件包名      #查看该软件包安装后的路径
rpm -ivh rpm软件包名      #安装软件包
rpm -e 软件名             #卸载软件
```

4.yum安装软件

yum 是一个在 Fedora 和 RedHat 以及 SUSE 中的 Shell 前端软件包管理器。基于 RPM 包管理，能够从指定的服务器自动下载 RPM 包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软件包，无须繁琐地一次次下载、安装。yum 提供了查找、安装、删除某一个、一组甚至全部软件包的命令，而且命令简洁而又好记。

```
常用命令
yum check-update  # 列出所有可更新的软件清单命令
yum -y update   # 升级所有软件和系统内核
yum install 包名     # 安装指定的软件
yum update 包名    # 仅更新指定的软件命令
yum remove 包名    # 删除软件包
yum search 关键字    # 查找软件包
```

更换国内yum源

```
1. 切换到/etc/yum.repos.d/
2. 备份当前yum源：cp CentOS-Base.repo CentOS-Base-repo.bak
3. 安装wget：yum install -y wget
4. 使用wget下载阿里yum源repo文件：wget http://mirrors.aliyun.com/repo/Centos-7.repo
5. 清理默认缓存包：yum clean all
6. 把下载下来的阿里云repo文件设置成为默认源：mv Centos-7.repo CentOS-Base.repo
7. 生成缓存：yum makecache

网易yum源repo文件下载地址：http://mirrors.163.com/.help/CentOS7-Base-163.repo
```

七、firewalld防火墙

1.放行端口

```
firewall-cmd --add-port=端口/通讯协议 --permanent
```

- --add-port=端口/通讯协议   指定添加的端口号和协议
- --permanent   永久生效

2.使配置生效

```
firewall-cmd --reload
```

3.删除放行端口

```
firewall-cmd --remove-port=端口/通讯协议 --permanent
--remove-port=端口/通讯协议 #指定移除的端口号和协议
```

4.批量放行端口

```
firewall-cmd --add-port=开始端口-结束端口/通讯协议 --permanent
```

5.查看开放端口

```
firewall-cmd --list-ports
```

6.通过服务放行端口

```
firewall-cmd --add-service=httpd --permanent
```

7.查看开放的服务端口

```
firewall-cmd --list-services
```

8.rich rules

富规则，即更细致、更详细的防火墙规则策略

```
firewall-cmd --add-rich-rule="rule family="ipv4" source address="192.168.3.68" accept" --
permanent
#允许192.168.3.68主机访问

firewall-cmd --add-rich-rule="rule family="ipv4" source address="192.168.3.68" reject" --
permanent
#禁止192.168.3.68主机访问

firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.100" service
name="ssh" accept' --permanent
#允许192.168.0.100主机访问ssh服务

firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.100" service
name="http" reject' --permanent
#禁止192.168.0.100主机访问http服务

firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" service
name="http" reject' --permanent
#禁止192.168.0.0/24网段主机访问http服务

firewall-cmd --add-rich-rule='rule family='ipv4' source address='192.168.3.68' port
protocol='tcp' port='80' accept' --permanent
#允许192.168.3.68主机访问80端口

firewall-cmd --add-rich-rule='rule family='ipv4' source address='192.168.3.68' port
protocol='tcp' port='22' reject' --permanent
#禁止192.168.3.68主机访问22端口

firewall-cmd --add-rich-rule='rule protocol value=icmp drop' --permanent
#禁止ping

firewall-cmd --list-rich-rules
#查看已经创建的富规则
```

第五节 CentOS 7 基线检查

一、身份鉴别

1.检查空密码账户

- 执行命令
  - awk -F: '($2 == "") { print $1}' /etc/shadow
    查看是否有返回信息，没有返回信息说明系统当中没有空密码账户，如果有返回值，参照下方的加固步骤进行操作。
- 基线合规性判定
  - 没有返回信息即合规，否则不合格
- 加固步骤
  - 执行命令：cp - p /etc/shadow /etc/shadow_bak 对文件进行备份
    执行命令：passwd username 为用户设置密码
    或者执行命令：userdel username 删除空密码用户

2.检查密码有效期

- 执行命令
  - cat /etc/login.defs 查看 PASS_MAX_DAYS 参数设置是否为 60-180 之间
- 基线合规性判定
  - PASS_MAX_DAYS 参数设置为 60-180 之间即合规，否则不合规
- 加固步骤
  - 执行命令：cp /etc/login.defs /etc/login.defs_bak 对文件进行备份
    执行命令：vi /etc/login.defs 将PASS_MAX_DAYS 参数设置为60-180之间，如：PASS_MAX_DAYS 90
    执行命令：chage -M 90 username 修改之前已存在用户的密码到期时间

3.检查密码修改最小间隔时间

- 执行命令
  - cat /etc/login.defs 查看 PASS_MIN_DAYS 参数设置是否为 7-14 之间
- 基线合规性判定
  - PASS_MIN_DAYS 参数设置为 7-14 之间即合规，否则不合规
- 加固步骤
  - 执行命令：vi /etc/login.defs 将PASS_MIN_DAYS 参数设置为7-14之间，建议为7
    执行命令：chage -m 7 username 修改之前已存在用户的密码修改最小间隔时间

4.确保密码到期警告为7天或以上

- 执行命令
  - cat /etc/login.defs 查看 PASS_WARN_AGE 参数设置是否为 7-14 之间
- 基线合规性判定
  - PASS_WARN_AGE 参数设置为 7-14 之间即合规，否则不合规
- 加固步骤

    - 执行命令：vi /etc/login.defs 将PASS_WARN_AGE 参数设置为7-14之间，建议为7
      执行命令：chage -W 7 username 修改之前已存在用户的密码修改最小间隔时间

5.确保root是唯一的UID为0的账户

- 执行命令
    - awk -F: '($3 == "0") {print $1}' /etc/passwd 查看返回值是否有 root 之外的用户
- 基线合规性判定
    - 返回值只有 root 即合规，否则不合规
- 加固步骤
    - 执行命令：userdel -rf username 删除root之外的UID为0的用户

6.检查密码复杂度要求

- 执行命令
    - grep -E '^minlen|^minclass' /etc/security/pwquality.conf
      查看返回值当中是否有 minlen 和minclass，并且minlen设置是否为8-32之间，minclass设置是否为3或4
- 基线合规性判定
    - 返回值当中存在上述两项且minlen设置为8-32之间，minclass设置为3或4即合规，否则不合规
- 加固步骤
    - 执行命令：cp -p /etc/security/pwquality.conf /etc/security/pwquality.conf_bak 对文件进行备份
      执行命令：authconfig --passminlen=10 --passminclass=3 --update 设置密码长度最小值与密码所需要的最少字符类型并更新设置

7.检查密码重用是否受限制

- 执行命令
    - cat /etc/pam.d/password-auth 和 cat /etc/pam.d/system-auth 查看上述两文件中 password sufficient pam_unix.so 这行末尾是否存在 remember参数
- 基线合规性判定
    - 上述两文件中的password sufficient pam_unix.so 末尾存在 remember参数且值为5-24之间即合规，否则不合规。
- 加固步骤
    - 执行命令：vi /etc/pam.d/password-auth 和 vi /etc/pam.d/system-auth 分别找到两文件中的 password sufficient pam_unix.so 在该行末尾添加remember=5即可

二、服务配置

1.禁止ssh空密码登录

- 执行命令
    - cat /etc/ssh/sshd_config 查看文件中 PermitEmptyPasswords 参数是否为 no 且该行是否被注释
- 基线合规性判定

    - /etc/ssh/sshd_config 文件中 PermitEmptyPasswords 参数为 no 且该行未被注释即合规，否则不合规。
- 加固步骤
    - 执行命令：cp /etc/ssh/sshd_config /etc/ssh/sshd_config_bak
      对文件进行备份
      执行命令：vi /etc/ssh/sshd_config 将PermitEmptyPasswords配置为no 并删除注释符号 #
      执行命令：systemctl restart sshd 重启sshd服务使配置生效

2.设置ssh MaxAuthTries

- 执行命令
    - cat /etc/ssh/sshd_config 查看文件中 MaxAuthTries 参数是否为 3-6 之间
- 基线合规性判定
    - /etc/ssh/sshd_config 文件中 MaxAuthTries 参数为 3-6 之间且该行未被注释即合规，否则不合规。
- 加固步骤
    - 执行命令：vi /etc/ssh/sshd_config 将MaxAuthTries配置为3-6 之间，建议为 4 ，并删除注释符号 #
      执行命令：systemctl restart sshd 重启sshd服务使配置生效

3.设置ssh超时退出时间

- 执行命令
    - cat /etc/ssh/sshd_config 查看文件中 ClientAliveInterval 与 ClientAliveCountMax 参数
- 基线合规性判定
    - ClientAliveInterval 为300到900 之间，ClientAliveCountMax为0-3之间，且未被注释即合规，否则不合规
- 加固步骤
    - 执行命令：vi /etc/ssh/sshd_config 将ClientAliveInterval 设置为300到900 之间，
      ClientAliveCountMax设置为0-3之间，并删除注释符号 #
      执行命令：systemctl restart sshd 重启sshd服务使配置生效
      注：个人建议将ClientAliveCountMax 设置为0

4.ssh LogLevel设置为INFO

- 执行命令
    - cat /etc/ssh/sshd_config 查看文件中 LogLevel 参数
- 基线合规性判定
    - LogLevel设置为 INFO且未被注释即合规，否则不合规
- 加固步骤
    - 执行命令：vi /etc/ssh/sshd_config 将LogLevel INFO，并删除注释符号 #
      执行命令：systemctl restart sshd 重启sshd服务使配置生效

三、文件权限

1.访问控制配置文件的权限设置

- 执行命令
  - ll /etc/hosts.allow /etc/hosts.deny 查看文件权限与所有者等信息
- 基线合规性判定
  - /etc/hosts.allow 和 /etc/hosts.deny 的所有者与所有组均为root 且权限为 644 即合规，否则不合规
- 加固步骤
  - 执行命令：
    chown root:root /etc/hosts.allow /etc/hosts.deny
    chmod 644 /etc/hosts.deny /etc/hosts.allow
    为两文件设置权限及所有者等信息

2.设置用户权限配置文件的权限

- 执行命令
  - ll /etc/passwd /etc/shadow /etc/group /etc/gshadow 查看文件权限与所有者等信息
- 基线合规性判定
  - 上述文件的所有者和所有组都是root且 /etc/group /etc/passwd 的权限小于等于 644 ，/etc/shadow /etc/gshadow 的权限小于等于 400即合规，否则不合规
- 加固步骤
  - 执行命令：
    chown root:root /etc/passwd /etc/shadow /etc/group /etc/gshadow
    chmod 644 /etc/group /etc/passwd
    chmod 400 /etc/shadow /etc/gshadow
    为文件设置权限及所有者等信息

四、安全审计

1.确保rsyslog服务已启用

- 检查步骤
  - 执行命令： systemctl status rsyslog 查看 rsyslog 服务是否启动
    执行命令：systemctl is-enabled rsyslog查看rsyslog是否设置开机自启
- 基线合规性判定
  - 已启动rsyslog服务且设置了开机自启动即合规，否则不合规
- 加固步骤
  - 执行命令：
    systemctl enable rsyslog
    systemctl start rsyslog

五、入侵防范

1.开启地址空间布局随机化

- 解释
  - 地址空间布局随机化（ASLR）可以帮助克服某些类型的缓冲区溢出攻击，ASLR可以将基数，库，堆和堆栈放在进程地址空间中的任意随机位置，这使攻击程序很难预测下一条指令的内存地址。
  - ASLR内置在Linux内核中，并由参数控制 /proc/sys/kernel/randomize_va_space,该randomize_va_space参数可以采用以下值：
    - 0 : 禁用ASLR。如果使用norandmapsboot参数引导内核，则将应用此设置。
      1 : 随机化堆栈，虚拟动态共享对象（VDSO）页面和共享内存区域的位置。数据段的基地址位于可执行代码段的结尾之后。
      2 : 随机化堆栈，VDSO页，共享内存区域和数据段的位置。这是默认设置。
- 执行命令
  - cat /etc/sysctl.conf 查看文件当中是否存在kernel.randomize_va_space = 2
- 基线合规性判定
  - 存在kernel.randomize_va_space = 2即合规，否则不合规
- 加固步骤
  - 执行命令：vi /etc/sysctl.conf 在文件内添加一行 kernel.randomize_va_space = 2
    执行命令：sysctl -w kernel.randomize_va_space=2
    注：使用ldd命令就可以观察到程序所依赖动态加载模块的地址空间

说明：
根据编译原理可知，C语言程序中所有变量的内存地址编译后都是确定了的，但是在linux平台上实际使用时可以发现变量的内存地址并不是固定的，如下：

```
$ vi test.c
#include <stdio.h>

int main(){
    int a;

    printf("%p\n", &a);
    return 0;
}

$ gcc test.c -o test
$ ./test
0x7ffdd2ec0124
$ ./test
0x7ffc577a9644
$ ./test
0x7fffef5827e4
$ ./test
0x7fff44595414
$ ./test
0x7fffe93e1964
```