《操作系统》
实验指导书

南昌大学软件学院

2023 年

预备实验 1：Linux 常用命令使用

1.1 实验概述

实验目的：了解 Linux 操作系统的 Shell 命令格式，熟练掌握常用命令和选项的功能。

实验环境：linux

1.2 实验知识

Linux 操作系统与用户的常用接口分为图形界面和字符界面两种。图形界面的使用类似于 Windows 操作系统，即可以使用键盘和鼠标进行操作；而字符界面的功能十分强大，能够完成所有的任务。Linux 字符界面是通过 Shell 来实现相关功能的，Shell 既是用户和 Linux 内核之间的接口，也是命令语言、命令解释程序及程序设计语言的统称。Linux 中的 Shell 有多种类型，其中最常用的是 Bourne Shell (sh)、C Shell (csh)和 Korn Shell (ksh)，这 3 种 Shell 各具特色。目前 Linux 默认的 Shell 是 Bourne Again Shell (bash)，bash 是对 sh 的扩展。

本节只介绍 Linux 字符界面下常用命令的基本用法，详细用法（包括选项说明等）可参见相关帮助信息。常用命令主要包括：Linux 操作系统的启动与退出命令、文件和目录的操作命令、文档备份和压缩命令、权限改变命令、与用户有关的命令、磁盘管理命令、帮助命令等。

1.2.1 Linux 系统的启动与退出命令

1．启动 Linux

在开机启动 Linux 系统完毕后，需要输入用户名和口令。当用户输入正确的用户名和口令后,计算机就能合法进入系统,屏幕上将显示输入提示符“#”或“$”。

2．关闭和重启系统

关闭系统的命令包括“shutdown”、“halt”、“init 0”等。

重启系统的命令包括“reboot”、“init 6”、“shutdown”等。

其中,shutdown 命令根据选项的不同，既可以实现关机,又可以重启系统，还可以发出警告信息。

命令格式：

```
shutdown［选项］时间［警告信息］
```

举例:

shutdown -h now 【立即关机】

shutdown -r 23:59 & 【定时于 23:59 重启系统】

shutdown -k 3 Warning: System will shutdown!

【发送信息，提示 3 分钟后系统进入维护模式(3 分钟后系统会自动取消关闭操作)。】

1.2.2 文件和目录的操作命令

1.显示目录内容

命令格式：

```
ls［选项］［目录或文件］
```

说明：

对于目录，列出其中的所有子目录与文件；对于文件，输出文件名及其要求的其他信息。

举例：

ls -la 【以长格式显示当前目录中所有文件的详细信息。】

ls -ld /usr 【以长格式显示指定目录/usr 的信息。】

ls -as -S 【显示当前目录下所有子目录与文件以及它们所使用的空间。】

2.文件或目录的复制

命令格式：

```
cp［选项］源文件或目录 目标文件或目标目录
```

说明：

把指定的源文件复制到目标文件或把多个源文件复制到目标目录下。

举例：

cp file1.txt file2.txt 【将当前目录下的文件 file1.txt 复制成 file2.txt。】

cp -i /usr/file3.txt /home/abc/file4.txt 【将/usr 目录下的 file3.txt 复制到/home/abc 目录下,并取名为 file4.txt。若/home/abc 目录下已经存在 file4.txt 文件,则在覆盖该文件前询问用户。】

3. 文件或目录的更名与移动

命令格式：

```
mv［选项］源文件或目录 目标文件或目标目录
```

说明：

根据命令中第二个参数类型的不同(是目标文件还是目标目录),将文件更名或移至目标目录下。

举例：

mv file1 file2 【将当前目录下的文件 file1 更名为 file2。】

mv file .. 【将当前目录下的文件 file 移至上一级目录下。】

4．删除文件或目录

命令格式：

```
rm［选项］文件名或目录名
```

说明:删除当前目录下的一个或多个文件，也可删除目录。举例:

rm file1.txt 删除当前目录下的 file1.txt 文件。

rm-i *交互式删除当前目录下的所有非隐藏文件。hodunld-16rm -rf dir1 强制删除当前目录下的 dir1 子目录。

5．创建目录

命令格式：

```
mkdir［选项］dir-name
```

说明：

创建使用 dir-name 参数命名的目录。

举例：

mkdir dir1 【在当前目录下创建默认权限且名为 dir1 的子目录。】

mkdir -p newdir/subdir 【在当前目录下建立嵌套目录 newdir/subdir。】

mkdir -m744 dir 【在当前目录下建立名为 dir 的子目录，并要求 dir 子目录的所有者拥有读、写和执行权限,而同组用户与其他用户只有读权限。】

6．删除目录

命令格式：

```
rmdir［选项］dir-name
```

说明：

删除一个或多个使用 dir-name 参数指定的目录。需要注意的是，目录在被删除之前必须是空的。

举例：

rmdir dir1 【在当前目录下删除一个空的名为 dir1 的子目录。】

7．改变工作目录

命令格式：

```
cd［路径］
```

说明：

将当前目录改变为使用路径参数所指定的目录。

举例：

cd .. 【返回上一级目录。】

cd subdir 【进入当前目录的 subdir 子目录。】

8.显示当前工作目录

命令格式：

```
pwd
```

说明：

显示当前工作目录的绝对路径。

9．显示文件内容

命令格式：

```
cat [选项] 文件列表
```

说明：

显示文件列表中指定文件的内容，如果没有指定文件（或选项），就从标准输入中读取。

举例：

cat -b linuxbook.txt 【显示文本文件 linuxbook.txt 的内容，并在每行开头显示行号。】

cat > testfile.txt 【新建文本文件 testfile.txt。】

cat file1 file2 > file3 【将文件 file1 和 file2 的内容连接起来并存放在文件 file3 中。】

10．查找文件

命令格式：

```
find [选项] 目录列表
```

说明：

搜索文件并执行指定的查找操作。

举例：

find /usr -user user1 -print 【在/usr 目录中查找所有属于用户 user1 的文件。】

find . -name "*.txt" -print 【在当前目录及其子目录中查找所有以“.txt”为扩展名的文件,并打印文件名。】

find -name "*.txt" -mtime +2 -mtime -7 -print 【在当前目录及其子目录中查找所有以“*.txt”为扩展名并在两天以前、七天以内被修改过的文件。】

11.按指定模式查找文件

命令格式：

```
grep [选项] 字符串 文件列表
```

说明：

搜索文件中包含指定字符串的行并将其显示出来。

举例：

grep "test file" example.txt 【在当前目录下的 example.txt 文件中搜索与模式字符串“test file”相匹配的行。】

grep data * 【搜索当前目录下所有文件中与模式字符串 data 相匹配的行。】

12. 按页显示文件内容

命令格式：

```
more/less [选项] 文件名
```

说明：

按指定方式在屏幕上显示文本文件的内容。

举例：

more linuxbook.txt 【按页显示文本文件 linuxbook.txt 的内容。】

less linuxbook.txt 【与 more 命令类似，即按页显示文本文件 linuxbook.txt 的内容，但 less 命令具有更高级的功能——允许用户在文件中向前或向后导航。】

1.2.3 文档备份和压缩命令

1. 为文件或目录创建档案文件

命令格式：

```
tar [主选项+辅助选项] 文件或目录
```

说明：

为指定文件或目录创建档案文件（备份文件/打包文件）。

举例：

tar -cvf data.tar * 【将当前目录下的所有文件打包成 data.tar 文件。】

tar -cvzf data.tar.gz * 【将当前目录下的所有文件打包并通过调用 gzip 命令压缩成 data.tar.gz 文件。】

tar -xzvf data.tar.gz 【将打包后的压缩文件 data.tar.gz 解压缩。可以先调用 gzip 命令进行解压缩，之后再对打包文件进行解包。】

2. 文件压缩命令

命令格式：

```
gzip/bzip2/compress/zip/xz [选项] 压缩/解压缩的文件名
```

说明：

根据选项,对文件压缩或解压缩。需要说明的是,这些压缩命令的使用方法虽然类似，但并不完全相同，并且每个压缩命令所生成的压缩文件的后缀不同。

举例：

gzip -best data.txt 【以最高压缩比压缩文件 data.txt。】

gzip -d data.txt.gz 【解压缩文件 data.txt.gz。】

1.2.4 权限改变命令

1．改变文件或目录的访问权限

命令格式：

```
chmod [选项] 文件名
```

说明：

根据指定方式改变文件的属性。chmod 有两种使用方式：一种是文字设定法，包含字母和操作符表达式；另一种是数字设定法，包含数字。

举例：

chmod a+x test.sh 【为脚本文件 test.sh 的所有用户增加可执行属性。】

chmod go-rwx test 【取消其他用户对目录 test 的读、写和执行权限。】

chmod 0751 file1 【将文件 file1 设置为 rwxr-x--x 权限。】

2．改变文件或目录的属主和属组

命令格式:

```
chown [选项] 用户或组文件名
```

说明：

将指定文件的拥有者改为指定的用户或组。

举例：

chmod user1 file1 【将文件 file1 的属主改为 user1。】

1.2.5 与用户有关的命令

1.修改用户口令

命令格式：

```
passwd [用户名]
```

说明：

修改用户口令。普通用户只能修改自己的口令，超级用户可以修改指定用户的口令。

举例：

passwd user1 【修改用户 user1 的口令。】

2．切换用户

命令格式：

```
su [用户名]
```

说明：

切换成指定用户。普通用户在使用 su 命令时需要有超级用户或指定用户的口令。

举例：

su 【切换成超级用户 root。】

1.2.6 磁盘管理命令

1.检查文件系统的磁盘空间占用情况

命令格式：

```
df [选项]
```

说明：

查看文件系统的磁盘空间占用情况，包括已占用空间、剩余空间等信息。

举例：

df -aT 【显示当前系统中所有文件系统的类型与磁盘空间的占用情况。】

2.显示目录或文件所占磁盘空间的大小

命令格式：

```
du [选项] 文件或目录
```

说明：

统计文件或目录所占磁盘空间的大小。

举例：

du 【显示当前工作目录所占磁盘空间的大小。】

du -a / 【显示系统中所有子目录所占磁盘空间的大小。】

1.2.7 帮助命令

1. 普通格式的帮助命令

命令格式：

```
man [选项] 命令
```

说明：

列出命令的详细使用说明，包括命令语法、各选项的意义及相关命令。

举例：

man cp 【查看 cp 命令的使用说明。】

man -k printf 【以 printf 作为关键字查找对应的手册。】

2.info 格式的帮助命令

命令格式：

```
info [选项] 命令
```

说明：

以 info 格式列出命令的帮助文档。

举例：

info pwd 【查看 pwd 命令的帮助文档。】

1.3 实验内容

练习常用的Linux Shell 命令及命令选项，包括文件目录命令、备份压缩命令、重定向及管道命令等。要求熟练掌握下列命令的使用。

（1）改变及显示目录命令：cd、pwd、ls

（2）文件及目录的创建、复制、删除和移动命令：touch、cp、mv、rm、mkdir、rmdir

（3）显示文件内容命令：cat、more、less、head、tail

（4）文件查找命令：find、whereis、grep

（5）文件和目录权限改变命令：chmod

（6）备份和压缩命令：tar、gzip、bzip2

1.4 实验指导

开展实验前，将实验知识中的命令均练习一遍，并查看结果。

实验步骤：

（1）打开终端，在提示符下输入命令；

（2）执行每一条命令后，分析结果，修改选项后再次执行，查看并记录结果的变化。

预备实验 2： 在 Linux 下编写 C 程序

2.1 实验概述

实验目的：

- 掌握 Linux 下 C 程序的编写、编译与运行方法。
- 掌握 gcc 编译器的编译过程，熟悉编译的各个阶段。
- 熟悉 Makefile 文件的编写格式和 make 编译工具的使用方法。

实验环境：linux，gcc

2.2 实验知识

2.2.1 Linux 下 C 程序的编写与运行

C 语言是 Linux 操作系统下最常用的程序设计语言，也是操作系统层面编程最常用的程序设计语言。Linux 操作系统中的大多数应用都是用 C 语言编写的。

在 Linux 下，从编写到运行 C 程序一般分为以下几步。

1．编写 C 程序

编写 C 程序时，可以使用 Linux 下的文本编辑工具编辑文档，并将其保存为 C 程序，文件名后缀为“.c”（假设为 test.c）。Linux 下的文本编辑工具有很多，比如图形方式下的 emacs、gedit、kwrite 等，以及文本模式下的 vi、vim（vi 的增强版）和 nano 等。其中 vi/vim 是 UNIX/Linux 最基本的文本编辑器，几乎所有的 Linux 发行版本都提供这一编辑器，同时因不需要图形界面，它也是效率最高的文本编辑器。vi 的使用较复杂，读者可以通过阅读相关资料了解常见的 vi 用法。

2．编译 C 程序

为了运行 C 程序，必须对其进行编译、链接，使之成为可执行文件（假设名为 test）。目前，Linux 下最常用的编译器是 gcc，它能将用户使用高级语言编写的源代码编译成可执行的二进制代码。

3．运行 C 程序

一旦把 C 程序成功编译为可执行文件，即可通过 Linux 下的 Shell 执行该程序，具体执行方式为：./可执行文件名（如./test）。

2.2.2 gcc 的使用

gcc 的命令格式：

gcc [选项] 源文件 [目标文件]

其中的“选项”为编译器所需要的编译选项。gcc 编译器的选项大约有 100 多个，但大多数选项不会被用到，最基本、最常用的选项如下所示。

-C：只编译，不链接成可执行文件。编译器只生成.o 后缀的目标文件，通常用于不包含主程序的子程序文件。

-o file：确定输出文件的名称为 file，该名称不能和源文件同名。若没有该选项，默认将生成可执行文件 a.out。

-Idirname：指定头文件的查找目录，将 dirname 指定的目录加入程序头文件目录列表中，在预编译过程中使用。

-Ldirname：指定库文件的查找目录，将 dirname 对应的目录加入程序函数档案库文件的目录列表中，在链接过程中使用。

-lname：在链接过程中，加载名为“libname.a”的函数库（位于系统预设的目录或由-L 选项确定的目录下）。

-Wall：编译文件时发出所有警告信息。

-w：编译文件时不发出任何警告信息。

和其他编译器一样，gcc 提供了灵活而强大的代码优化功能。利用它可以生成执行效率更高的代码。gcc 对标准 C 和 C++进行了大量的扩充，提高了程序的执行效率；此外还对代码进行了优化，减轻了编程工作量。常用的调试优化参数

如下。

-g：产生符号调试工具（如 GNU 的 gdb）必要的符号信息。可在对源代码进行调试时加入该选项。

-O：在编译、链接过程中进行优化处理，从而提高可执行文件的执行效率，但编译、链接的速度相应要慢一些。

-O2：能比-O 更好地对编译、链接过程进行优化。

gcc 还包含完整的出错检查和警告提示功能，以帮助程序员写出更加专业、优美的代码。

2.2.3 Makefile 文件的编写

使用 C/C++语言编写的工程项目通常包含很多个源代码文件。它们均按不同类型、功能放在不同的目录下。Linux 下没有支持 C/C++语言的集成环境，当项目源代码文件较少的时候，可以直接使用 gcc 命令编译；但对于复杂工程项目，直接使用 gcc 命令编译就很困难。因此 Linux 提供了 make 工具来支持工程项目的编译任务。它是一个用来控制从程序的源文件中生成程序的可执行文件和其他非源文件（如汇编文件、目标文件等）的工具。

make 工具根据 Makefile 文件的内容来构建程序。Makefile 文件描述了每一个非源代码文件以及从其他文件构造这些文件的方法。make 命令在被执行时，会扫描当前目录下的 Makefile（或 makefile）文件，找到目标及其依赖关系。如果这些依赖自身也是目标，就继续为这些依赖扫描 Makefile 文件，并建立其依赖关系，然后编译它们。

Makefile 文件定义了一系列的规则来告诉 make 何时以及如何生成或更新目标文件，此外还告诉 make 执行何种命令来对可执行文件进行管理，如执行文件、清除目标文件等。其中规则的一般形式如下。

```
    target：依赖文件列表
    <TAB>执行命令
    ...
```

target 是目标或规则的名称，通常是一个程序即将产生的文件的名称（例如可执行文件或目标文件的名称）。同时，target 也可以是一个即将执行的活动的名称，如 clean，其表示执行清除活动。

观察如下 Makefile 文件中的内容：

```
main: main.o hello1.o hello2.o
    gcc -o main main.o hello1.o hello2.o
main.o: main.c hello1.h hello2.h
    gcc -c main.c
hello1.o: hello1.c hello1.h
```

```
    gcc -c hello1.c
hello2.o: hello2.c hello2.h
    gcc -c hello2.c
clean:
    rm main hello1.o hello2.o main.o
```

Makefile 文件主要包括 5 个部分：显式规则、变量定义、隐式规则、指令及注释。

（1）显式规则：告诉 make 何时以及如何重新编译或更新一个或多个目标文件。例如：

```
hello1.o: hello1.c hello1.h
    gcc -c hello1.c
```

以上就是一条显式规则，它列出了目标所依赖的文件，并且给出了用于创建或更新目标的命令。

（2）变量定义：为一个变量指定一个字符串，在执行 make 命令时，该变量将被其所代表的字符串替换。

（3）隐式规则：指出何时以及如何根据名称重新编译或更新一类文件。隐式规则描述了目标如何依赖一个与目标名称相似的文件,并且给出了创建或更新目标的命令。

（4）指令：当使用 make 读取 Makefile 文件时，指令用来告诉 make 执行一-些特殊活动，例如：

①读取其他 Makefile 文件；

②根据变量的值决定是忽略还是使用 Makefile 文件中的部分内容；

③定义多行变量。

（5）注释：Makefile 文件中的注释以“#”开头，表示该行将在执行时被忽略。

前文所述的 Makefile 文件可以根据变量定义、自动推导和隐式规则进行修改，例如改写成如下内容：

```
obj=main.o hello1.o hello2.o     #变量定义
main: $(obj)
    gcc -o main $(obj)
hello1.o: hello1.h     #自动推导和隐式规则
hello2.o:hello2.h
clean:
    rm $(obj)
```

Makefile 文件的编写相对复杂，包括对规则编写、指令使用、函数调用等的

详细说明等。同学们可参阅其他相关帮助文档加以了解。

2.3 实验内容

练习使用 gcc 编译器编译 C 程序并执行，编写 Makefile 文件，使用 make 工具编译程序并执行

具体要求：

（1）编写简单的 C 程序，功能为在屏幕上输出“Hello gcc！”。利用该程序练习使用 gcc 编译器的 E、S、c、o、g 选项，观察不同阶段所生成的文件，即*.c、*.i、*.s、*.o 文件和可执行文件。

（2）编写一个由头文件 greeting.h、自定义函数文件 greeting.c、主函数文件 myapp.c 构成的 C 程序，并根据这三个文件的依赖关系编写 Makefile 文件

2.4 实验指导

对于实验内容（1），可将其分为三个步骤：

①创建空文档，修改名称为 myhello.c，输入程序代码，保存并退出；

②打开终端，用 gcc 命令对 myhello.c 程序进行分阶段编译；

③利用 ls 命令查看编译过程中所产生的各个文件，即 myhello.i、myhello.s、myhello.o 文件和可执行文件（如 myhello.c）。

除 3 个源代码文件外，最重要的是 Makefile 的编写，最后使用 make 工具编译程序。