Linux基础 2022-2023

1

> Q：使用命令useradd webftp 可以创建一个名为webftp的用户，一般来说，该用户的主目录为什么，该用户注册的账户信息存储在什么文件中，为了口令的安全，用户的口令一般单独存放在什么文件中

一般情况下，使用命令 useradd webftp 创建名为 "webftp" 的用户时，该用户的主目录通常被设置为 /home/webftp。主目录是用户登录后默认进入的目录，也是用户存储自己文件的位置。

用户注册的账户信息通常存储在系统中的密码文件中。在大多数Linux系统中，用户账户信息被记录在 /etc/passwd 文件中。每一行代表一个用户账户，包含了用户名、用户ID、主目录、登录Shell等信息。

为了口令的安全性，用户的口令通常单独存放在一个称为 "shadow" 文件的文件中。在大多数Linux系统中，用户口令被存储在 /etc/shadow 文件中。这个文件对普通用户是不可读的，只有特定的系统管理员或具有特权的用户才能访问它。通过将口令单独存放在该文件中，可以增加口令的安全性，限制对敏感信息的访问。

2

> Q：linux系统服务脚本通常位于什么目录，查看系统运行级别的命令

Linux系统服务脚本通常位于 `/etc/init.d/` 目录下。这个目录包含了系统中安装的服务所使用的启动脚本。每个服务都有一个相应的启动脚本，用于控制服务的启动、停止、重启等操作。

要查看系统运行级别，可以使用以下命令：

```
runlevel
```

或者使用 `who -r` 命令也可以查看当前的运行级别。

运行上述命令后，会显示当前系统的运行级别。运行级别是一个标识系统处于哪种状态的数字或字符。不同的发行版可能使用不同的运行级别编号和表示法，但常见的运行级别包括 0（关机）、1（单用户模式）、2（多用户模式无网络服务）、3（多用户模式带网络服务）等。

3

> Q：Linux系统作为一个多用途的操作系统，在网络领域提供了许多常见的网络服务。以下是一些常见的网络服务，可以在基于Linux的系统上提供：

Web服务器（HTTP）：例如Apache HTTP Server（httpd）、Nginx等。
文件传输协议（FTP）服务器：例如vsftpd、ProFTPD等。
邮件服务器（SMTP/POP/IMAP）：例如Postfix、Exim、Dovecot等。
DNS服务器：例如BIND（named）、PowerDNS等。
DHCP服务器：例如ISC DHCP Server、dnsmasq等。

文件共享服务（SMB/CIFS）：例如Samba。
远程登录服务（SSH）：例如OpenSSH。
网络时间协议（NTP）服务器：例如NTPd、Chrony等。
虚拟专用网络（VPN）服务器：例如OpenVPN、IPsec等。
网络监控服务（SNMP）：例如Net-SNMP。
网络文件系统（NFS）服务器：例如NFS、NFSv4等。
数据库服务：例如MySQL、PostgreSQL、MongoDB等。

4

> Q：在Linux系统中，根目录（Root Directory）是整个文件系统的顶级目录，表示为 `/`。根目录下有许多主要的目录，每个目录都有其特定的作用。以下是一些常见的主要目录及其主要作用：

1. `/bin`：该目录包含了可执行的二进制文件，这些文件是系统启动和基本功能所必需的。例如，常用的命令如 `ls`、`cp`、`mv` 等就位于此目录。
2. `/boot`：该目录包含了引导（boot）相关的文件，如内核映像文件（vmlinuz）和引导加载程序（bootloader）配置文件。这些文件用于系统启动过程。
3. `/dev`：该目录包含了设备文件，用于与硬件设备进行交互。例如，硬盘、键盘、鼠标等设备在此目录下有相应的设备文件。
4. `/etc`：该目录包含了系统的配置文件。各种系统服务、应用程序和组件的配置文件通常存放在此目录下。
5. `/home`：该目录是用户的主目录，每个用户都有一个对应的子目录在其中。用户可以在自己的主目录下存储个人文件和配置。
6. `/lib`：该目录包含了共享库文件，这些文件是用于支持可执行程序的共享代码和函数。
7. `/mnt`：该目录通常用于临时挂载文件系统，例如将外部存储设备挂载到该目录下。
8. `/opt`：该目录用于安装可选软件包，通常是第三方提供的软件，不属于系统默认安装的一部分。
9. `/proc`：该目录提供了进程信息的虚拟文件系统，可以通过读取和修改这些文件来访问和管理运行中的进程。
10. `/root`：该目录是root用户的主目录，类似于普通用户的 `/home` 目录。
11. `/sbin`：该目录包含了系统管理员使用的系统命令和脚本，这些命令通常需要管理员权限才能执行。
12. `/tmp`：该目录用于存储临时文件，这些文件可能会在系统重启后自动删除。
13. `/usr`：该目录包含了用户应用程序和文件，类似于系统的第二层次结构。其中，`/usr/bin` 存放用户可执行文件，`/usr/lib` 存放用户共享库文件，`/usr/share` 存放共享数据等。
14. `/var`：该目录用于存储可变数据，例如日志文件、数据库文件、邮件等。

这些是根目录下的一些主要目录及其主要作用。每个目录在Linux系统中都有其特定的用途和目标。

5

> Q: tar打包解包

在Linux中，常用的打包和解包命令是 `tar`。`tar` 命令用于创建和提取归档文件，可以将多个文件或目录打包成一个单独的文件，或从归档文件中提取文件。

以下是一些常见的 `tar` 命令及其参数：

1. 创建归档文件：

```
tar -cvf archive.tar file1 file2...            # 创建.tar文件
tar -czvf archive.tar.gz file1 file2...        # 创建.tar.gz（压缩）文件
tar -cjvf archive.tar.bz2 file1 file2...       # 创建.tar.bz2（压缩）文件
```

2. 提取归档文件：

```
tar -xvf archive.tar               # 提取.tar文件
tar -xzvf archive.tar.gz           # 提取.tar.gz（压缩）文件
tar -xjvf archive.tar.bz2          # 提取.tar.bz2（压缩）文件
```

3. 列出归档文件内容：

```
tar -tvf archive.tar               # 列出.tar文件内容
tar -tzvf archive.tar.gz           # 列出.tar.gz（压缩）文件内容
tar -tjvf archive.tar.bz2          # 列出.tar.bz2（压缩）文件内容
```

4. 向已存在的归档文件添加文件：

```
tar -rvf archive.tar file1 file2...            # 添加文件到.tar文件
tar -rzvf archive.tar.gz file1 file2...        # 添加文件到.tar.gz（压缩）文件
tar -rjvf archive.tar.bz2 file1 file2...       # 添加文件到.tar.bz2（压缩）文件
```

这些是 `tar` 命令的一些常见用法和参数。`tar` 命令还有其他选项和参数，可以根据具体需求进行调整和使用。可以使用 `man tar` 命令查看更详细的手册页来了解更多关于 `tar` 命令的信息。

6

> Q: ls -l文件操作

在使用 `ls -l` 命令时，输出结果的各列含义如下：

1. 列 1：文件类型和权限信息
   - `-` 表示普通文件
   - `d` 表示目录
   - `l` 表示符号链接（软链接）
   - `c` 表示字符设备文件
   - `b` 表示块设备文件
   - `p` 表示命名管道（FIFO）
   - `s` 表示套接字（Socket）
2. 列 2：硬链接的数量
   - 指示有多少个硬链接指向该文件或目录。对于目录来说，该值通常至少为2，因为 `.`（当前目录）和 `..`（上级目录）都是目录的硬链接。
3. 列 3：所属用户（所有者）

    - 显示文件或目录的所有者用户名。
4. 列 4：所属组
    - 显示文件或目录所属的用户组名称。
5. 列 5：文件大小（以字节为单位）
    - 显示文件的大小。对于目录来说，通常显示为0。
6. 列 6：修改时间
    - 显示文件或目录的最后修改时间。
7. 列 7：文件名或目录名
    - 显示文件或目录的名称。

例如，以下是 `ls -l` 命令的示例输出：

```
-rw-r--r--  1 user group  4096 Dec 24 10:30 file.txt
drwxr-xr-x  2 user group  4096 Dec 24 10:30 directory
lrwxrwxrwx  1 user group    10 Dec 24 10:30 link -> file.txt
```

在上面的示例中，第一列的 `-` 表示普通文件，`d` 表示目录，`l` 表示符号链接。第二列显示了硬链接的数量。第三列和第四列分别显示了文件的所有者和所属组。第五列显示了文件的大小。第六列显示了最后修改时间。第七列显示了文件或目录的名称。

7

> Q： 软硬链接

软链接和硬链接是在文件系统中创建链接的两种方式。它们都可以用来创建文件或目录的别名，但它们之间有一些关键的区别。

软链接（Symbolic Link）是一个指向文件或目录的符号链接。它类似于Windows系统中的快捷方式。软链接创建一个新的文件，该文件包含指向源文件或目录的路径。软链接是一个特殊的文件，它只包含源文件或目录的路径信息，并不实际存储文件数据。当访问软链接时，系统会自动跟随链接并访问源文件或目录。

下面是软链接的使用和删除的示例：

创建软链接：
ln -s /path/to/source /path/to/symlink

这将在指定路径下创建一个名为`symlink`的软链接，指向源文件或目录。

使用软链接：
当你访问软链接时，实际上是在访问源文件或目录。对软链接的任何更改都会反映在源文件或目录上。

删除软链接：

```
rm /path/to/symlink
```

这将删除软链接，但不会影响源文件或目录本身。
硬链接（Hard Link）是一个指向文件的物理链接。它创建一个与源文件相同的新的文件入口，并且两个文件入口指向相同的文件数据。换句话说，硬链接是文件系统中的两个文件入口，它们共享相同的文件内容。

下面是硬链接的使用和删除的示例：

创建硬链接：

ln /path/to/source /path/to/hardlink

> 这将在指定路径下创建一个名为`hardlink`的硬链接，它与源文件共享相同的文件内容。
>
> 使用硬链接：
> 无论你通过源文件或硬链接访问文件，实际上都是在访问相同的文件内容。对任何一个文件入口的更改都会反映在另一个文件入口上。
>
> 删除硬链接：
>
> ```
> rm /path/to/hardlink
> ```

删除硬链接并不会影响源文件或其他硬链接。只有当所有链接（包括源文件和其他硬链接）被删除时，文件的实际内容才会被释放。
需要注意的是，软链接可以跨越文件系统边界，而硬链接只能在同一文件系统中创建。另外，对源文件的重命名或移动并不会影响已创建的硬链接，因为它们与文件内容关联，而不是文件路径。

总结：
软链接是一个指向源文件或目录的符号链接，它创建一个新的文件，当访问软链接时，系统会跟随链接并访问源文件或目录。删除软链接不会影响源文件或目录。
硬链接是源文件的物理链接，它创建一个与源文件相同的新的文件入口，两个文件入口共享相同的文件内容。删除硬链接不会影响源文件，只有当所有链接都被删除时，文件的实际内容才会被释放。

8

要启动和关闭 vsftpd（Very Secure FTP Daemon），可以使用以下命令：

启动 vsftpd 服务：sudo service vsftpd start

关闭 vsftpd 服务：sudo service vsftpd stop
这些命令假设你的系统上已安装并配置了 vsftpd 服务。请确保在执行上述命令时使用了管理员权限（sudo）。根据你的系统和配置，可能会有一些差异，但通常情况下，这些命令可以启动和关闭 vsftpd 服务。

9

> Q:举例说明linux shell中输入输出的实现

标准输入（stdin）：可以通过键盘输入数据。例如，使用 read 命令读取用户的输入：

```
echo "Please enter your name:"
read name
echo "Hello, $name!"
```

标准输出（stdout）：默认情况下，命令的输出会发送到终端。例如，使用 echo 命令输出文本：

```
echo "Hello, World!"
```

10

> Q: 关于shell脚本

(1)

```
lines=0
for f in *.c
do
ln=`wc  -l $f | awk '{print $1}'`
lines=$((lines+ln))
done
echo "Total lines: ${lines}."
```

lines=0：初始化变量lines为0，用于存储总行数。

for f in *.c：*通过通配符*.c遍历当前目录下所有以".c"为扩展名的文件，将每个文件名赋值给变量f。

ln=$(wc -l $f | awk '{print $1}')：使用wc -l命令获取文件$f的行数，并通过管道将输出传递给awk命令。awk '{print $1}'提取wc输出的第一个字段，即文件行数，并将其赋值给变量ln。

lines=$((lines+ln))：将变量ln的值加到lines上，以累加该目录下所有 .c 文件的行数。

echo "Total lines: ${lines}."：输出总行数。

(2)

> 脚本里写. /lib/lsb/init-functions

使用"./lib/lsb/init-functions"是为了在脚本中引入或执行指定路径下的脚本文件。通过这种方式，脚本可以使用"init-functions"脚本中定义的函数或变量，或者执行其中的命令。

> UPSTART=
> [-x /sbin/initctl] && UPSTART=yes

1. `[-x /sbin/initctl]`：这是一个条件判断语句的一部分，使用了`-x`参数来检查`/sbin/initctl`文件是否存在并且可执行。
2. `&&`：这是一个逻辑运算符，表示逻辑与操作。在这里，它用于连接两个条件判断语句。
3. `UPSTART=yes`：这是一个赋值语句，将变量`UPSTART`的值设置为`yes`。

   跟以下写法一致

```
if [ -x /sbin/initctl ]; then
  UPSTART=yes
fi
```

第七章了解基本语法 shell的for语句

其余脚本难得地方是书上P178页的测试语句

ps,top命令，src源码编译，vi的149页三角图
87页的文件权限

shell 执行的三种方式

等等等等自己背