

Linux 101 文件操作与软件安装*

*以Debian buster为例

目录

文件操作

编辑文件

处理文件中的数据

软件包管理及编译安装

环境变量与命令的位置

创建、复制、 移动和删除文件

创建一个空文件：\$ touch a.txt

向文件中写入一段文字：\$ printf "Linux 101\nHello World!\n">a.txt

将文件在屏幕上打印出来：\$ cat a.txt 应当出现

```
Linux 101
Hello World!
```

创建一个目录：\$ mkdir a

将a.txt复制进目录：\$ cp a.txt a/

或者可以指定文件名：\$ cp a.txt a/a_copy.txt

列出本目录和 a/ 下的内容，应当都含有 a.txt：\$ ls ./ a/

创建、复制、 移动和删除文件

删除 a/ 目录下的 a.txt : `$ rm a/a.txt`

将 a.txt 移动至 a/ : `$ mv a.txt a/[a.txt]`

列出本目录和 a/ 下的内容, 应当只有 a/ 目录含有 a.txt

用上面学到的方法将 a 移动或复制出来


删除 a/ 目录及其内部所有文件: `$ rm -rf a/`




在使用 `rm -rf` 时千万要小心, 不要不小心加不该加的空格! 如果目录是空的, 请尽量使用 `rmdir`

国际表情包一览:

<https://github.com/MrMEEE/bumblebee-Old-and-abandoned/commit/a047be85247755cdbe0acce6>


创建、复制、 移动和删除文件


 MrMEEE / [bumblebee-Old-and-abbandoned](#)


 Watch ▾ 10  Star 388  Fork 45

[<> Code](#) [🔔 Issues 17](#) [🔗 Pull requests 0](#) [📁 Projects 0](#) [📖 Wiki](#) [📊 Insights](#)

install script does `rm -rf /usr` for ubuntu #123 [New issue](#)

 **Closed** ginoputrino opened this issue on May 24, 2011 · 34 comments









ginoputrino commented on May 24, 2011 

An extra space at line 351:
`rm -rf /usr /lib/nvidia-current/xorg/xorg`

causes the install.sh script to do an `rm -rf` on the `/usr` directory for people installing in ubuntu.

Totally uncool dude!!! The script deletes everything under `/usr`. I just had to reinstall linux on my pc to recover.

Removing the space will fix this. Probably should do it quickly!!!

 89  15  195  73  15  95

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

下载与上传文件

从互联网上下载文件: `$ wget`

`https://jaist.dl.sourceforge.net/project/qbittorrent/qbittorrent/qbittorrent-4.1.5/qbittorrent-4.1.5.tar.xz`

将这个文件上传至服务器: `$ scp qbittorrent-4.1.5.tar.xz
[your-username@]your-server-addr:~/`

scp和cp的用法非常类似。相应的, 从服务器下载文件可以使用
: `$ scp
[your-username@]your-server-addr:~/ffmpeg-3.4.2.tar.bz2 ~/`

如果你记不住文件名, 可以使用 `sftp`: `$ sftp
[your-username@]your-server-addr`

使用 `get` 来下载, 使用 `put` 来上传

编辑文本

通常而言Linux发行版会自带nano和vi两种编辑器。原生的vi非常不好用，但是其衍生版vim则是许多人常用的编辑器。

编辑文件: `$ vi/vim/nano <filename>`

如果文件不存在，会自动创建，但是目录则不会

nano类似于windows系统的记事本，直接按照页面下方的指示操作即可

对于vim，最开始打开一个文件的时候进入的是“普通模式”，可以按i或者a进入插入模式。按ESC返回普通模式。

更多的vim详情可以参考上次小聚李嘉豪同学的讲座
http://ftp.ustclug.org/weekly_party/2018.04.01_Vim%26Emacs/vim/slides.html

数据处理

将a.txt复制为b.txt, 然后将b.txt末尾的空格删去

比较两个文件的文本内容的差异: `$ diff a.txt b.txt`

将b.txt末尾的空格删除, 并在两行之间插入一个空行, 比较差异。

将b.txt中的空行删除, 并将World改为Wrold, 分别用diff和cmp比较差异: `$ cmp a.txt b.txt`

将b.txt中的101改为102, 用cmp比较差异

cmp会输出两个文件的第一个在字节上不同的地方

打包与压缩

Linux下通常会使用tar进行打包: `$ tar -cf a.tar a.txt b.txt`

如果要列出每个处理的文件, 请使用-v参数。如果要压缩, 请使用下面的参数:

-j bzip2

-J xz

-z gzip

其中bzip2使用bzip2算法, xz使用LZMA算法, gzip使用DEFLATE算法。

软件包管理

Debian及其衍生版使用 apt来管理软件包

更新软件缓存: # apt-get update

安装软件包: # apt-get install <软件包名>

更新软件包: # apt-get upgrade

完全更新: # apt-get dist-upgrade

upgrade和dist-upgrade的区别: 如果一个软件包的依赖没有改变, 那么可以通过upgrade参数来更新。如果一个软件包的依赖改变了, 则需要通过dist-upgrade参数来更新。

dist-upgrade主要用于更新内核等软件包

软件包管理

搜索一个软件包: `$ apt-cache search <软件包名>`

列出一个软件包的信息(版本、依赖等): `$ apt-cache show <软件包名>`

卸载软件包: `# apt-get remove <软件包名>`

移除软件包的配置文件: `# apt-get remove <软件包名> --purge`

列出全部已安装的软件包: `$ dpkg -l`

安装一个已经下载到本地的软件包: `# dpkg -i <文件名>`

重新配置软件包: `# dpkg-reconfigure <软件包名>`

编译安装

有的软件不在软件源中

为了能够自行编译安装, 需要先安装 build-essential

注: 只有 Debian 系的发行版的 make 在这个包中。如果你使用 CentOS/RHEL, 请先执行 `# yum groupinstall 'Development Tools'`

解压前面下载的 ffmpeg: `$ tar -Jxf qbittorrent-4.1.5.tar.xz`

进入目录并执行 `./configure --disable-gui`

出错了！

解决依赖问题

先在软件源中搜索缺失的软件包

如果提示的是缺失某个头文件，则可以执行apt-file

```
# apt-file update
```

```
$ apt-file search <缺失的头文件>
```

安装该软件包

再执行./configure

编译安装

如果没有报错, 下一步则是make

有的开发者会提供make check

完成之后, # make install

或者直接运行

对于大部分软件, 卸载可以使用 make uninstall

命令的位置与 环境变量

所安装的软件一定有位置供其存放

查看命令安装的位置: `$ which <命令名>`

locate类似于windows上的搜索功能。执行`$ locate ls`

报错了吗？

```
# updatedb
```

再执行, 一长串

查看环境变量: `$ env`

如果只需要查看PATH则执行 `$ echo $PATH`

命令的位置与 环境变量

进入刚才编译ffmpeg的目录

直接执行configure无法执行

备份: `$ export pathbk="$PATH"`

`$ export PATH="./:$PATH"`

现在可以执行了

这样不安全, 因此恢复之前的 环境变量: `$ export PATH="$pathbk"`