

**软件学院学生实验报告**

**实验课程名称：** **操作系统** **教师： 林鑫泓**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验项目名称** | **操作系统用户接口实验** | | | **实验成绩** | **A** |
| **学生姓名** |  | **学 号** | **014** | **年级专业班级** |  |
| **小组成员** | **无** | | | **实验日期** | **17年9月21日** |

**一、实验目的**

* 熟悉操作系统的命令、图形、程序接口

**二、实验软硬件环境**

个人笔记本（win10）、VM虚拟机、Ubuntu16.04.3

**三、实验内容及要求**

* 安装操作系统（选做）
* 熟悉操作系统的命令、图形、程序接口

**四、实验记录**

**1、Ubuntu的命令接口**

**1.1、命令接口介绍**

命令行接口（command line interface，CLI）是计算机操作系统或应用程序为用户提供的可视提示接口，使用CLI，可以在特定的行中输入命令，从操作系统接收回应，然后再次输入其它命令等。

**1.2、命令行解释器介绍**

**A、**命令行解释器是一个单独的软件程序，它可在用户和操作系统之间提供直接的通讯。命令行解释器是解释器的一种，用于对命令行进行解释执行。

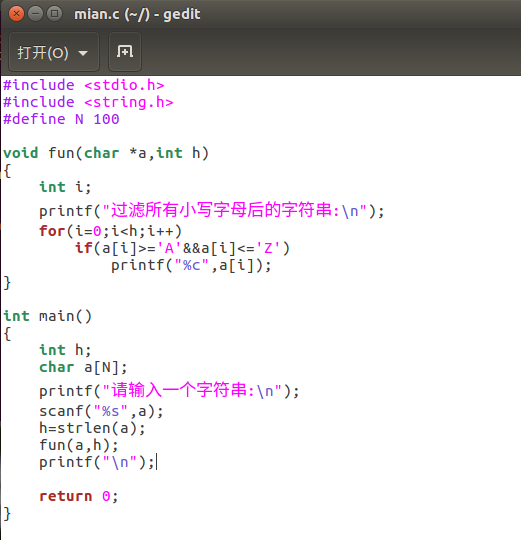
**B、**Linux下的Shell是系统的用户界面，提供了用户与内核进行交互操作的一种接口。它接收用户输入的命令并把它送入内核去执行。

**1.3、Linux命令一览表**

|  |  |  |  |
| --- | --- | --- | --- |
| **命令类别** | **命令** | **参数** | **含义** |
| **字符模式中字符界面切换命令** | **当前模式为字符模式时，按下Alt+F1～Alt+ F6** | **无** | **打开一个新的字符界面（有些计算机需要再同时按一个shift，请读者操作时注意）。** |
| 当前模式为图形模式时，按Ctrl+Alt+F1～CIrI+Alt+F6 | 无 | 切换至控制台(即字符模式)，之后想再返回到图形界面，可以直接按Alt+F7。 |
| vi编辑器 | **vi的启动：如vi test.txt**  **vi的命令模式**：在命令模式下，用户的键盘操作被视为命令，而不是内容。如要删除字符、想进行移动和复制某段文字等。  **vi的插入模式**：在插入模式下，可以对文件内容进行编辑。在插入模式下按esc键可切换到命令模式。  **vi的末行模式**：末行模式可以完成退出、查找、环境设置等功能，此模式的功能最复杂，涉及的方面也多，本教程中仅对较常用的功能进行讲解。在命令模式下键入“:”或“/”或者“?”会发现光标从原来的位置移到了屏幕的最下面一行，即末行模式。 | 无 | 打开vi编辑器，编辑test.txt文件，默认进入模式为命令模式 |
| 命令模式切换到插入模式的常用命令如下：  i：在光标之前输入字符。  a：在光标之后输入字符。  o：在光标所在行的下面新建一行的第一个位置输入字符。  I：光标移动到本行的行首，开始输入。  A：光标移动到本行的末尾，开始输入。  O：在光标所在行的上面新建一行的第一个位置输入字符。  在vi中y表示复制，d表示删除，p表示粘贴。  命令模式下复制粘贴等命令如下：  yw：复制光标为首的单词。  yy：复制光标所在行。  nyy：复制光标行在内的以下n行，命令也可写为yny。  dw：删除光标之后的一个单词。  dd：删除光标所在行。  ndd：删除光标以下n行。  p：粘贴。  u：撤销最后一次修改，Ctrl+R回复。  .：小数点，重复执行最后执行的操作。 | 无 | 命令模式下移动光标的常用命令如下：  移动光标：通过键盘方向键或k、j、h、l键实现光标移动，相当于键盘上的上、下、左、右。  M：将光标移到当前屏幕的中间一行的行首。M表示Middle。  L：将光标移到当前屏幕的最下行的行首。L表示Lowest。  H：将光标移到当前屏幕的最上行的行首。H表示Highest。  0：数字0，左移光标到本行的开头。  $：右移光标到本行的末尾。  G：光标移至文件末尾。  nG：快速定位至第n行位置。  gg：光标移至文件首行。  Ctrl+b：在文件中向上翻一页，相当于PageUp键。  Ctrl+f：在文件中向下翻一页，相当于PageDown键。 |
|  | **vi的末行模式下的一些命令：**  :w：保存当前文件，然后可以继续编辑。  :q：退出vi。  :x：命令x= wq，保存文件并退出。  :q!：不保存文件并退出Vi。  :e!：放弃所有修改，从上次保存文件开始再编辑。  /string：向下查找string中指定的字符串，按“n”键继续向下查找，shift +n反方向查找。  ?string：向上查找string中指定的字符串。  :起始行号,结束行号 s/被替换串/新串/g：用于替换字符串，无/g时只替换行中第一个，使用/g将替换所有指定字符串。  :起始行号,结束行号 y：复制指定的多行。  :起始行号,结束行号d：删除指定的多行。  :起始行号,结束行号 g/字符串/d：删除带有某字符串的多行。  :set nu：显示行号，:set nonu则关闭行号显示。  :set ai：自动缩进，当前行与上一行行首对齐，:set noai取消 |  |  |
| init程序与运行级别 | id:3:initdefault | 无 | 修改默认启动级别 |
| runlevel | 无 | 查看当前运行级别 |
| vi /etc/inittab | 无 | 编辑配置文件 |
| init 0 | 无 | 停机(千万不能把initdefault 设置为0) |
| init 1 | 无 | 单用户模式闪吧新社区B"e |:Ra­U`t0|(I |
| init 2 | 无 | 多用户，没有 NFS(和级别3相似，会停止部分服务)闪吧新社区 dYbR/Z1pK$L!V |
| init 3 | 无 | 完全多用户模式，字符模式 |
| init 4 | 无 | 没有用到闪吧新社区 IrK+b\*x r |
| init 5 | 无 | x11(Xwindow) |
| init 6 | 无 | 重新启动(千万不要把initdefault 设置为6) |
| 计算机使用基本命令 | hostname 计算机名 | 无 | 临时修改计算机名，计算机重启后失效 |
| 使用命令vi /etc/sysconfig/network，假设计算机名是test.com，将HOSTNAME=localhost.localdomain修改为HOSTNAME=test.com即可。 | 无 | 永久更改计算机名 /etc/sysconfig/network |
| ifconfig 网络接口名（eth0）  网络接口名可以省略，如省略则查看所有活动的网络接口。  ifconfig只能看到活动网络接口的IP，若想查看所有网络接口的IP，而不论网络接口是否处于活动状态，可以使用ifcofnig –a命令。 | 无 | 查看网卡1的ip地址 ifconfig eht0 |
| ifconfig eth0 IP地址 netmask 子网掩码 | 无 | 临时设置网卡1的ip地址和子网掩码 ifconfig eht0: 172.  netmask 255.255.255.0 |
| ifconfig eth0:0 192.168.0.212 netmask 255.255.255.0 | 无 | 为网卡1再添加一个IP地址 |
| ifconfig eth0 up | 无 | 激活网卡1 ifconfig eht0 up; ifup eth0 |
| ifconfig eth0 down | 无 | 禁用网卡1 |
| vi /etc/sysconfig/network-scripts/ifcfg-eth0 | 无 | 永久设置IP地址 方式一：setup，方式二：配置文件：  etc/sysconfig/network-scripts/ifcfg-eth0 |
| ifup eth0 | 无 | 激活网卡1 |
| ifdown eth0 | 无 | 禁用网卡1 |
| service network restart | 无 | 重启网络服务 service network restart |
| setup | 无 | 图形设置IP和DNS setup |
| vi /etc/resolv.conf  若需要多个DNS，可以另起一行，再写入一行“nameserver DNS的IP地址”修改完后，保存退出。 | 无 | 修改DNS |
| ping –c 4 172.16.2.124 | -c | ping IP地址四次停止。 |
| 用于显示计算机及操作系统的相关信息 | uname 参数 | -a | 显示全部的信息 |
| -m | 显示当前电脑架构类型 |
| -n | 显示在网络上的主机名称。 |
| -r | 显示操作系统的发行编号，即内核版本号。 |
| -s | 显示操作系统名称，与不加参数的uname的显示结果相同。 |
| -v | 显示操作系统的版本。 |
| 查看和设置日期 | date 参数 | +%c | 显示日期和时间“\*年\*月\*日 \*时\*分\*秒” |
| +%x | 显示日期“\*年\*月\*日” |
| +%D | 显示日期“月/日/年”。 |
| +%Y | 显示完整年份(0000～9999)。 |
| +%y | 显示年份的最后两位数字(00～99)。 |
| +%b | 月份( Jan～Dec，或1月～12月)。 |
| +%B | 月份(January～December，或一月～十二月)。 |
| +%m | 月份(01～12)。 |
| +%d | 日(01～31)。 |
| +%U | 一年中的第几周(00～53)(以Sunday为一周的第一天)。 |
| +%W | 一年中的第几周(00～53)(以Monday为一周的第一天)。 |
| +%w | 一周中的第几天(0～6)。 |
| +%a | 星期几(Sun～Sat，或日、一～六)。 |
| +%A | 星期几(Sunday～Saturday，或星期日～星期六)。 |
| +%j | 一年中的第几天(001～366)。 |
| +%r | 显示12时制的时间，格式为“时:分:秒 AM|PM”，或者“上午|下午 时:分:秒”。 |
| +%X | 显示“\*时\*分\*秒”。 |
| +%s | 从1970年1月1日00: 00:00 UTC到目前为止的秒数。 |
| +%H | 24小时制表示小时数(00～23)。 |
| +%k | 24小时制表示小时数(0～23)。 |
| +%I | 12小时制表示小时数(01～12)。 |
| +%l | 12小时制表示小时数(1～12)。 |
| +%M | 显示分钟(00～59)。 |
| +%p | 显示本地AM或PM或上午、下午。 |
| +%Z | 显示时区。 |
| 设置日期和时间 | date -s “年月日 时分秒”  date -s “20140727 10:22:50”表示将当前日期和时间修改为2014年7月27日10点22分50秒 | 无 | date命令只能修改系统时间 date –s “20140803 11:07:20” |
| 查看硬件时间 | hwclock | 无 | 查看硬件时间 hwclock |
|  | hwclock -w | 无 | 会使用系统时间的值修改BIOS时间 hwclock –w |
| 查看、清除历史记录 | history | 无 | 查看历史记录 history |
| !命令序号 |  | 直接执行输入过的历史命令 |
| history | -c | 清除所有历史命令 |
| 挂载和卸载 | mount /dev/cdrom /media | 无 | 光盘挂载：将/dev下的cdrom设备挂接到/media目录下，以后访问/media就可以看到光盘中的内容。  mount dev/cdrom /media |
| umount /media | 无 | 光盘卸载 |
| fdisk -l | 无 | 查看一下系统是否已经加载U盘 |
| mount /dev/sdb1 /media | 无 | U盘挂载：将/dev下的sdb1设备挂接到/media目录下，以后访问/media就可以看到U盘中的内容。  mount dev/sdb1 /media |
| 建立链接文件 | ln | 无 | 硬链接 ln index.html index1.html |
| ln | -s | 软链接,符号链接 ln –s index.html index1.html |
| 输出字符串在屏幕上 | echo -n “字符串”  在Linux中，字符串的定界符有3种，双引号、单引号、倒引号。  （1）双引号""：之中的字符串，除$、倒引号、英文叹号、斜杠\外，都是普通字符，其中\表示转义。  （2）单引号''：之中的字符串全部是普通字符，原样输出。  （3）倒引号``(键盘直接敲击键盘上的~，不要按shift即可出现)：之中的字符串视为命令。 | 无 | 输出字符串后不换行，默认换行  echo -n “zifuchuan” |
| 往文件中写入文件 | echo "210.22.70.3" >> /etc/resolv.conf  echo “nihao”>> test.txt | 无 | 往/etc/resolv.conf文件中写入210.22.70.3  将“nihao”写入文件test.txt文件中。 |
| 清屏 | clear | 无 | 清屏，快捷键Ctrl+L进行清屏 |
| 查看目录和文件 | pwd | 无 | 用于显示当前所在目录的完整路径 |
| ls | -l | 会显示文件的类型、权限、链接数、文件所有者、文件所有者所在的组、文件大小(以B为单位)、最后修改时间、文件的完整名称(文件名+扩展名)。每一行显示一个文件或者目录 |
| -a | 列出所有文件及子目录，包括以“.”开始的隐藏文件 |
| -h | 以k字节为单位表示文件的大小，通常要配合-l使用，可以写为ll -h。 |
| -s | 根据文件大小排序。 |
| -t | 按文件修改时间排序。 |
| -r | 逆序排列。 |
| ls \*.txt | 无 | 显示所有的txt文件 |
| ls ??.txt | 无 | 显示名字只有2个字符的txt文件：hi.txt |
| ls [abc]\* | 无 | 显示以a、b或c开头的文件或目录 |
| tree | 无 | 以树状形式显示目录结构 |
| 统计目录或文件所占磁盘空间的大小 | du 参数 文件或目录名 | -a | 显示全部目录和其子目录下的每个文件所占的磁盘空间。 |
| -b | 大小用bytes来表示。 |
| -c | 显示完毕后再加上总计(默认值)。 |
| -s | 只显示文件或者目录大小的总和。 |
| -h | 用容易理解的单位表示文件大小，如K、M、G。其计算单位为1024，即1G=1024M。 |
| -H | 与-h基本相同，但计算单位为1000，即1G=1000M。 |
| 查看每个分区的空间占用情况 | df 参数 | -a | 显示所有文件系统的磁盘使用情况，包括0块(block)的文件系统，如/proc文件系统。 |
| -k | 以k字节为单位显示。 |
| -h | 用容易理解的单位表示文件大小，单位显示为K、M、G等。其计算单位为1024，即1G=1024M。 |
| -H | 与-h基本相同，但计算单位为1000，即1G=1000M。 |
| 查看文件类型 | file 文件名 | 无 | 查看文件类型。  具体的文件类型包括：  （1）普通文件：对应于ll命令显示结果中的“-”，还包括文本文件、程序文件、压缩文件等；  （2）目录：对应于ll命令显示结果中的第1位为“d”；  （3）设备文件：对应于ll命令结果中的第1位为“c”或“b”，其中c表示字符文件，如键盘；b表示块设备，如硬盘；  （4）链接文件：对应于ll命令显示结果中的第1位为“l”。  例如： |
| 将文件内容输出至标准输出设备。 | cat 参数 文件名 | -n | 由1开始对所有输出行编号。 |
| -b | 与-n相似，只不过对于空白行不编号。 |
| -s | 当遇到有连续两行以上的空白行，仅显示一行的空白行。 |
| 连接文件 | cat filel file2 > file3 | 无 | file2中的内容连接到file1文件的后面，然后将全部内容传输至文件file3。若文件file3存在，则内容全部被替换；若file3不存在，则新建file3后再传输数据。 |
| cat file4 >> file3 | 无 | 将file4的内容追加到file3之后。 |
| 分屏显示文件内容 | more 文件名 | 无 | more在显示的时候会显示百分比，但显示内容只能向下翻屏，不能向上翻。使用回车可以向下多显示一行，使用空格可以向下多显示一屏，方向键均不起作用。 |
| less 文件名 | 无 | less的显示内容可以上下翻屏，使用上下方向键控制即可，退出时在“:”后输入q即可。 |
| 显示文件前部的内容 | head 参数 文件名 | -c | 字节数：显示文件前面指定个数的字节。 |
| -n | 行数：显示文件前面指定个数的行，若省略n，则显示10行。 |
| -v | 详细输出模式(即显示文件头)。 |
| 显示文件尾部的内容 | tail 参数 文件名 | -c | -c 字节数：显示文件尾部指定个数的字节，单位可选B、KB、MB。 |
| -n | -n 行数：显示文件尾部指定个数的行。 |
| -v | -v：详细输出模式(即显示文件头)。 |
| 统计文件中的行数、单词数和字节数 | wc 参数 文件名 | 无 | wc将统计文件中的行数、单词数和字节数。 |
| -c | 显示字节数。 |
| -m | 显示字符数。 |
| -l | 显示行数。 |
| -L | 显示最长行的字符数。 |
| -w | 显示单词数。 |
| 创建目录 | mkdir 参数 目录名列表 | -p | 递归创建目录，即创建多级目录。如：  mkdir -vp testl/test2/test3 |
| -v | 对已创建的目录输出一个信息。如：  mkdir -v newdir |
| 创建文件 | vi 文件名 | 无 | 新建并编辑文件，但是如果文件中不写入任何数据而且不保存的话，在退出vi后文件内容是不保留的，但可以保存空文件。 |
| touch 文件名 | 无 | touch本来是修改文件访问时间的命令，如果文件不存在，touch会创建一个空文件。 |
| 进入目录 | cd相对路径或绝对路径 | 无 | cd ..：切换至父目录（注意：冒号不是命令）  cd ../home：切换至父目录下的子目录home  cd /var/www/html：绝对路径的写法，进入/var/www/html  cd ～：切换至当前用户的家目录，即home目录  cd ～zhsan：切换至用户zhsan的家目录，即home目录 |
| 进入文件 | vi 文件名 | 无 | 进入文件 |
| 删除目录和文件 | rmdir 参数 目录  备注：只能删除空目录 | -v | 对已经删除的目录给出提示信息。 |
| -p | 递归删除目录。  如：rmdir -p testl/test2/test3 |
| rm 参数 目录或文件名  备注：可以删除多个文件或目录，多个文件或目录用空格或逗号间隔。 | -i | 交互式删除，删除时需要使用者确认是否删除(这是默认值)。 |
| -r | 递归删除，非空目录也可删除。 |
| -f | 不提示确认信息，强制删除。 |
| -v | 显示执行结果信息。 |
| 复制目录或文件 | cp 参数 源目录或文件 目的目录或文件  备注：把源复制到目的地，可以复制文件和目录。 | -r | 若要复制目录，将递归复制该目录下所有的子目录和文件，此时目标必须也是目录。 |
| -u | 如果源比较新，或者未写目的，才会进行复制。 |
| -f | 当目标存在时，强制覆盖。 |
| -i | 和-f选项相反，在覆盖目标之前给出提示，要求用户确认，回答“y”时目标文件将被覆盖。 |
| -p | 除复制源的内容外，还将把其修改时间和访问权限也复制到目的地中。 |
| -a | 常在复制目录时使用，该选项保留链接、文件属性，并递归地复制目录 |
| 对文件或者目录进行移动或更名 | mv 参数 源 目的  备注：若源和目的在同一目录下，表示改名；若不在同一目录下表示移动，移动时也可以同时改名。 | -f | 强制覆盖现有的文件或目录，不进行任何提示。 |
| -i | 覆盖前先行询问用户。 |
| -u | 在移动文件或者目录时，若目标文件或目录已存在，并且源文件日期比目标文件新，则覆盖更新。 |
| -v | 执行时显示详细的信息。 |
| 压缩、解压命令 | gzip 参数 文件名  例如：  gzip \*：将目录下所有文件进行压缩，源文件被删除  gzip -dv \*：解压所有gzip压缩包并查看详细信息，压缩文件被删除 | -c | 将输出写到标准输出上，并保留原有文档。 |
| -d | 将压缩文档解压。 |
| -r | 递归式地查找指定目录并压缩其中的任何文档或是解压缩。 |
| -t | 测试，检查压缩文档是否完整。 |
| -v | 对每一个压缩和解压的文档，显示文档名和压缩比。 |
| -num | 用指定的数字num调整压缩的速度。  -1或-fast：表示最快压缩方法(低压缩比)。  -9或-best：表示最慢压缩方法(高压缩比)，系统默认值为6。 |
| tar 参数 包文件名 目录或文件列表  例如：  tar cvf backup.tar /etc：将etc下所有内容打包成backup.tar  tar xvf backup.tar：将backup.tar解包  tar zcvf hi.tar.gz hi.txt：把hi.txt打包压缩为hi.tar.gz。  要解压文件，可用以下命令：  tar xvfz hi.tar.gz |  | -c：用于创建打包文件。  -v：表示回显模式(Verbose)，显示一系列打进包的文件列表。  -f：指定使用打包的文件，即输出的打包文件名。  -z：使用gzip压缩方式创建和解开打包文件。这是GNU的新版tar的功能，系统需要同时支持gzip功能。  -x：将打包文件展开，即将文件分离出来。 |
| 搜索命令 | find 目标目录 搜索参数 操作参数  find / -name "\*.txt"：表示查找/目录下名字中有.txt的所有文件 | -name | 文件名：根据文件名搜索文件，可以使用通配符。 |
| -user | 用户名：属于某个用户的目录和文件。  常用操作参数： |
| -delete | 找到后删除文件。 |
| locate 参数 字符串  用于定位文件，找到后显示完整路径。执行此命令前先执行updatedb。 | -i | 忽略大小写。 |
| -c | 显示符合条件的文件数量。 |
| grep 参数 字符串 文件名  在文件中搜索文本需要使用grep来实现。  如：  grep "5" goods.txt：命令表示在goods.txt中查找有5的行  grep -c "5" goods.txt：命令表示在goods.txt中查找有5的行共有几行 | 无参 | 显示匹配行 |
| -c | 显示匹配行数 |
| -e | 匹配特殊字符串，如-开头 |
| -i | 忽略大小写 |
| -v | 输出不匹配行 |
| 查询用户 | whoami | 无 | 用于查询当前登录账号的名字 |
| finger 用户名 | 无 | 用于返回指定用户账号的信息。 |
| who | 无 | 用于查询所有在线账户信息 |
| 添加用户 | useradd （adduser） 参数 用户名 | -u | 直接指定一个特定的UID给这个账号，普通用户的UID默认情况下大于499。 |
| -c | 指定用户信息描述，即用户别名。 |
| -d | 指定用户家目录，不使用系统默认值/home/用户名。 |
| -s | 指定用户shell，而不使用默认的/bin/bash。 |
| -g | 指定用户所在组的ID。 |
| -G | 指定用户所在的多个组集合，组织间用逗号间隔。 |
| -mk | 复制其他用户主目录下的文件和配置信息来创建用户。 |
|  | -e expire\_date：账号终止日期。日期的指定格式为MM/DD/YY。 |
|  | -f inactive\_days：账号过期几日后永久停权。当值为0时账号则立刻锁定禁用。而当值为-1时则关闭此功能，预设值为-1。 |
| 添加组名 | groupadd 组名 |  | 添加组名 |
| 设置密码 | passwd 参数 用户名 |  | -d：设置用户口令为空，且允许登录到系统。  -l：锁定用户账号。  -u：解锁用户账号。  -x：密码使用最大时间(天)。  -n：密码使用最小时间(天)。  -w：在距多少天提醒用户修改密码。  -i：密码过期多少天之后被禁用。 |
| 删除用户 | userdel 参数 用户名 | -r | 删除用户的同时删除用户的家目录及其拥有的所有文件。 |
| 修改用户属性 | usermod 参数 用户名 |  | -c：修改用户账号的备注即别名。  -d：修改用户家目录。  -l：修改用户账号名称。  -p：修改用户密码。  -e：修改账号有限期限。  -f：修改账号密码过期多少天之后即关闭该账号。  -g：修改用户所属的群组。  -G：修改用户所属附加群组。  -s：修改用户登录后的shell。  -u：修改用户uid。  -L：锁定用户不得登录。  -U：解除用户锁定。 |
| 切换用户 | su 参数 用户名 |  | 如果没有指定用户，默认切换到root。  -m：不改变环境变量，如切换完成后仍在处在前一用户的家目录中。  -l：使用指定用户的登录环境，例如HOME、SHELL和USER等，并且工作目录也会改变。  -c command:临时变更账号为指定用户，并执行指令(command)后再返回为原来的SHELL环境。 |
| 查看文件权限 | ll：  -rw-r--r-- 1 root root 3671 08-13 23:32 install.log.syslog |  | 文件权限共由10位组成，最左边为第1位，表示文类型，第1位为空，即显示为“-”，表示文件为普通文件，为“d”表示是一个目录；之后的2-10位中每3位为一组。第1组表示文件所有者的权限，第2组表示同组用户的权限，第3组表示组外其他用户对此文件或目录的权限。这3组的权限顺序都是读、写、可执行，表示为r、w、x。 |
| 设定权限：字符设定法 | chmod 参数 权限模式 文件或目录名  如：  chmod ugo+r hi.txt ：将文件hi. txt设为所有人都可读取  chmod ug+w,o-w hi.txt：将文件hi.txt设为所有者与所有者所在组的成员可写入，其他人则不可写入 |  | 参数含义：  -c：若该文件权限确实已经更改，才显示其更改动作信息。  -f：即使该文件权限无法被更改，也不要显示错误信息。  -v：显示权限变更的详细资料。  -R：对目前目录下的所有文档与子目录进行相同的权限变更(即以递归的方式逐个变更)。  权限模式含义：  u：表示用户( user)，即文件或目录的所有者。  g：表示同组( group)用户。  o：表示其他( others)用户。  a：表示所有( all)用户，它是系统默认值。  +：添加某个权限。  -：取消某个权限。  =：赋予指定权限并取消其他所有权限(如果有的话)。  r：表示有读取的权限。  w：表示有写入的权限。  x：表示有执行的权限。 |
| 设定权限：数字设定法 | chmod 参数 权限模式 文件或目录名  chmod 700 hi.txt ：设定hi.txt只有所有者有读、写、执行权限  chmod 755 hi.txt ：设定hi.txt对所有者具有读、写、执行权限，而所在组用户和其他用户只有读、执行权限。 |  | 数字设定法用一系列八进制数表示相应的权限，其中0表示没有权限，1表示拥有可执行权限(1的二进制为001，对应到rwx表示r=0，w=0，x=1)，2表示可写权限(010)，4表示可读权限(100)，然后将其相加。所以数字属性的格式应为3个从0～7的八进制数，其顺序是u、g、o。 |
| 更改文件或目录的所有者 | chown 参数 所有者 文件或目录名  chown命令的常用参数及含义与chmod基本相同。 |  | chown zhsan hi.txt ：将hi.txt的所有者设定为zhsan |
|  | chgrp 参数 组名 文件或目录名 |  | chgrp zhsan hi.txt ：将hi.txt的所属组设定为zhsan |
| rpm软件安装和升级 | rpm -i 安装选项 软件包名  rpm -U 升级选项 软件包名  例如：  rpm -ivh bind-9.3.6-16.P1.el5.i386.rpm ：安装DNS |  | -h：安装时输出hash记号“#”。  --nodeps：不检查依赖性关系，不推荐使用。  --test：只对安装进行测试，并不实际安装。  --percent：以百分比的形式输出安装的进度。  --excludedocs：不安装软件包中的文件。  --includedocs：安装文件。  --replacepkgs：强制重新安装已安装的软件包。  --replacefiles：替换属于其他软件包的文件。  --force：强制安装，忽略软件包及文件的冲突。  --noscripts：不运行预安装和后安装脚本。  --prefix：将软件包安装到指定的路径下。  --ignorearch：不校验软件包的结构。  --ignoreos：不检查软件包运行的操作系统。 |
| rmp软件查询 | rpm -q 查询选项  这里的选项包括两大类：软件包选项和信息选项。  rpm -q bind：查看系统内是否安装bind软件包。  rpm -qa：查看系统安装的所有软件包。  rpm -qf /etc/named.conf：查看/etc/named.conf文件由哪个软件包所安装。  rpm -qi bind：查看已安装的软件包相关信息。  rpm -ql bind：查看软件包所安装的文件位置及名称。 |  | 软件包选项有以下几个：  软件包名：查询系统上是否已经安装这个软件包。  -a：查询所有安装的软件包。  -f：查询文件属于哪一个软件包。  --whatrequires(功能)：查询所有需要(功能)才能正常运行的软件包。  --whatprovides：查询所有能提供某功能的软件包。  --requiredby(软件包)：查询所有安装(软件包)所需要的其他软件包。  -p：查询未安装的软件包。这个软件包可以指定为FTP路径，这时，RPM会下载它的头部信息并查询。  信息选项有以下几个：  -i：显示软件包的信息，包括名称、版本、描述信息。  -R：显示这个软件包所依赖的软件包。  -s：显示这个软件包的状态。状态是指normal、not installed或replaced。  --provides：显示这个软件包所提供的功能。  -l：列出这个软件包内所包含的文件。  -d：列出被标注为文档的文件(如man手册、info信息、README等)。  -c：列出被标注为配置文件的文件。这些文件是需要在安装完毕后加以定制的，如sendmail.cf、passwd、inittab等。 |
| rmp软件校验 | rpm -V 校验选项 软件包名  校验会把软件包所安装的软件与原先数据库中存储的软件信息做一个比较。RPM可以验证文件的大小、MD5校验码、文件权限、类型、所有者等信息。  rpm -Vp bind-9.3.6-16.P1.el5.i386.rpm ：检测所下载的RPM包是否被破坏。  rpm -Va：验证所有的软件包或者是查看误删除了哪些文件。 |  | 常用校验选项及含义如下：  -f：校验文件所属软件包属性是否变化。  -p：校验已安装的软件包。  -a：校验所有已安装的软件包。  -g：校验所有属于某个组的软件包。  校验输出的格式是一个包括8个字符的字符串，这8个字符中每个字符都代表同数据库中信息所比较的结果。如果结果是“.”，就代表没有问题。相应字符的意义如下所示:  S：文件大小。  M：属性(包括读写权限和文件类型)。  5：MD5校验。  L：符号链接。  D：设备文件。  U：用户名。  G：组名。  T：文件修改时间。  ?：不可读文件。 |
| 删除软件包 | rpm -e 软件名  说明：在卸载软件包时使用软件名称bind或bind-9.3.6-16.P1.el5，而不是软件包文件名称bind-9.3.6-16.P1.el5.i386.rpm。 |  | rpm -e bind：卸载软件bind。 |
| 使用yum管理rmp软件包 | yum 参数 指令 软件包名  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。  使用yum安装软件的步骤：   1. 配置/etc/yum.repos.d/rhel-debuginfo.repo   baseurl=file:///media/RHEL5.6\_i386/Server  enabled=1   1. 使用yum -y install 软件包名安装软件。   例如：yum -y install gcc-c++  yum list gcc-c++：查询gcc-c++是否安装成功 |  | 常用参数如下：  -y：自动应答yes。  -e：静默执行 。  -t：忽略错误。  -R 分钟数：设置等待时间。  --nogpgcheck：忽略GPG验证。  常用的指令如下：  install：安装包。  update：更新包。  search 关键词：搜索包。  remove：卸载包。  reinstall：重新安装包。  list：可安装和可更新的RPM包。  check-update：检查可更新的包。  clearn：清除全部。  deplist：列出包的依赖。  clean packages：清除临时包文件（/var/cache/yum 下文件）。  list installed：已安装的包。  list extras：已安装且不在资源库的包。  info：可安装和可更新的RPM包信息。  info installed：已安装包的信息(-qa 参数相似)。  resolvedep：指定依赖。 |
| 网络配置文件 | /etc/services文件  /etc/services文件中条目的格式为：  服务名 端口/协议 别名 注释 | 无 | 该文件中存放着端口信息。在Internet上，各主机间通过TCP/IP协议发送和接收数据报，各个数据报根据其目的主机的IP地址来进行互联网络中的路由选择。但是当多个应用程序在同一主机上运行，目的主机必须通过一种方法来确定应该把接收到的数据报传送给众多同时运行的程序中的哪一个，这就需要用到端口 |
| /etc/hosts文件  每行包括一个IP地址、一个完整域名(可选)和一个主机名 | 无 | 该文件是保存主机名和IP地址映射的静态文件，用于本地的名称解析，是域名系统DNS的前身，如果网络中没有DNS而且网络的规模不大，可采用这种解析方式，将所有主机的名称与IP地址的映射信息写入到该文件，然后将该文件发给网络中所有的主机。 |
| /etc/host.conf文件  该文件用来指定如何解析主机名，文件内容为order hosts,bind。 | 无 | order用于设置主机名称解析的方法及顺序，hosts表示使用/etc/hosts文件进行解析，bind表示使用DNS解析。上述内容表示先根据/etc/hosts文件的内容对域名进行解析；若hosts解析失败，再按DNS解析。如果不希望先从本地进行解析，可以将hosts与bind的位置进行颠倒更换。 |
| 网络命令 | service 服务名 start 丨 stop 丨 restart 丨 status | reload  例如：service network status：查看网络连接服务的状态可以使用命令 | 无 | service命令用于调用/sbin/service脚本，以实现对网络服务的控制。  status：查看服务状态。  start：启动服务。  stop：停止服务。  restart：重新启动服务。  reload：重新载入服务。 |
| route：可以查看系统的当前路由信息。 | 无 | 路由显示的信息中各项含义如下：  Destination：该路由项指向的目标IP，可以是一个子网或单个主机的地址。  Gateway：该路由项的网关地址。  Genmask：该路由项的子网掩码。  Flags：路由标志，U表示是活动的；H表示目标是单个主机；G表示使用网关；R表示对动态路由进行复位；D表示路由项是动态安装的,M表示动态修改路由；!表示拒绝路由。  Metric：路由的开销。  Ref：依赖于该路由的其他路由数目。  Use：该路由项被使用的次数。  Iface：该路由项发送数据包使用的网络接口。 |
| route add 丨 del 目标子网掩码 网关 dev 网络接口  例如：route add default gw 192. 168. 125. 4：  将默认网关设置为192. 168. 125.4 | 无 | 其中add用于添加路由项，同时还需要使用-net和-host选项，指明要到达的目标是一个子网还是单个主机；del用于删除指定的路由项。 |
| 网络诊断工具 | ping 参数 目标主机  ping app：确定主机app是否可达；  ping [www.google.com](http://www.google.com)：确定网络性能 | 无 | -c：指定ping命令发出的ICMP数据包数量，如果不指定将会不停地发送，直到使用Ctrl+C组合键终止命令。  -i：指定ping命令发出每个ICMP数据包的时间间隔，默认为1s。  -s：指定ping命令发出每个ICMP数据包的大小，默认为64字节，最大为65507字节。  -t：设置TTL(Time to Live，生存期)。 |
| 路由跟踪命令 | traceroute命令  traceroute Vermeer：显示本地主机到达目的主机的路径 |  | 该命令用来跟踪路由器，即跟踪本地与远程主机之间UDP数据报，并根据接收到的回应信息判断网络故障可能的位置。 |
| 查看网络信息命令 | netstat命令  该命令用来查看各种与网络相关的信息，包括网络的连接情况、接口的统计信息、查看路由表以及端口的监听情况等。  例如：  netstat -atu：只查看UDP和TCP服务清单，不考虑连接状态。  netstat -a：显示所有监听和非监听的套接字。 |  | 网络的连接(TCP协议)常见的有以下几种情况：ESTABLISHED：已经建立连接。  SYN SENT：尝试发起连接。  SYN RECV：接受发起的连接。  TIME WAIT：等待结束。  LISTEN：监听。  netstat命令常用的选项及其描述如下：  -a：显示所有监听和非监听的套接字。  -j：显示接口的统计信息。  -n：显示内核路由表。  -s：为所有协议显示统计信息。 |
| OpenSSH的安装与配置 | rpm qa | grep ssh：查看RHEL5中是否安装了OpenSSH  openssh-4.3p2-26. e15. i386. rpm  openssh-askpass-4.3p2-26. e15. i386. rpm  openssh-clients-4.3p2-26. e15. i386. rpm  openssh-server-4.3p2-26. e15. i386. rpm  OpenSSH服务器的配置：  主要是通过/etc/ssh/sshd\_config来完成，部分代码如下（13、14、15表示行号）：  13 #Port 22  14 #Protocol2,1  15 Protocol 2  查看OpenSSH服务状态：service sshd status  启动OpenSSH服务状态：service sshd start |  | 网络管理员对服务器的配置、维护与更新数据往往都是采用远程管理的方式来进行，其中较常用的工具与服务有telnet、ftp、rlogin、rsh和rcp等，但这些工具都是采用明文方式进行传输，信息在传输的过程被截获后易被破解，基于安全的考虑，芬兰一家公司开发了SSH(Secure shell)软件，它对网络中传输的数据进行加密，保障了系统的安全性，但SSH现在已经成为商业软件，故在RHEL5中采用了由OpenBSD维护的OpenSSH来取代早期的SSH，它分为服务端和客户端产品。 |
| 使用OpenSSH远程管理Linux服务器 | 服务器本机验证：[zhsan@localhost ~]$ ssh 192.168.1.211  使用命令exit与logout均可断开与远程服务器的连接。 |  |  |
| 利用proc文件系统查看系统内核信息 | /proc不是一个真实目录，不占用任何磁盘空间，用户可以通过目录/proc下的文件得到系统的内核与运行信息，如下所示。  /proc/cpuinfo文件：处理器详细信息。  /proc/modules文件：系统当前加载模块信息。  /proc/version文件：当前系统版本信息。  /proc/partitions文件：系统分区信息。 |  |  |
| who命令 | who 参数  用于查看当前系统登录用户、系统中有哪些登录者、使用的终端、登录时间、CPU使用率、动作等。 |  | 常用参数及含义如下：  -H：显示各栏位的标题列。  -a：显示当前所有登录用户信息。  -u：显示闲置时间，若该用户在前一分钟之内有进行任何动作，将标示成“.”号，如果该用户已超过24小时没有任何动作，则标示出“old”字符串。  -q：只显示登入系统的账号名称和总人数。  -w：显示用户的信息状态栏。  --help：在线帮助。  --version：显示版本信息。 |
| w命令 | w 参数 用户名  该命令用于查看系统登录信息。w命令也用于显示登录到系统的用户信息，但是与who不同的是，w命令功能更加强大，它不但可以显示有谁登录到系统，还可以显示出这些用户当前正在进行的工作，并且统计数据相对who命令来说更加详细和科学，可以认为w命令就是who命令的一个增强版。  [root@localhost ~]# w  显示信息如下：  12:05:35 up 38 min, 3 users, load average: 0.00, 0.00, 0.00  USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT  root tty1 - Sun15 2days 0.02s 0.02s -bash  root :0 - Sun15 ?xdm? 17.58s 0.14s /usr/bin/gnom  root pts/0 :0.0 Sun15 0.00s 0.10s 0.03s w  在显示信息中，第一行共有4个字段，分别表示如下：  系统当前时间：也就是执行w命令时的时间。  系统启动后运行的时间：up38min表示从开机到当前共运行了38分钟。  当前登录系统的用户总数：3 user表示当前有若干用户登录此系统。在Linux中，允许同一用户账号通过终端反复登录，故有时会见到重复的账号名。  系统平均负载：load average:0.00，0.00，0.00中的数据，表示系统在过去1、5、10 min内的平均负载程度。数值越接近0.00，表示系统负载越轻，性能更佳。  第二行共有8个字段，分别表示如下：  USER：登录的用户账号名。  TTY：用户登录的终端代号。登录的形式不同，终端代号也不相同。  FROM：从哪个终端登录。如果是本地登录，则此字段为“-”；若是远程登录，则显示远程主机的IP地址或主机名；如果是通过本地图形窗口登录，则显示“:0.0”。  LOGIN@：就是LOGIN At的意思，表示该用户登录系统的时间。  IDLE：空闲计时器，表示用户闲置的时间，一旦用户执行任何操作，该计时器便重置计时。  JCPU：终端相关进程计时器。表示该终端所有的相关进程执行时所消耗的CPU时间。每当进程结束就停止计时，开始新的进程时则会重新计时。  PCPU：CPU执行程序消耗的时间。  WHAT：显示用户正在执行的程序的名称。 |  | 常用参数及含义如下：  -f：开启或关闭显示用户从何处登入系统。  -h：不显示各栏位的标题信息列。  -l：使用详细格式列表，此为预设值。  -s：使用简洁格式列表，不显示用户登入时间，终端机阶段作业和程序所耗费的CPU时间。  -u：忽略执行程序的名称，以及该程序耗费CPU时间的信息。 |
| last命令 | last 参数 账号名称 终端编号  此命令用于列出目前与过去成功登入系统的用户相关信息。 |  | 常用参数及含义如下：  -a：把从何处登入系统的主机名称或IP地址显示在最后一行。  -d：将IP地址转换成主机名称。  -f：指定记录文档。  -n：配置列出名单的显示列数。  -R：不显示登入系统的主机名称或IP地址。  -x：显示系统关机，重新开机，连同执行等级的改变等信息。 |
| lastb命令 | 显示用户不成功登录信息，命令格式和参数同last。此命令需要有root权限。 |  | [root@localhost ~]# lastb  zhsan tty1 Wed Jan 15 12:23 - 12:23 (00:00) |
| lastlog命令 | lastlog -u 用户名 |  | 此命令用于显示用户最后登录时间。如不使用参数，则显示所有用户的最后登录时间。 |
| ps命令 | ps 参数  用于查看系统中执行的进程。  ps aux：查看系统当前进程。  显示信息如下：  USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND  root 1 0.0 0.1 2076 640 ? Ss 02:06 0:01 init [5]  root 2 0.0 0.0 0 0 ? S< 02:06 0:00 [migration/0]  root79050.0 0.2 7008 1056 ? Ss 02:09 0:00 /usr/sbin/sshd  各列含义如下：  USER：进程拥有者。  PID：进程号。  %CPU：占用的CPU使用率。  %MEM：占用内存使用率。  VSZ：进程全部调用至内存需要占用的虚拟内存大小。  RSS：真实占用的内存大小。  TTY：从哪个终端登录。  STAT：该进程的状态。  START：进程开始时间。  TIME：执行的时间。  COMMAND：所执行的指令。 |  | 常用参数及含义如下：  -e：显示所有进程和环境变量。  -f：全格式显示。  -h：不显示标题。  -l：长格式显示。  -w：宽格式输出。  a：显示终端上的所有进程，包括其他用户的进程。  r：只显示正在运行的进程。  x：显示没有控制终端的进程。  u：以用户为主的格式来显示进程。 |
| pstree命令 | pstree 参数 用户名 |  | 用于查看进程树。  常用参数及含义如下：  -a：显示每个程序的完整指令，包含路径，参数或是常驻服务的标识。  -c：不使用精简标识法。  -G：使用VT100终端机的列绘图字符。  -h：列出树状图时，特别标明现在执行的程序。  -H 程序识别码：此参数的效果和指定“-h”参数类似，但特别标明指定的程序。  -l：采用长列格式显示树状图。  -n：用程序识别码排序。默认值是以程序名称来排序。  -p：显示程序识别码。  -u：显示用户名称。  -U：使用UTF-8列绘图字符。  -V：显示版本信息。 |
| kill命令 | kill 参数 进程号  用于发送特定的信号给指定的进程或进程组。 |  | 常用参数及含义如下：  -2：等价于Ctrl+ C。  -9：发送强制终止信号。  -15：发送正常终止信号，与-9不一样。 |
| free命令 | free 参数  用于查看内存使用情况。  例如：free -m -s5：每隔5s报告内存使用情况，以MB为单位，也可以写为  free -ms5 或free -ms 5。  显示信息如下：  total used free shared buffers cached  465 439 25 0 63 223  -/+buffers/cache: 151 313  Swap: 1027 0 1027  Total used free shared buffers cached  465 439 25 0 63 223  -/+ buffers/cache: 151 313  Swap: 1027 0 1027  第1行信息中，共有6个字段，含义如下：  total：内存总数。  used：已经使用的内存数。  free：空闲的内存数。  shared：该参数当前已经废弃不用，其值总为0。  buffers：缓冲内存。  cached：页面缓存。 |  | 常用参数及含义如下：  -b：以字节为单位显示内存使用情况。  -k：以KB为单位显示内存使用情况。  -m：以MB为单位显示内存使用情况。  -g：以GB为单位显示内存使用情况。  -s：每隔多少秒显示一次内存使用情况。  -t：显示内存和SWAP的总和。 |
| top命令 | top 参数  用于动态监控系统资源，包括内存、交换分区和CPU的使用率等，并且以周期性时间更新内容。  [root@ localhost ~] # top  显示结果如下：  top-04:22:49 up 2:16, 1 user,load average: 0. 00, 0. 00, 0.00  Tasks: 93 total,1 running,92 sleeping, 0 stopped, 0 zombie  Cpu(s):0.9%us,2.4%sy,0.1%ni,95.3%id,0.9%wa,0.3%hi,0.1%si, 0.1% st  Mem: 476160k total, 426052k used,50108k free,47984k buffers  Swap:048568k total, Ok used, 1048568k free,280976k cached  PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND  10330root 15 0 2204 920 716 R 3.1 0.2 0:00.19 top  7816 root 18 0 22984 1304 568 S 1.5 0.3 0:00 92pcscd  1root 15 0 2076 640 548 S 0.0 0.1 0:01.39 init  2root RT -5 0 0 0 S 0.0 0.0 0:00.00 migration/0  top命令显示的项目很多，默认是每5s更新一次，当然这是可以设置的。第1行的含义依次为当前时间、系统启动时间、当前系统登录用户数目、平均负载(1min、5min、15min前到现在)。第2行为进程情况，依次为进程总数、运行进程数、休眠进程数、终止进程数、僵尸进程数。第3行为CPU状态，依次为用户占用、系统占用、优先进程占用、闲置进程占用。第4行为内存状态，依次为平均可用内存、已用内存、空闲内存、共享内存、缓存使用内存。第5行为交换状态，依次为平均可用交换容量、已用容量、闲置容量、高速缓存容量。然后下面是和ps相仿的各进程情况列表。 |  | 常用参数及含义如下：  -d：指定更新的间隔，以秒计算。  -q：没有任何延迟的更新。如果调用程序者拥有超级用户权限，则top命令将会以最高的优先序执行。  -c：显示进程完整的路径与名称。  -S：累积模式，将已完成或消失的子行程的CPU时间累积起来。  -s：安全模式，去除交互命令所带来的潜在危险。  -i：不显示任何闲置(Idle)或无用(Zombie)的进程。  -n：显示更新的次数，完成后将会退出top。  top显示内容中可以使用交互式命令，熟练掌握这些交互命令的应用对系统管理是非常必要的，常用交互命令有以下几个：  空格：立即刷新显示。  h或者?：显示帮助画面，给出一些简短的命令总结说明。  k：终止一个进程系统，将提示用户输入需要终止的进程PID，以及需要发送给该进程什么样的信号。一般的终止进程可以使用15信号；如果不能正常结束那就使用信号9强制结束该进程。默认值是信号15。在安全模式中此命令被屏蔽。  q：退出程序。  s：改变两次刷新之间的延迟时间。系统将提示用户输入新的时间，单位为s。如果有小数，就换算成ms。输入0值则系统将不断刷新。需要注意，设置太小的时间很可能会引起不断刷新，从而根本来不及看清显示的情况，而且系统负载也会大大增加。  m：切换显示内存信息。  t：切换显示进程和CPU状态信息。  c：切换显示命令名称和完整命令行。  M：根据驻留内存大小进行排序。  P：根据CPU使用百分比大小进行排序。  T：根据时间/累计时间进行排序。  A：按进程生命顺序进行排序，最新进程显示在最前面。 |
| sar命令 | 用于监测CPU、内存、硬盘、网络状态。此命令需要先安装并启动服务，如下命令：   1. rpm -ivh sysstat\*.rpm 2. service sysstat start |  |  |
| sar监测CPU  sar -u 间隔时间 采样次数  在所有的显示中，应主要注意%iowait和%idle。若%iowait的值过高，表示硬盘存在I/O瓶颈；若%idle值高，表示CPU较空闲，如果%idle值高但系统响应慢时，有可能是CPU等待分配内存，此时应加大内存容量。%idle值如果持续低于10，那么系统的CPU处理能力相对较低，表明系统中最需要解决的资源是CPU。 |  | sar -u 1 3  显示结果如下：  Linux 2.6.18-238.el5 (localhost.localdomain) 2014年01月15日  14时50分16秒  CPU %user %nice %system %iowait %steal %idle  14时50分17秒 all 0.99 0.00 1.98 0.00 0.00 97.03  14时50分18秒 all 4.00 0.00 2.00 0.00 0.00 94.00  14时50分19秒 all 0.00 0.00 0.00 4.00 0.00 96.00  Average: all 1.66 0.00 1.33 1.33 0.00 95.68  各列含义如下：  %user：用户进程使用的CPU时间百分比。  %nice：正常运行的进程使用的CPU时间百分比。  %system：内核进程使用的CPU时间百分比。  %iowait：为I/O等待所占用的CPU时间百分比。  %steal：为虚拟CPU等待物理CPU所占用的时间百分比。  %idle：空闲状态百分比。  Average：平均值统计。 |
| sar监测内存  sar -r 时间间隔 采样次数  各列含义如下：  kbmemfree：空闲物理内存大小。  kbmemused：已使用物理内存大小。  %memused：已使用内存占内存总容量的百分比。  kbbuffers：缓存容量。  kbcached：页面缓存容量。  kbswpfree：空闲的虚拟内存大小。  kbswpused：已使用虚拟内存大小。  %swpused：已使用虚拟内存百分比。  kbswpcad：已缓存的虚拟内存容量。 |  | sar -r 1 3  显示结果如下：  Linux 2.6.18-238.el5 (localhost.localdomain) 2014年01月15日15时03分27秒  kbmemfree kbmemused %memused kbbuffers kbcached kbswpfree kbswpused %swpused kbswpcad  15时03分28秒 31520 483388 93.88 42572 262796 1020088 0 0.00 0  15时03分29秒 31528 483380 93.88 42580 262796 1020088 0 0.00 0  15时03分30秒 31528 483380 93.88 42580 262796 1020088 0 0.00 0  Average: 31525 483383 93.88 42577 262796 1020088 0 0.00 0 |
| sar监测磁盘  sar -d 间隔时间 采样次数  sar -d 1 1  各列含义如下：  DEV：磁盘设备名。  tps：每秒磁盘的I/O流量。  rd\_sec/s：每秒从设备读取的扇区数。  wr\_sec/s：每秒写入设备的扇区数。  avgrq-sz：平均每次设备I/O的数据大小。  avgqu-sz：平均队列长度。  await：平均每次I/O操作的等待时间，单位为毫秒。  svctm：平均每次I/O的操作时间。  %util：I/O操作在每秒占用的时间百分比。  通常，svctm值小于await时间，如果await远远大于svctm，说明磁盘速度成了系统瓶颈，%util值如果接近100%说明也磁盘性能不足。 |  | 显示结果如下：  Linux 2.6.18-238.el5 (localhost.localdomain) 2014年01月15日  15时09分43秒 DEV tps rd\_sec/s wr\_sec/s avgrq-sz avgqu-sz await svctm %util  15时09分44秒 dev8-0 106.93 3152.48 815.84 37.11 0.41 5.59 2.39 25.54  15时09分44秒 dev8-1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  15时09分44秒 dev8-2 106.93 3152.48 815.84 37.11 0.41 5.59 2.39 25.54  15时09分44秒 dev8-3 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  15时09分44秒 dev8-4 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  15时09分44秒 dev8-5 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  15时09分44秒 dev8-6 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  15时09分44秒 dev22-0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00  Average: DEV tps rd\_sec/s wr\_sec/s avgrq-sz avgqu-sz await svctm %util  Average:dev8-0 106.93 3152.48 815.84 37.11 0.41 5.59 2.39 25.54  Average:dev8-1 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 |
| sar监测网络  sar -n 统计信息 时间间隔 采样次数  sar -n DEV 1 1  显示结果各列含义如下：  IFACE：网络接口设备。  rxpck/s：每秒接收的数据包大小。  txpck/s：每秒接收的压缩数据包数量。  txcpm/s：每秒发送的压缩数据包数量。  rxmcst/s：每秒接收的多播数据包数量。 |  | 统计信息内容如下：  DEV：统计网络接口。  EDEV：统计网络的错误信息。  NFS：统计NFS客户端的网络信息。  NFSD：统计NFS服务器的网络信息。  SOCK：统计套接字信息。  ALL：显示以上所有信息。 |
| iostat命令 | 此命令可以监测CPU、磁盘。  监测CPU：iostat -c 间隔时间 采样次数  监测硬盘：iostat -d 设备名 时间间隔 采样次数  不加参数时，CPU和磁盘同时检测。  如iostat -d /dev/sda可以单独检测磁盘。 |  | Iostat  显示结果各列含义如下:  avg-cpu是CPU的平均负载数值。  Device：磁盘设备名。  tps：每秒磁盘的I/O流量。  Blk\_read/s：每秒读取的数据块数。  Blk\_wrtn/s：每秒写入的数据块数。  BLK\_read：总共读取的数据块数。  Blk\_wrtn：总共写入的数据块数。 |
| DNS服务器安装配置 | 安装前的配置：关闭防火墙：service iptables stop |  |  |
| 安装软件：  **（1）bind-9.3.6-16.P1.el5.i386.rpm：**默认未安装，bind主程序包，但不生成主配置文件。  **（2）bind-libs-9.3.6-16.P1.el5**：默认已安装，提供库文件。  **（3）bind-utils-9.3.6-16.P1.el5**：默认已安装，DNS客户端工具包，包含测试命令：dig、host、nslookup、nsupdate。  **（4）bind-chroot-9.3.6-16.P1.el5.i386.rpm**：默认未安装，用于建立chroot目录。  **（5）**  **caching-nameserver-9.3.6-16.P1.el5.i386.rpm**：默认未安装，用于配置缓存域名服务器。 |  |  |
| 安装后配置：   1. 修改主配置文件：**/var/named/chroot/etc/named.conf** |  | listen-on port 53 { any; };  …  allow-query { any; };  allow-query-cache { any; };  …  match-clients { any; };  match-destinations { any; }; |
| （2）配置解析记录清单（该文件用于记录正向域和反向域） |  | 假设域名为test.com，对应的网段为192.168.1：  zone "test.com" IN {  type master;  file "test.com.zone";  allow-update { none; };  };  zone "1.168.192.in-addr.arpa" IN {  type master;  file "named.192.168.1";  allow-update { none; };  }; |
| （3）配置解析记录文件：两步骤  **一、创建并配置正向解析记录文件**  **cd /var/named/chroot/var/named**  **cp -p named.zero test.com.zone** |  | $TTL 86400  @ IN SOA localhost. root.localhost. (  42 ; serial (d. adams)  3H ; refresh  15M ; retry  1W ; expiry  1D ) ; minimum  IN NS localhost.  dns IN A 192.168.1.211  www IN A 192.168.1.212  mail IN A 192.168.1.213  说明：A表示主机(地址)，IP地址之后不要再加“.”。 |
|  | **二、创建并配置反向解析记录文件**  **cp -p named.local named.192.168.1**  **vi named.192.168.1** |  | 修改内容为下述代码：  $TTL 86400  @ IN SOA localhost. root.localhost. (  1997022700 ; Serial  28800 ; Refresh  14400 ; Retry  3600000 ; Expire  86400 ) ; Minimum  IN NS localhost.  211 IN PTR dns.test.com.  212 IN PTR www.test.com.  213 IN PTR mail.test.com.  说明：PTR表示指针；域名后的“.”不要省略，若不加点，则“www.test.com”会被当成www.test.com.test.com.来解析。 |
| DNS服务启动关闭 | **service named start丨stop丨restart丨status丨reload** |  |  |
| DNS检测 | **使用BIND工具检测**  **1.named-checkconf工具：**用于检查主配置文件/var/named/chroot/etc/named.conf是否存在错误。 |  | **cd /var/named/chroot/etc**  **named-checkconf named.conf** |
| **2.named-checkzone工具：**用于检测区域文件，如果有错误将给予提示，并报告错误。 |  | **cd /var/named/chroot/var/named**  **named-checkzone test.com test.com.zone** |
| **使用host进行检测**  **host -l test.com** |  | 可以显示test.com下所有的主机解析信息 |
| nslookup进行检测  **(1)非交互式：**非交互式通常用于查询单个域名的解析结果，其格式如下：  **nslookup 需查询的域名 | IP地址** |  |  |
| **(2)交互式：**交互式通常用于查询多个域名解析的情况，其命令格式为：**nslookup**  在进行查询时，还可以输入**nslookup IP地址**进行反向解析，也可以在交互式查询方式下的提示符“>”后直接输入域名或IP查询解析情况 |  | > **set type=soa** #测试授权SOA资源记录  > **test.com**  **。。**  > **set type=ns** #测试名称服务器NS资源记录  > **test.com** |
| DNS客户端设置 | 编辑配置文件：vi /etc/resolv.conf |  | 使用nameserver选项来指定DNS服务器的IP地址 |

**2、Ubuntu的程序接口测试**

**2.1、编辑创建于root目录下的文件mian.c**

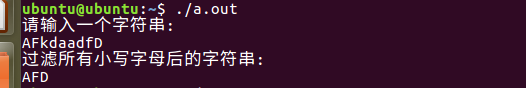


**2.2、调用ubuntu编译器gcc编译mian.c文件**



**2.3、执行程序**

**2.4、编程接口(API)初步测试解释**

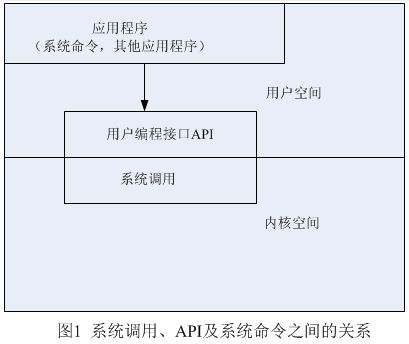


C语言printf、scanf函数通过调用对应的存放在动态链接库里的API函数，访问内核空间实现在默认I/O设备上进行输出与输入。

**2.5、操作系统的应用编程接口(API)**

**A、**应用程序接口（英语：Application Programming Interface，简称：API），又称为应用编程接口，就是软件系统不同组成部分衔接的约定。由于近年来软件的规模日益庞大，常常需要把复杂的系统划分成小的组成部分，编程接口的设计十分重要。程序设计的实践中，编程接口的设计首先要使软件系统的职责得到合理划分。良好的接口设计可以降低系统各部分的相互依赖，提高组成单元的内聚性，降低组成单元间的耦合程度，从而提高系统的维护性和扩展性。

**B、**应用程序接口系统调用关系图解：



**3、Ubuntu的图形界面**

**3.1、Unity 用户界面小史**

A、出生于特殊时期

Unity 用户界面的最早出现，是在 2010 年的 5 月份，当时 Ubuntu 的母公司 Canonical 创始人 Mark Shuttleworth 发表博文《[Unity, and Ubuntu Light](http://www.markshuttleworth.com/archives/383)》首次介绍了两个新概念：Unity 界面，和 Ubuntu Light 操作系统。

2010 年中期正值上网本大行其道的年代，Linux 借助上网本进军桌面市场的思潮也正在盛行。在当时的双启动上网本概念中，用户可以在数秒内快速进入一个简化的 Linux 系统处理一些浏览网络、收取邮件的简单任务，而当需要完成大型任务时则可以启动 Windows 系统。为了迎合这一趋势，Canonical 准备推出名为 Ubuntu Light 的概念系统，而 Unity 就是应用于 Ubuntu Light 系统的用户界面。最初的 Unity 界面概念图是这样的：



在这张概念图中已经可以见到今天 Unity 界面的基本结构和很多界面元素。所不同的是，在当时的概念中，左侧快速启动栏上的项目主要是一些浏览器的快捷方式（图中有 Gmail、Facebook、Youtube、Flickr 等），而界面的主体部分则基本完全是一个最大化的浏览器（图中正在运行一个地图网页）。总之，在最初的构想中，Unity 是一个彻底为快速完成网络浏览任务而打造的界面。

B、成长于特殊时期

此时，Canonical 方面也计划在 Ubuntu 10.10 的上网本专用版（Netbook Remix）中应用这一界面。当时距离 Ubuntu 10.10 的发布还有将近六个月的时间，Mark Shuttleworth [认为](http://www.markshuttleworth.com/archives/383)，在这六个月的时间里实现以下三者，即可成功地用 Unity 界面替换掉当时被认为体验不佳的上网本界面。

* 应用管理：在 Ubuntu Light 中，用户基本只需要一个浏览器，Unity 的界面也确实基本都留给了浏览器。而上网本专用版作为一个完整的桌面版本，则需要满足用户安装、卸载和管理软件的需求。
* 文件管理：Ubuntu Light 作为一个轻量级的系统，只需为用户提供浏览器界面既可。而上网本专用版则需具有文件管理的能力。
* 全局菜单：在[稍早的博文](http://www.markshuttleworth.com/archives/359)中，Mark Shuttleworth 宣布，为了尽量节省上网本小屏幕上的宝贵空间，Ubuntu 10.10 的上网本专用版系统将使用全局菜单替换掉传统菜单。

为了实现上面三项功能，Ubuntu 的开发者们为 Unity 界面添加了一个名为 Dash 的特性。用户可以在 Dash 中完成应用管理、文件管理等任务。而全局菜单也被逐步地实现出来。就这样，Ubuntu 10.10 的上网本专用版成为了第一个使用 Unity 界面的正式版本。



而也正由于这样的成长历程，导致了 Unity 始终无法摆脱“上网本界面”的影子。很多为上网本的小屏幕进行优化而来的特点，到后来都成为了累赘和缺陷。

C、转正于特殊时期

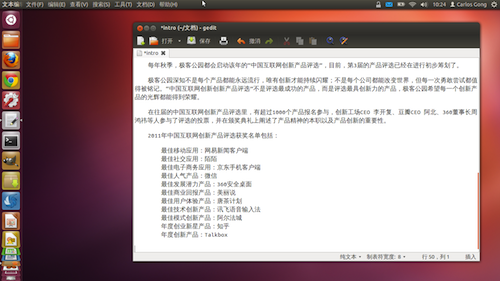
按照惯例，每个新版 Ubuntu 系统发布之后，Canonical 都要举行一次开发者大会。2010 年 10 月底，在 Ubuntu 10.10 版本发布之后的那次开发者大会上，Mark Shuttleworth [宣布](http://www.omgubuntu.co.uk/2010/10/ubuntu-11-04-unity-default-desktop/)下一个版本，即 Ubuntu 11.04，将把上网本专用版和普通桌面版合二为一，统一后的 Ubuntu 11.04 版本将全部使用 Unity 界面，取代掉传统上沿用多年的 Gnome 2 桌面环境的界面。

2010 年末 2011 年初对于 Linux 的主流桌面环境来说也是一个比较特殊的时期。Ubuntu 传统上使用的 Gnome 2 桌面环境已经持续小修小补了 9 年，而新版 Gnome 3 在经历了多年的难产和跳票之后处于即将发布的阶段。Gnome 3 的默认用户界面是 Gnome-shell，从当时已有的情况来看，这一变革过大的界面并不一定符合用户的口味，并且其硬件支持状况十分堪忧。而 Canonical 此时也正意欲借 Unity 界面来打造自己独特的用户体验，在这种特殊的情况下，Unity 用户界面就从一个上网本界面转正为 Ubuntu 的标准界面。

于是，6 个月后的 2011 年 4 月初，以 Gnome-shell 为用户界面的 Gnome 3 桌面环境发布，十几天后，Ubuntu 11.04 发布，以 Unity 为默认用户界面。并且不兼容 Gnome 3。

**3.2、Ubuntu 12.04 中的 Unity 界面**

默认情况下，Unity 界面的结构如下：



A、Dash 和 Lens

Dash 是 Unity 的应用管理和文件管理界面。点击左上角的 Ubuntu 标志或者在桌面按下 Super 键（Windows 键/Command 键）即可进入，Lens 是 Dash 界面下方的一行图标，点击每个图标都可以切换到对应标签页，每个标签页致力于满足用户的一类特定需求。Dash 界面的基本结构如下：

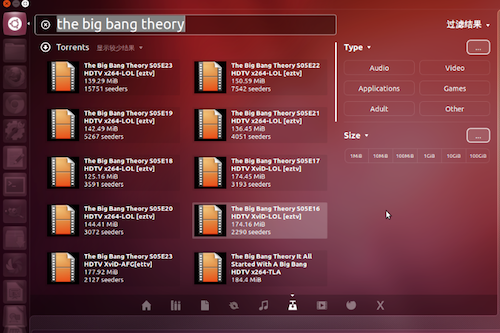


Dash 在首页上显示最近使用的应用、打开的文件和下载的内容，而其后的各个 Lens 则分别满足各项特定的需求，默认的 Lens 有软件（应用程序管理）、文件（文件管理）、音乐（音乐管理）和视频（视频管理）。每个 Lens 都可以对相关的内容进行搜索、展示和分类过滤。

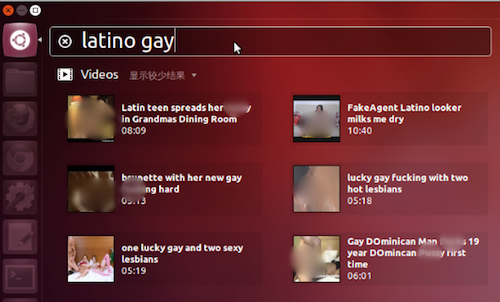
此外，用户还可以自行添加 Lens 来满足特定的需求。例如社交网络 Lens 可以快速地搜索、显示和过滤社交网络信息：



海盗湾的 Lens，找种子什么的变得很简单：



当然，作为一种[刚性需求](http://www.geekpark.net/read/view/155894)，色情内容 Lens 肯定也是有的：



并且和[我们先前介绍过的](http://www.geekpark.net/read/view/156091) Snatch.ly 一样，也可以按照自己的偏好来过滤结果：



Dash 的这种界面布局和 Lens 的可扩展设计看起来是比较理想的，然而，在真正使用的时候却并不一定能为用户提高效率。以最为人诟病的应用管理 Lens 为例：

在采用 Unity 界面之前，Ubuntu 采用的是 Gnome 2 的经典用户界面，在这个界面上，寻找特定的应用程序是一件非常简单的事情，因为所有的应用都得到了妥善的分类和直观的展示，三次鼠标点击之内就可以找到应用程序。



而在 Unity 上，应用管理 Lens 默认显示的三行程序中，第二行“已安装”程序是按照首字母顺序来排列的，这样的排列方式很难为用户提供快速定位应用的方便，因为很显然大多数情况下用户进入这个 Lens 并不是为了寻找排在最前面的 “AMD 显卡催化剂控制中心” 等应用。第三行“可供下载的软件”对于用户来说意义也并不大，当用户已经配置好各种软件之后，他是很少会再去软件中心额外安装更多软件的。



此时用户有两种选择，一种是展开第二行的所有“已安装”软件，然后通过右侧的类型过滤器对结果进行过滤。而过滤器的行为也是存在问题的，因为当用户在一个类型中找不到所需的软件而要转到另一个类型时，系统默认会同时选中这两个类型，而不是在两个类型中进行切换。用户会面对很多已经确定不需要的候选结果。

第二种选择是在上方的搜索栏中进行搜索。而 Unity 的搜索栏并不智能，大部分情况下，除非你能记住程序的准确名称，否则你是很难通过搜索得到相应程序的。举例来说，在播放音乐方面，我比较喜欢 Banshee 播放器而不是系统自带的 Rhythmbox，而除非我每次都拼写和搜索“Banshee”，否则常规的模糊查询词汇是无法找到这个程序的。例如搜索“音乐”，Unity 只给出了 Rhythmbox（因为其全名是“Rhythmbox 音乐播放器”），而同样作为音乐播放器的 Banshee 却不见踪影（因为其全名是“Banshee 媒体播放器”，不包括“音乐”字样）。



与此形成对比的是，在 Mac OS X 上同样搜索“音乐”，系统将会识别和列出安装的所有音乐相关程序，即使结果中的 iTunes，Diumoo 和 GarageBand 在名称中都不包含“音乐”字样也毫无影响。



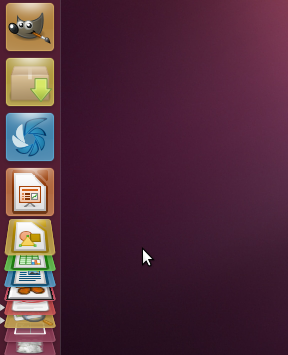
B、Launcher 与 Quicklist

Unity 界面的最左侧部分是一条纵向的快速启动条，正式名称是 Launcher。快速启动条上的图标有三类：系统强制放置的功能图标（Dash 主页、工作区切换器和回收站），用户自定义放置的常用程序图标，以及正在运行中的应用程序图标。

Launcher 的用户体验中规中矩，细节处理得相对比较理想。程序图标的左右两侧可以附加小三角形指示标志。正在运行的程序图标会在左侧有小三角形指示，如果正在运行的程序包括多个窗口，则小三角形的数量也会随之变化。而当前的活动窗口所属的程序，则同时还会在图标右侧显示一个小三角形进行指示。



当 Launcher 上图标增多，开始超出屏幕纵向范围时，最下侧的图标会自动进行折叠。



这样的做法显得新颖而美观，然而却在一定程度上给用户造成了不便。因为在系统操作中非常重要的“工作区切换器”和“回收站”这两个图标是强制固定在 Launcher 最下方的，因此几乎无法逃脱被折叠的命运。当这两个图标被折叠时，无论是切换工作区还是将文件拖进回收站的操作都会变得不那么简单。

在图标上点击右键，可以调出 Quicklist，Quicklist 中包括一些与程序本身相关的常用任务。例如用户可以在 Chrome 浏览器的 Quicklist 中直接启动一个隐身窗口。

这样的右键菜单在 Winodws 7 和 Mac OS X 中都有存在，而与这两者不同的是，Unity 的 Quicklist 可以自行编辑项目和添加常用任务。例如作为一个经常需要“科学上网”的人，经过我编辑定制的终端 Quicklist 是这样的：



C、全局菜单

要让 Firefox，Thunderbird 和 OpenOffice 等程序支持全局菜单相对比较困难。

很多其它应用程序的开发者也应当做出一些优化，以将传统的“每个窗口有不同菜单栏”的显示方式转换到单一菜单栏上来。

——Mark Shuttleworth

上面这段话是 Mark Shuttleworth [第一次宣布采用全局菜单的决定](http://www.markshuttleworth.com/archives/359)时指出的两个问题。可以看到他对全局菜单存在的问题有着清醒的认识，并且很有自信在未来的开发中解决这些问题。然而很遗憾的是，两年时间过去了，Ubuntu 的全局菜单依然实现得并不理想，上述两个突出问题依然存在。

**Unity 的全局菜单行为**

Unity 的全局菜单会在不同的状态下显示不同的内容。

* 当窗口未被最大化，且鼠标未悬停在全局菜单位置时，该位置只显示程序名称。
* 当窗口未被最大化，且鼠标悬停在全局菜单位置时，该位置显示部分窗口名称和菜单内容。
* 当窗口最大化，且鼠标未悬停在全局菜单位置时，该位置只显示程序名称。
* 当窗口最大化，且鼠标悬停在全局菜单位置时，该位置显示窗口控制按钮和菜单内容。

可见，当用户不需要使用全局菜单时，鼠标不会悬停在全局菜单位置，此时该位置只显示程序名称，以减少对用户的打扰。而当用户需要使用全局菜单时，鼠标会悬停在全局菜单位置，此时全局菜单才会出现。

一个显示出设计团队精细用心的地方是，当程序名称过长时，全局菜单会“吞掉”一部分程序名称，以确保不管处于什么状态，打开的是什么程序，全局菜单的各个项目都是基本对齐的，且第一个项目总是出现在同一个地方。



**缺陷一：程序名称？窗口名称？**

在 Unity 中，顶部的全局菜单总是显示程序名称，而窗口的标题栏则总是显示窗口名称。

在大多数时候，这并不是一个问题，因为程序名称和窗口名称一般是一致的。然而当程序名称和窗口名称不一致时，用户将受到一定的困扰。

例如，文件管理器的程序名称是“主文件夹”，因而任何时候全局菜单栏都会显示“主文件夹”。而其窗口名称会随着打开文件夹名称的不同而变化。于是就会出现下面的情况：全局菜单栏上显示的是“主文件夹”，而窗口标题栏上显示的是“极客公园”。由于“主文件夹”也是文件系统的一部分，因而用户可能会对这样的不一致感到困扰。



在这个例子中，问题出在了“主文件夹”这个程序名称上。如果程序名称是“文件管理器”，那么这样的困扰便可以得到缓解。

**缺陷二：全局菜单？窗口菜单？**

传统上，Linux 系统和其它大部分桌面操作系统一样，每个窗口都有自己的菜单。Ubuntu 所做的“全局菜单”，在大部分情况下只是机械地把窗口上的菜单搬到顶部面板而已，而并没有针对全局菜单这一显示方式进行对应的优化。在使用中，这样简单粗暴的做法给用户造成了很多不便。

例如，在 Mac OS X 上，打开一个文件属性窗口的同时，如果想要再打开一个文件管理器窗口，则可以很方便地直接从全局菜单上选择“文件”-“打开新窗口”。因为文件属性窗口是附属于文件管理器的，而文件管理器只有一个唯一的全局菜单。

而在 Ubuntu 12.04 中，打开一个文件属性窗口的同时，如果想要再打开一个文件管理器窗口，则必须先将焦点切换回文件管理器窗口，再从全局菜单上选择“文件”-“新建窗口”。因为文件属性窗口本身并没有菜单栏，因此当焦点处于文件属性窗口时，顶部的全局菜单面板上将不会有任何东西出现。

可见，一个合格的全局菜单系统，必须确保每个程序的所有窗口都只有单一的全局菜单内容，而 Ubuntu 显然并没有做到这一点，在 Ubuntu 中，同一程序之下的各个窗口是互相分裂的，各自有各自的全局菜单栏内容。

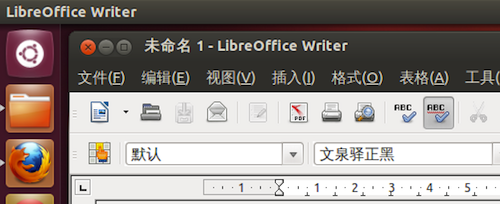
机械地把当前窗口菜单栏内容搬运到全局菜单位置的做法造成的另外一个不便之处是将很多原本不必要搬到全局菜单上的东西放到了全局菜单位置。例如当在 Firefox 浏览器中打开 Autoproxy 配置窗口时，这个窗口的菜单内容居然也会被搬到全局菜单位置上。



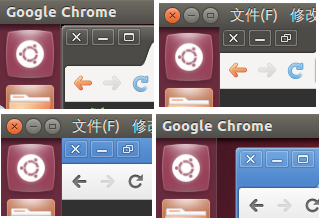
**缺陷三：有效显示？无效显示？**

Ubuntu 的开发者们提供了一种方式，可以自动把窗口上的菜单栏搬到顶部面板显示，而不需要程序开发者额外做出任何修改。这种方式适用于普通 Gtk 和 Qt 程序。而其它特殊类型的程序则无法通过这种方式自动获得全局菜单特性。这就导致了非常尴尬的情况：大部分程序成功获得了全局菜单，而一小部分程序则无法使用全局菜单。

在系统预装的应用程序中，无法自动使用全局菜单的应用程序主要是 Firefox、Thunderbird 和 LibreOffice 套件。Ubuntu 的开发者们通过给 Firefox 和 Thunderbird 预装附加组件的形式变相解决了这两个程序中的全局菜单问题，而对于 LibreOffice，这个问题则一直没有得到解决。



由于同样的原因，很多常用常用软件也无法使用全局菜单特性，例如基于 Java 的脑图绘制软件 Freemind，而常用的浏览器 Google Chrome 更是在各种配置方式下都无法与全局菜单和默认界面风格和谐相处。经常发生两套窗口控制按钮并存、或者界面风格互相冲突的混乱情况。



D、HUD 菜单

HUD 菜单（Head-up Display）是 Ubuntu 12.04 最引人注目的新特性之一。Mark Shuttleworth 在[宣布这一功能时](http://www.markshuttleworth.com/archives/939)自豪地表示这将是对传统菜单系统的一次革命。根据他的表述，HUD 菜单的理念是“告诉系统你的需求，系统就会自动满足你”。HUD 菜单将解决传统菜单的三项缺陷：

* 寻找不易。传统的多层式菜单结构对于用户来说不够友好，初级用户很难在层层叠叠的菜单中找到他们需要的选项。对于大型专业软件来说尤其如此。HUD 菜单让用户提出需求，由系统自动挑选出合适的操作并推动到用户眼前。
* 效率低下。在传统的多层式菜单结构中，用户不得不使用鼠标多次展开菜单才能找到和点击所需项目。为了提高效率用户需要记忆各种复杂的快捷键组合。在 HUD 菜单中，用户只需按 Alt 键触发 HUD 界面，然后直接用键盘搜索，用鼠标回车触发操作，全过程不需要依赖鼠标，也不需要记忆任何快捷键。
* 位置混乱。在不同的程序和不同的操作系统中，同一个菜单项的位置往往各不相同。例如 Firefox 浏览器的“设置”菜单项在 Windows 上位于“工具”下，而在 Linux 上位于“编辑”下，而在 Chrome 中“设置”菜单项又被藏到了地址栏左侧的按钮里。这样的混乱状况给用户带来了很大的不便。使用 HUD 菜单之后，用户只需要告诉系统需要使用“设置”这个菜单项目，而无需记忆这个项目在不同的平台，不同的程序中分别位于哪里。

Mark Shuttleworth 表示，HUD 菜单的长远目标是彻底取代掉传统菜单栏，而未来也会向语音方向继续改进，未来用户甚至不需要动用键盘，直接把需求说出来就能自动得到满足。

HUD 菜单的概念为我们展示了 Linux 桌面系统在人机交互上的美好未来，然而当前阶段的 HUD 菜单实现方式却显得相当原始：系统只是简单地索引了程序的菜单内容和通知区域的各种内容，用户可以按下 Alt 键触发 HUD 菜单界面，然后在搜索框中直接搜索和执行菜单中的项目和通知区域的各种功能。当前阶段用户已经可以通过 HUD 菜单做到以下一些任务：

* 快速寻找菜单项目。在 GIMP 和 Inkscape 等大型软件中，用户可以不再从层叠的菜单中慢慢寻找需要的选项，通过 HUD 菜单可以快速找到选项完成任务。
* 快速打开浏览器书签和历史地址。由于 Firefox 浏览器会直接把书签内容和历史地址放在菜单里，因此用户现在可以方便地通过搜索来打开书签和浏览历史中的项目。
* 快速处理社交信息。由于 Ubuntu 将各种社交工具的信息聚合到了通知区域的信息指示器图标里，因此用户现在可以直接通过 HUD 菜单来完成修改即时聊天工具状态、撰写和查看邮件和发微博等常见任务。
* 快速修改系统状态。由于 Ubuntu 系统将与系统状态相关的各种选项都聚合到了通知区域的各种指示器里，用户现在可以直接在 HUD 菜单中完成连接 VPN、锁定屏幕、关机和重启等相关的操作。

虽然尚处于初级阶段的 HUD 菜单已经可以让用户完成一定的工作，但是当前版本的 HUD 菜单仍然存在很多突出的缺陷，使其无法被广泛应用，对于中国用户来说更是如此。



**适用性不高**

从前文对全局菜单的分析中我们可以看到，目前全局菜单尚未被完整地应用到整个系统。随之而来的坏消息是，HUD 菜单目前却仅支持有全局菜单的应用程序。那些希望从此不再埋头于 LibreOffice 办公套件谜一样的层叠菜单里的用户要失望了，指望借助 HUD 菜单快速画出脑图的 Freemind 用户也要失望了。情况类似的应用程序还有很多。

**快捷键冲突**

HUD 菜单通过 Alt 键进行触发且这一设置无法修改。而 Alt 键在其它程序中有很多用途，二者会发生冲突。例如：

* 由于 Linux 版 QQ 不给力，Linux 用户广泛使用 WebQQ，而最新版 webQQ 的发送消息快捷键中使用到了 Alt
* 很多 Linux 用户通过 Wine 运行《魔兽争霸 3》以便玩 DotA 等流行的游戏，在 DotA 中，显示人物生命值的快捷键是 Alt
* 几乎所有的韩语 Linux 用户都使用 Alt 键来切换输入状态，当不胜其烦的韩语用户[向开发团队报告这一冲突问题时才发现](https://bugs.launchpad.net/unity/+bug/940927)，原来开发团队成员武断地以为全世界的输入法用户都使用 Ctrl+Space 来切换输入法

事实上，Mac OS X 提供了一个与 HUD 菜单理念不同但功能类似的菜单项目搜索特性，触发这个特性的快捷键是 Command+Shift+/，Mac OS X 用三键组合的方式规避了可能的快捷键冲突，而 Ubuntu 让 HUD 菜单独占 Alt 键，导致冲突几乎是不可避免的。



**匹配内容广度和精度低下**

目前阶段 HUD 菜单能够进行匹配的内容范围很小，仅限于程序本身的菜单项目和通知区域的内容。因此如果你想要通过 HUD 搜索到某项东西，必须先让它出现在菜单里。这样机械的行为大大限制了 HUD 菜单的能力。

例如，在巨大的曲库中快速找到音乐并播放是很多音乐爱好者的重要需求，而官方的 HUD 菜单演示视频里也确实显示 HUD 菜单可以用来在 Banshee 音乐播放器中控制乐曲的播放。但实际上，当你打开 Banshee 音乐播放器，在 HUD 菜单里搜索音乐内容不会看到任何结果：



带上放大镜仔细查看官方的视频演示之后发现，官方演示者居然在展示 HUD 菜单控制音乐之前先把音乐全都加到了书签里，用这样的方法来强行使音乐名称出现在菜单中，然后才可以用功能有限的 HUD 菜单来“控制”音乐播放。很显然官方为了展示这个功能，背弃了“书签”的正常用法。

**中文支持非常差**

和其它操作系统类似，Ubuntu 预装的 ibus 输入法框架也不是同类产品中最好的。很多用户会另外安装 fcitx 等其它输入法解决方案。而由于种种原因当前版本的 Unity 界面却仅支持使用 ibus 进行输入，[其它类型的输入法无法在搜索框内输入汉字](https://bugs.launchpad.net/nux/+bug/983254)。这意味着大量中文用户将无法享受到 HUD 菜单带来的便捷体验。

此外，从 Mark Shuttleworth 的介绍和相关的本地化文件中我们可以看到，为了让用户更好地搜索到想要的结果，HUD 菜单对搜索结果采用了智能排序和模糊匹配的优化方法。例如当用户搜索“File”时，即使其不小心输入了“Fiile”，也可以获得“File”这个想要的结果。

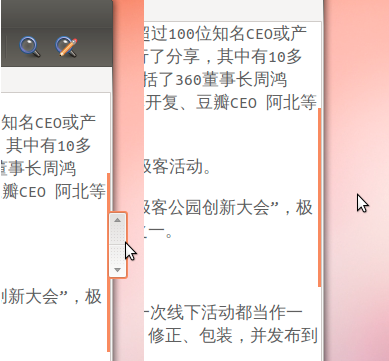
然而类似这样的模糊匹配却并未被应用到中文里来，中文用户要使用 HUD 菜单，首先必须使用 ibus 输入法，其次还必须从头开始输入菜单内容，如果输入的信息只是菜单中的一个词，那么 HUD 菜单将无法准确判断出用户的搜索意图。

例如，Banshee 音乐播放器中有一个菜单项叫做“修复音乐元数据”，当在 HUD 中搜索“修复”的时候，这个选项会出现在候选列表中，但是如果用户并不熟悉菜单项目的具体措辞，而仅仅输入了关键字“元数据”，那么用户什么结果都不会得到。



E、叠加式滚动条

叠加式滚动条（Overlay Scrollbar）是 Unity 界面的另一个与众不同之处。叠加式滚动条由一个主滚动条和一个叠加滑块组成。传统滚动条上下两侧的箭头按钮被集成到叠加滑块上。在用户不需要操作滚动条时，窗口将只显示主滚动条，为了尽量减轻对用户的打扰，主滚动条被做得非常细。当用户将鼠标移动到滚动条上开始使用时，叠加滑块将自动出现，用户可以拖动滑块来滚动窗口内容，也可以用鼠标点击滑块上的上下箭头来滚动窗口内容。



根据 Ubuntu 设计团队官方博客的说法，这样的做法主要是[为了节省触摸屏和小屏幕设备上的空间](http://design.canonical.com/2011/03/introducing-overlay-scrollbars-in-unity/)，缩减滚动条的宽度，将最大的窗口显示面积留给用户，而当用户确实需要滚动窗口时，叠加滑块又可以即时出现来满足用户的需求。

这样的想法确实是很不错的，然而在实际的实现过程中却暴露出了种种缺陷，例如：

**缺陷一：滚动条与滑块不同步**

Ubuntu 设计团队试图用滚动条和滑块分离的方式来优化小屏幕和触屏的滚动体验，然而在实际的使用中，主滚动条和叠加滑块的行为经常不同步，当主滚动条到达顶部或底部不再运动时，滑块却仍然可以运动，而这样的运动却又不会对窗口内容造成任何影响。如下面的视频所示：

**缺陷二：滚动条样式不统一**

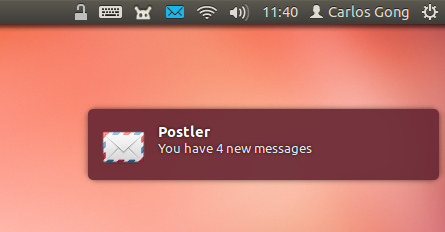
与全局菜单和 HUD 菜单的处境一样，这样的叠加式滚动条样式仅在一部分程序中有用。系统预装程序中的老大难问题依然是 Firefox 和 LibreOffice，而其它一些常用程序也无法使用这个滚动条样式。下面这张图展示了常用程序中的各种滚动条样式中的四种，从左到右依次为 Gedit（标准叠加式滚动条）、Hotot（彩色滚动条）、Chrome（方形滚动条）和 Firefox（圆角滚动条）。



F、通知区域

Ubuntu 的开发人员一向很重视对通知区域的优化，因此 Ubuntu 拥有一个聚合程度很高的通知区域。大量的常见信息来源都被聚合到通知区域的各个指示器中。用户对各路信息的把握和对系统状态的控制非常方便。

例如，来自电子邮件、即时通讯工具、社交网络和各种微博的信息都会被聚合在通知区域的信息指示器中，当有新的信息进入时，信息指示器的颜色会发生改变，同时弹出通知：

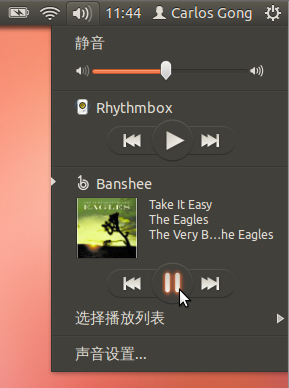


而用户也可以在通知指示器的下拉菜单中查看到来自几乎所有社交工具的信息，并直接更改聊天工具状态、查看收到的信息或创建新信息。这个菜单经过多个版本的反复优化和集成之后，功能已经变得非常完善。



而很多常见的操作也同样被聚合到了通知区域，用户可以在通知区域里方便地控制和切换这些系统状态。

例如，音乐播放器的控制按钮会被集成到声音指示器中，用户可以直接在声音指示器的下拉菜单中控制音乐播放器的行为。而音乐切换时也同样会在通知区域进行弹出通知。



虽然在程序兼容性上还存在一点小问题（例如，我最常用的邮件客户端 Sylpheed 无法被集成到信息指示器菜单里，聊天工具状态切换仅适用于预装的 Empathy）之外，通知区域的设计和实现基本上是理想的，

G、任务切换机制

Unity 的任务切换机制在经过多个版本的改进和打磨之后已经相当完善。用户可以非常高效和直观地在各种场景下进行切换。

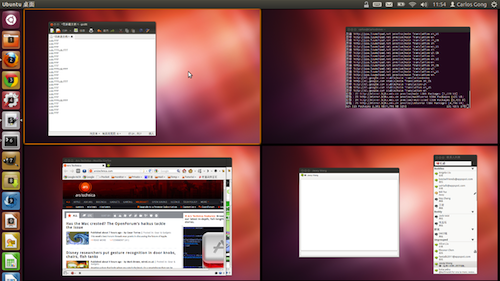
**程序间切换**

通过 Alt+Tab 可以切换不同的程序，如果此时需要查看同一程序下不同的窗口，按下 Alt+` 或者按动向下箭头既可进行细化预览。



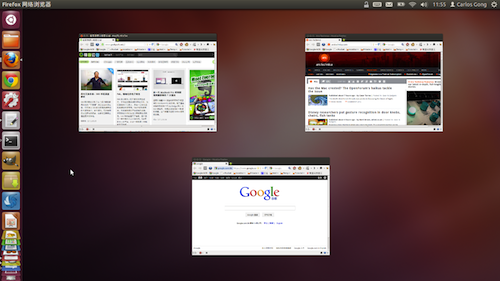
**桌面间切换**

Unity 界面延续了为用户提供多个桌面工作区的传统，单击工作区切换器图标或者使用快捷键 Super（Win/Command）+S 既可俯瞰多个工作区的状况。



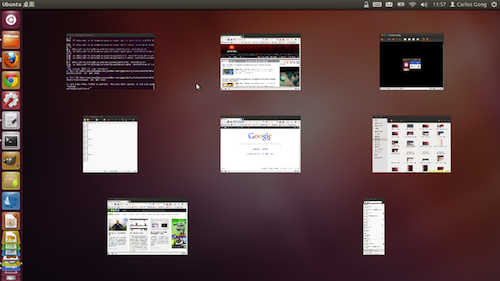
**预览同程序窗口**

如果要预览同一程序的多个窗口，点击一次程序图标，系统会在同一屏幕上展示该程序的所有窗口。



**预览所有窗口**

如果要预览系统中已经打开的所有窗口，使用组合键 Super+W 即可让系统在同一屏幕上列出工作区内所有窗口以供预览。



**3.3、unity图形界面介绍结语**

* 经过多个版本的更新和打磨，Unity 界面已经逐步走向成熟。对于日常操作来说，Unity 足够稳定，也足够完整，可供日常使用。
* Unity 界面已经逐步形成了自己的特色，拥有了一部分独特的贴心细节和创新功能，例如始终保持位置对齐的全局菜单、创新的 HUD 菜单、可供自定义的 Quicklist 和高效的任务切换机制等等。
* 由于规划和设计上的多次转向，Unity 界面中存在一些长时间遗留下来的问题。例如脱离 Linux 生态环境实际情况的全局菜单行为和各种为小屏幕和触屏设备进行优化后遗留下来的不便之处。
* 在对包括中文在内的非字母语言和包括中国用户在内的输入法用户的支持上 Unity 存在很大的不足。这样的种种不足也反映在了国内 Linux 用户社区对 Unity 界面的较低接受程度上。要让中国用户接受 Unity，让 Unity 的种种特性和创新功能可以给中国用户带来实实在在的方便，Unity 还有很长的路要走。

**3.4、Ubuntu中添加卸载桌面环境**

**A、在终端中运行安装：**

　　a、安装XFACE： sudo apt-get install xUbuntu-desktop 点上面的菜单“应用程序－附件－终端”进入到终端，先输入你的用户的密码

　　b、安装KDE：sudo apt-get install kubuntu-desktop

　　使用方法同上

　　c、安装Gnome：sudo apt-get install ubuntu-desktop

使用方法同上

**B、卸载桌面环境（应备份重要文件）：**

　　a、卸载gnome：sudo apt-get --purge remove liborbit2

　　b、卸载kde：sudo apt-get --purge remove kdelibs4c2a libarts1c2a

　　c、卸载xface：sudo apt-get --purge remove xfce4

**五、实验小结**

1、安装Ubuntu至VM虚拟机，初步使用了Ubuntu系统，但还不能够较好的使用该系统对于其中功能不知道如何使用。

2、需要借助Linux命令一览表，进行关于Ubuntu系统下的命令、编程API接口的调用的测试。