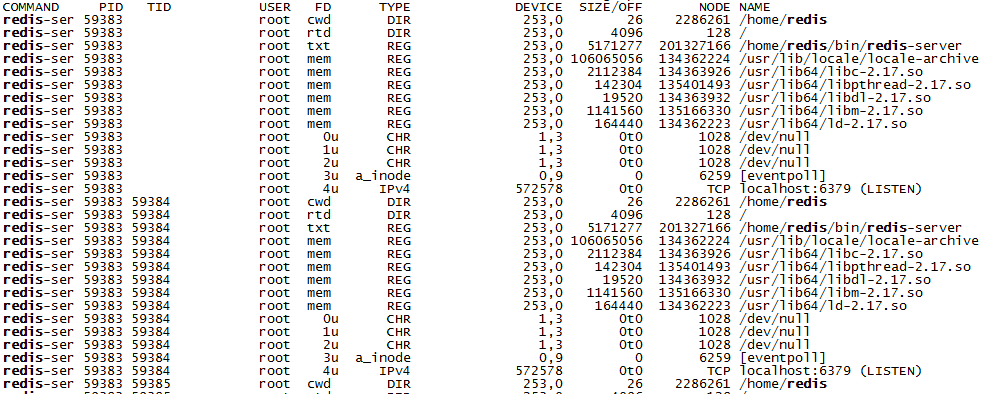
lsof(list open files)是一个列出当前系统打开文件的工具，在Linux环境下任何事物都以文件的形式存在，通过文件不仅仅可以访问常规数据，还可以访问网络连接和硬件。所以如传输控制协议 (TCP) 和用户数据报协议 (UDP) 套接字等，系统在后台都为该应用程序分配了一个文件描述符，无论这个文件的本质如何，该文件描述符为应用程序与基础操作系统之间的交互提供了通用接口。因为应用程序打开文件的描述符列表提供了大量关于这个应用程序本身的信息，因此通过lsof工具能够查看这个列表对系统监测以及排错将是很有帮助的。

## 1 输出信息含义

lsof | more

lsof | awk 'NR==1' && (lsof | awk 'NR==2,NR==10000' | grep 'redis')

lsof -n | more



COMMAND     PID      USER   FD      TYPE             DEVICE  SIZE/OFF       NODE NAME  
init          1      root  cwd       DIR                8,3      4096          2 /  
init          1      root  rtd       DIR                8,3      4096          2 /  
init          1      root  txt       REG                8,3    150352   43122743 /sbin/init  
init          1      root  mem       REG                8,3    154520    3014661 /lib64/ld-2.12.so  
init          1      root    0u      CHR                1,3       0t0       4064 /dev/null  
init          1      root    1u      CHR                1,3       0t0       4064 /dev/null  
init          1      root    2u      CHR                1,3       0t0       4064 /dev/null  
init          1      root    3r     FIFO                0,8       0t0       9147 pipe  
init          1      root    4w     FIFO                0,8       0t0       9147 pipe  
init          1      root    5r      DIR               0,10         0          1 inotify  
init          1      root    6r      DIR               0,10         0          1 inotify  
init          1      root    7u     unix 0xffff8802290f5980       0t0       9148 socket  
init          1      root    9u     unix 0xffff88042b4703c0       0t0      12717 socket

每行显示一个打开的文件，若不指定条件，默认将显示所有进程打开的所有文件。

lsof输出各列信息的意义如下：

**COMMAND**：进程的名称

**PID**：进程标识符

**USER**：进程所有者

**FD**：文件描述符，应用程序通过文件描述符识别该文件，如cwd、txt等

**TYPE**：文件类型，如DIR、REG等

**DEVICE**：指定磁盘的名称

**SIZE**：文件的大小

**NODE**：索引节点（文件在磁盘上的标识）

**NAME**：打开文件的确切名称

FD 列中的cwd(Current Work Directory)为当前工作目录；rtd(Root Directory)为根目录；txt(Text)是程序代码，如应用程序二进制文件本身或共享库，如上列表中显示的/sbin/init 程序；mem(Memory)是加载库（so文件）的内存。

其次数值表示应用程序的文件描述符，这是打开该文件时返回的一个整数。如上的FIFO行，其文件描述符为3r和4w，表示该文件处于只读(r)或只写(w)模式，另外还有读写(u)模式。同时还有大写的W，它表示该应用程序具有对整个文件的写锁，该文件描述符用于确保每次只能打开一个应用程序实例。初始打开每个应用程序时，都具有三个文件描述符，从 0 到 2，分别表示标准输入、输出和错误流。所以大多数应用程序所打开的文件的 FD 都是从 3 开始。

与 FD 列相比，Type 列则比较直观。REG文件和DIR目录；而CHR字符设备和BLK块设备；或者UNIX、FIFO和IPv4，分别表示 UNIX 域套接字、先进先出 (FIFO) 队列和网际协议 (IP) 套接字。

## 2 常用参数

lsof 列出所有打开文件的情况（如上述的例子）

lsof my.log 显示开启文件my.log的进程

lsof -c redis-server显示redis-server进程（cmd）现在打开的文件

lsof -p 1234 列出进程号为1234的进程（process ID）所打开的文件

lsof -g gid 显示归属gid（process group IDs）的进程情况

lsof +d /usr/local/ 显示目录（dir）下被进程开启的文件

lsof +D /usr/local/ 同上，但是会搜索目录下的目录，时间较长

lsof -d 4 显示使用fd为4的进程

lsof -u loginname 登录用户的进程的文件使用情况

lsof -i 用以显示符合条件的网络文件（详见下方）

lsof -i[46] [protocol][@hostname|hostaddr][:service|port]

46 --> i4 i6, version: IPv4 or IPv6

protocol --> TCP or UDP

hostname --> Internet host name

hostaddr --> IPv4地址

service --> /etc/service中的 service name (可以不止一个)

port --> 端口号 (可以不止一个)

**lsof +aL1** select unlinked open files

lsof -i:port1-port2 查看端口port1-port2的使用情况

## 3 使用实例

**恢复删除的文件**

当Linux计算机受到入侵时，常见的情况是日志文件被删除，以掩盖攻击者的踪迹。管理错误也可能导致意外删除重要的文件，比如在清理旧日志时，意外地删除了数据库的活动事务日志。有时可以通过lsof来恢复这些文件。

当进程打开了某个文件时，只要该进程保持打开该文件，即使将其删除，它依然存在于磁盘中。这意味着，进程并不知道文件已经被删除，它仍然可以向打开该文件时，提供给它的文件描述符，进行读取和写入。除了该进程之外，这个文件是不可见的，因为已经删除了其相应的目录索引节点。

**/proc 目录包含了反映内核和进程树的各种文件， /proc目录挂载的是在内存中所映射的一块区域，所以这些文件和目录并不存在于磁盘中，因此当我们对这些文件进行读取和写入时，实际上是在从内存中获取相关信息**。大多数与 lsof 相关的信息，都存储于以进程的 PID 命名的目录中，即 /proc/1234 中包含的是 PID 为 1234 的进程的信息。每个进程目录中存在着各种文件，它们可以使得应用程序，简单地了解进程的内存空间、文件描述符列表、指向磁盘上文件的符号链接和其他系统信息。lsof 程序使用该信息和其他关于内核内部状态的信息，来产生其输出。所以，lsof 可以显示进程的文件描述符和相关的文件名等信息。也就是我们通过访问进程的文件描述符，可以找到该文件的相关信息。

当系统中的某个文件被意外地删除了，只要这个时候系统中还有进程正在访问该文件，那么我们就可以通过lsof从/proc目录下恢复该文件的内容。 假如由于误操作将/var/log/messages文件删除掉了，那么这时要将/var/log/messages文件恢复的方法如下：

首先使用lsof来查看当前是否有进程打开/var/logmessages文件，如下：

# **lsof | grep /var/log/messages**

**syslogd 1283 root 2w REG 3,3 5381017 1773647 /var/log/messages (deleted)**

从上面的信息可以看到，PID 1283（syslogd）打开文件的文件描述符为 2；/var/log/messages已经标记被删除了。因此我们可以在 /proc/1283/fd/2 （fd下的每个以数字命名的文件，表示进程对应的文件描述符）中查看相应的信息，如下：

# head -n 10 /proc/1283/fd/2

Aug 4 13:50:15 holmes86 syslogd 1.4.1: restart.

Aug 4 13:50:15 holmes86 kernel: klogd 1.4.1, log source = /proc/kmsg started.

Aug 4 13:50:15 holmes86 kernel: Linux version 2.6.22.1-8 (root@everestbuilder.linux-ren.org) (gcc version 4.2.0) #1 SMP Wed Jul 18 11:18:32 EDT 2007 Aug 4 13:50:15 holmes86 kernel: BIOS-provided physical RAM map: Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 0000000000000000 - 000000000009f000 (usable) Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 000000000009f000 - 00000000000a0000 (reserved) Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 0000000000100000 - 000000001f7d3800 (usable) Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 000000001f7d3800 - 0000000020000000 (reserved) Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 00000000e0000000 - 00000000f0007000 (reserved) Aug 4 13:50:15 holmes86 kernel: BIOS-e820: 00000000f0008000 - 00000000f000c000 (reserved)

从上面的信息可以看出，查看 /proc/1283/fd/2 就可以得到所要恢复的数据。如果可以通过文件描述符，查看相应的数据，那么就可以使用 I/O 重定向将其复制到文件中，如:

# cat /proc/1283/fd/2 > /var/log/messages

对于许多应用程序，尤其是日志文件和数据库，这种恢复删除文件的方法非常有用。

## 4 实用命令

lsof `which httpd` //哪个进程在使用apache的可执行文件，注意倒引号

lsof `locate redis-server`

lsof /etc/passwd //哪个进程在占用/etc/passwd

lsof /dev/hda6 //哪个进程在占用hda6

lsof /dev/cdrom //哪个进程在占用光驱

lsof -c sendmail //查看sendmail进程的文件使用情况

lsof -c courier -u ^root //显示出哪些文件被以courier打头的进程打开，但不属于用户root

lsof -p 30297 //显示哪些文件被pid为30297的进程打开

lsof +D /tmp //显示所有打开/tmp文件夹(包括子目录)文件实例的进程

lsof -u1000 //查看uid为1000的用户的进程的文件使用情况

lsof -umysql //查看用户mysql的进程的文件使用情况

lsof -u^mysql //查看不是用户mysql的进程的文件使用情况(^是取反的意思)

lsof -i //显示所有打开的IPv[46]文件

lsof -i:80 //显示所有打开80端口的进程

lsof -i:1-1024 //查看端口1-1024运行情况

lsof -i -U //显示所有打开的端口和UNIX domain文件

lsof -iTCP/-iUDP //仅仅显示TCP/UDP连接信息

lsof -i -sTCP:LISTEN //显示TCP监听端口，或者lsof -i | grep -i LISTEN来替代

lsof -i -sTCP:ESTABLISHED //显示已经建立的TCP连接

lsof -u daniel -i @192.168.1.1 //显示daniel连接到192.168.1.1所做的一切

lsof -i @fw.google.com:2150=2180 //显示某个端口范围的打开的连接

lsof -i UDP@localhost //显示哪些进程打开了到localhost的UDP链接

lsof [-i@www.qq.com](mailto:-i@www.qq.com) //显示系统中到远程主机[www.qq.com](http://www.qq.com)的所有连接

lsof [-i@www.qq.com](mailto:-i@www.qq.com):80 //显示系统中到远程主机[www.qq.com](http://www.qq.com) 80端口的所有连接

lsof -i tcp@www.yunlauncher.com:ftp -r //不断查看目前ftp连接的情况

//(-r，lsof会永远不断的执行，直到收到中断信号；+r，lsof会一直执行，直到没有档案被显示,缺省是15s刷新)

lsof -i -n //lsof -n 不将IP转换为hostname，缺省是不加上-n参数，加-n后会很快

注：-r参数表示置lsof为repeat模式，-r t表示每个t秒执行lsof一次，默认为15秒。

## 5 问题排查

在[Linux](http://lib.csdn.net/base/linux)中，当我们使用rm在linux上删除了大文件，但是如果有进程打开了这个大文件，却没有关闭这个文件的句柄，那么linux内核还是不会释放这个文件的磁盘空间，最后造成磁盘空间占用100%，整个系统无法正常运行。这种情况下，通过df和du命令查找的磁盘空间，

在Linux中，当我们使用rm在linux上删除了大文件，但是如果有进程打开了这个大文件，却没有关闭这个文件的句柄，那么linux内核还是不会释放这个文件的磁盘空间，最后造成磁盘空间占用100%，整个系统无法正常运行。这种情况下，通过df和du命令查找的磁盘空间，两者是无法匹配的，可能df显示磁盘100%，而du查找目录的磁盘容量占用却很小。

遇到这种情况，基本可以断定是某些大文件被某些程序占用了，并且这些大文件已经被删除了，但是对应的文件句柄没有被某些程序关闭，造成内核无法收回这些文件占用的空间。

那么，如何查找那些文件被某些程序占用呢：

|  |  |
| --- | --- |
| 1  2  3 | **lsof -n | grep deleted**  **COMMAND     PID      USER   FD      TYPE             DEVICE        SIZE       NODE NAME**  **dd        31708      higkoo    1w      REG                8,2 5523705856     429590 /data/filetest (deleted)** |

命令：lsof -n| grep deleted打印出所有针对已删除文件的读写操作，这类操作是无效的，也正是磁盘空间莫名消失的根本原因。

解决办法1：kill -9 PID   ----只需把进程删掉就能释放空间

解决办法2：echo > big.log