linux lsof详解 - guoguo1980的专栏 - CSDN博客

lsof简介

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。

lsof使用

lsof输出信息含义

在终端下输入lsof即可显示系统打开的文件，因为 lsof 需要访问核心内存和各种文件，所以必须以 root 用户的身份运行它才能够充分地发挥其功能。

```
COMMAND    PID      USER   FD      TYPE     DEVICE     SIZE       NODE      NAMEinit       1
root  cwd      DIR       3,3       1024       2         /init       1         root  rtd
DIR       3,3       1024       2         /init       1         root  txt      REG       3,3
38432      1763452  /sbin/initinit       1         root  mem      REG       3,3       106114
1091620  /lib/libdl-2.6.soinit       1         root  mem      REG       3,3       7560696
1091614  /lib/libc-2.6.soinit       1         root  mem      REG       3,3       79460
1091669  /lib/libselinux.so.1init       1         root  mem      REG       3,3       223280
1091668  /lib/libsepol.so.1init       1         root  mem      REG       3,3       564136
1091607  /lib/ld-2.6.soinit       1         root  10u      FIFO      0,15
1309     /dev/initctl
```

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

其中FD 列中的文件描述符cwd 值表示应用程序的当前工作目录，这是该应用程序启动的目录，除非它本身对这个目录进行更改。

txt 类型的文件是程序代码，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。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。如上的最后一行文件/dev/initctl，其文件描述符为10。u 表示该

文件被打开并处于读取/写入模式，而不是只读 ® 或只写 (w) 模式。同时还有大写 的W 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。文件和目录分别称为 REG 和 DIR。而CHR 和 BLK，分别表示字符和块设备；

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。

lsof常用参数

lsof 常见的用法是查找应用程序打开的文件的名称和数目。可用于查找出某个特定应用程序将日志数据记录到何处，或者正在跟踪某个问题。

例如，linux限制了进程能够打开文件的数目。通常这个数值很大，所以不会产生问题，并且在需要时，应用程序可以请求更大的值（直到某

个上限）。如果你怀疑应用程序耗尽了文件描述符，那么可以使用 lsof 统计打开的文件数目，以进行验证。

lsof语法格式是：

lsof [options] filename

常用的参数列表：

lsof filename 显示打开指定文件的所有进程lsof -a 表示两个参数都必须满足时才显示结果lsof -c string 显示COMMAND列中包含指定字符的进程所有打开的文件lsof -u username 显示所属user进程打开的文件lsof -g gid 显示归属gid的进程情况lsof +d /DIR/ 显示目录下被进程打开的文件lsof +D /DIR/ 同上，但是会搜索目录下的所有目录，时间相对较长lsof -d FD 显示指定文件描述符的进程lsof -n 不将IP转换为hostname，缺省是不加上-n参数lsof -i 用以显示符合条件的进程情况lsof -i[46] [protocol][@hostname|hostaddr] [:service|port] 46 --> IPv4 or IPv6 protocol --> TCP or UDP hostname --> Internet host name hostaddr --> IPv4地址 service --> /etc/service中的 service name (可以不只一个) port --> 端口号 (可以不只一个)

例如： 查看22端口现在运行的情况

lsof -i :22COMMAND PID USER FD TYPE DEVICE SIZE NODE NAMEsshd 1409 root 3u IPv6 5678 TCP *:ssh (LISTEN)

查看所属root用户进程所打开的文件类型为txt的文件:

lsof -a -u root -d txtCOMMAND PID USER FD TYPE DEVICE SIZE NODE NAMEinit 1 root txt REG 3,3 38432 1763452 /sbin/initmingetty 1632 root txt REG 3,3 14366 1763337 /sbin/mingettymingetty 1633 root txt REG 3,3 14366 1763337 /sbin/mingettymingetty 1634 root txt REG 3,3 14366 1763337 /sbin/mingettymingetty 1635 root txt REG 3,3 14366 1763337 /sbin/mingettymingetty 1636 root txt REG 3,3 14366 1763337 /sbin/mingettymingetty 1637 root txt REG 3,3 14366 1763337 /sbin/mingettykdm 1638 root txt REG 3,3 132548 1428194 /usr/bin/kdmX 1670 root txt REG 3,3 1716396 1428336 /usr/bin/Xorgkdm 1671 root txt REG 3,3 132548 1428194 /usr/bin/kdmstartkde 2427 root txt REG 3,3 645408 1544195 /bin/bash... ...

lsof使用实例

一、查找谁在使用文件系统

在卸载文件系统时，如果该文件系统中有任何打开的文件，操作通常将会失败。那么通过lsof可以找出那些进程在使用当前要卸载的文件系统，如下：

lsof /GTES11/COMMAND PID USER FD TYPE DEVICE SIZE NODE NAMEbash 4208 root cwd DIR 3,1 4096 2 /GTES11/vim 4230 root cwd DIR 3,1 4096 2 /GTES11/

在这个示例中，用户root正在其/GTES11目录中进行一些操作。一个 bash是实例正在运行，并且它当前的目录为/GTES11，另一个则显示的是vim正在编辑/GTES11下的文件。要成功地卸载/GTES11，应该在通知用户以确保情况正常之后，中止这些进程。 这个示例说明了应用程序的当前工作目录非常重要，因为它仍保持着文件资源，并且可以防止文件系统被卸载。这就是为什么大部分守护进程（后台进程）将它们的目录更改为根目录、或服务特定的目录（如 sendmail 示例中的 /var/spool/mqueue）的原因，以避免该守护进程阻止卸载不相关的文件系统。

二、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。

在/proc 目录下，其中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、指向磁盘上的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/messagessyslogd   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。同时还可以看到/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/2Aug  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 2007Aug  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/8663/fd/15 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。