Python 问题集

屈庆磊整理

quqinglei@icloud.com

2013 年 5 月

目录

1 sshpass 工具

```
sshpass -p123456 ssh user@192.168.1.2 df | grep /dev/sda1 | awk '{print $5}'
```

2 strace 系统调用追踪工具

这个工具可用来跟踪进程的系统调用状态，下面是一个简单的例子，我们跟踪的是 vsftpd 这个进程的系统调用过程。

```
root@ubuntu:~# strace -p4449
Process 4449 attached - interrupt to quit
accept(3, {sa_family=AF_INET, sin_port=htons(57334), sin_addr=inet_addr("127.0.0.1")},
[16]) = 4
clone(child_stack=0, flags=0x28000000|SIGCHLD) = 6875
close(4)                                = 0
```

```
accept(3, 0xbfc055e0, [28])             = ? ERESTARTSYS (To be restarted)
--- SIGCHLD (Child exited) @ 0 (0) ---
alarm(1)                                = 0
sigreturn()                             = ? (mask now [])
alarm(0)                                = 1
waitpid(-1, NULL, WNOHANG)              = 6875
waitpid(-1, NULL, WNOHANG)              = -1 ECHILD (No child processes)
accept(3,
```

3 iptables 介绍

3.1 一些例子

```
# 删除现有的规则
iptables -F
# or
iptables --flush

# 屏蔽制定的 IP 地址
BLOCK="192.168.1.2"
iptables -A INPUT -i eth0 -s "$BLOCK" -j DROP
# 仅仅屏蔽 TCP 数据包
iptables -A INPUT -i eth0 -p tcp -s "$BLOCK" -j DROP

# 允许来自外部的 PING 测试
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

# 允许从本机 PING 外部主机
iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT

# 允许回环测试
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# 允许所有 SSH 连接请求
iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state \
NEW,ESTABLISHED -j ACCEPT
iptables -A output -o eth0 -p tcp --dport 22 -m state --state \
NEW,ESTABLISHED -j ACCEPT

# 仅允许来自内网的 SSH 连接请求
iptables -A INPUT -i eth0 -p tcp -s 192.168.1.0/24 --dport 22 -m state \
NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state \
ESTABLISHED -j ACCEPT
```

```
# 不允许任何网络访问
iptables -A INPUT -j DROP
iptables -A OUTPUT -j DROP
```

4 gdb 用法

4.1 基础

4.1.1 说明

不是所有的二进制文件都是可以使用 gdb 来进行调试的，这是因为他们不一定包含一些用来调试的符号标志。当运行这些二进制文件的时候这些符号或多或少的告诉 gdb 去哪里寻找源代码。

使用 gdb 进行调试必须满足两个条件。第一源代码路径不要移动，也就是说你编译完成后不要把源代码移动位置，移动生成的二进制文件到其他路径没关系，第二件事情就是在编译的时候要加参数，比如 *gcc hello.c -o hello -g* 或者 *gcc hello.c -o hello -ggdb3*

4.1.2 启动 gdb

启动方法很多，下面是最简单的一种，它会自动装载二进制文件。

```
bash$ gdb binary
```

当 gdb 启动的时候，会有类似如下的输出

```
GNU gdb (gdb) 7.5-ubuntu
Copyright (C) 2012 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This gdb was configured as "i686-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
(gdb)
```