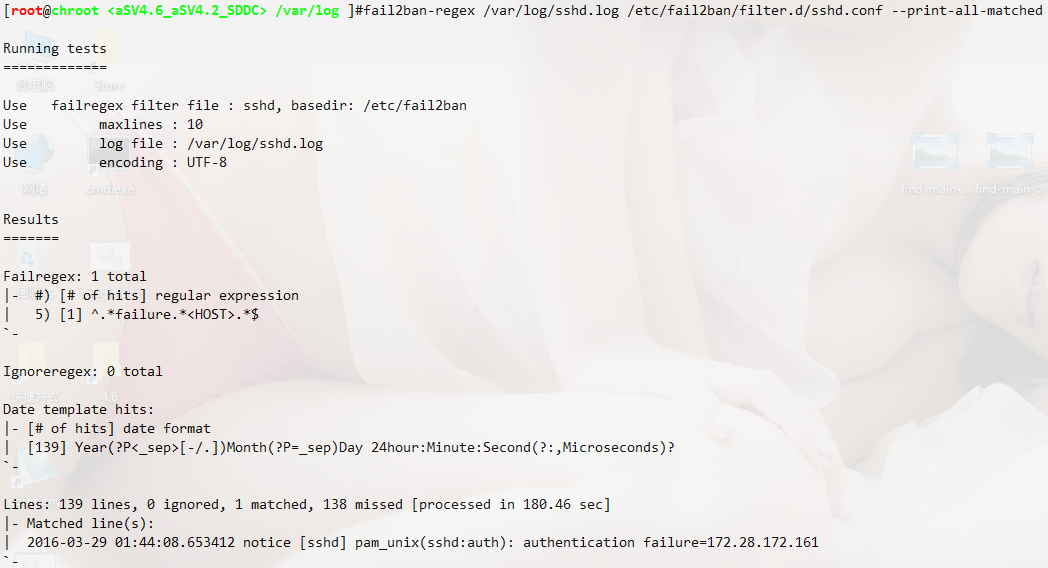
# 利用日志做监控

[mnstory.net](http://mnstory.net)

利用日志做监控，适用于不便修改源码或不便破坏现有环境时，可以通过日志内容采取不同动作，通过对日志本身的实时监控并分析，提取有效信息并采取对应行动。

应用场景可随意想象：例如，之前发现aSV的ssh密码被暴力破解，我们可以通过日志监控暴力破解行为，然后做出阻止。

这并不是一个新技术，也不困难，fail2ban <http://www.fail2ban.org/> 做的挺不错，使用起来效果如图：



fail2ban需要python运行时，aSV环境默认是不装python，但日志监控的思想是可以借鉴的，比如，我在大集群版本应用过这种方式。

问题描述：

QEMU的hmp有的时候会链接不上，报timeout，比较容易在集群负载较高的时候出现，我当时的主观推论主线程忙于处理磁盘IO，导致accept被延迟，导致timeout，但是还需要证据。

如果碰巧，timeout的时候，我们可以调试QEMU，那可以看到主线程在做啥，那就可以知道是不是由于IO耗时导致，但是这种机遇分布零散，而且出现那么一会儿就会消失，如果能在timeout的时候，打印出QEMU的堆栈信息，那也不错。

利用日志监控思想，先写一段脚本：

cat\_bt()

{

    echo "$\*  " $(date) >> /sf/log/thread.log

    local pid=$(ps auxf | grep -P "kvm -id $1" | grep -v grep | awk '{print $2}')

    if [ -z "$pid" ]; then

        lerror "can't find vmid $1 process"

        return 1

    fi

    rtinject $pid 'thread apply all bt' >> /sf/log/thread.log

}

extra\_hmp\_error()

{

    while read line

    do

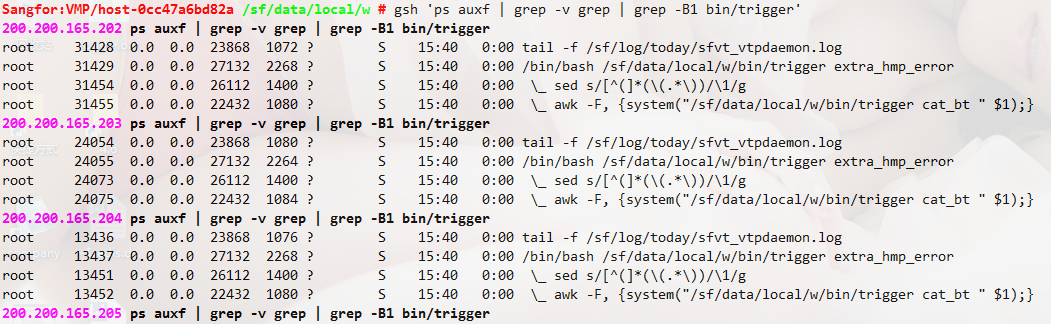
        grep -oP 'vm\_human\_monitor\_command\(\d+,.\*\)' | sed 's/[^(]\*(\(.\*\))/\1/g' | awk -F, '{system("'$0' cat\_bt " $1);}'

    done < "${1:-/dev/stdin}"

}

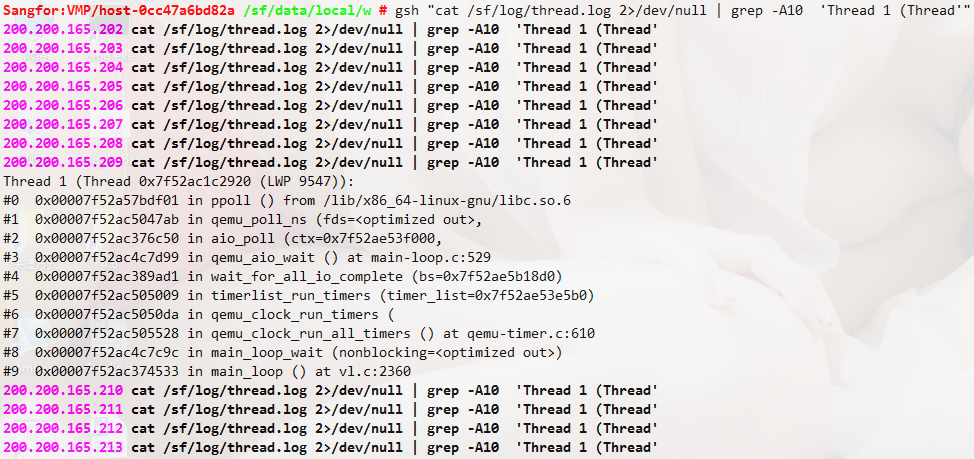
当检测到vm\_human\_monitor\_command的时候，提取QEMU ID, 通过ID找到PID，然后将栈信息输出到/sf/log/thread.log。

先在每台主机都运行这脚本：



然后回去睡觉！

就像渔夫打鱼，第二天过来一收网，捕到不少条：



每条的栈信息都直指wait io complete，证明确实是主线程提交IO卡主，进一步查出，当时磁盘的io util非常高，写入老慢老慢。

2016/3/29