

cron script hangs after a while (seems to)

Source: <http://unix.derkeiler.com/Newsgroups/comp.unix.shell/2005-01/0313.html>

From: Nick Sinclair (*null_device_at_NOFRILLS.ssl-mail.com*)

Date: 01/07/05

Date: Fri, 07 Jan 2005 15:26:08 +1100

Hi all,

THis might be off-topic, but I was wondering if anyone else has ever experienced this kind of behaviour.

I have this package of scripts that act as a 'bot' running on an 'ACID/SNORT/MySQL' installation. Basically, every 2 mins Cron runs 3 of these scripts from my package, which do the following respectively.

1. Extract ip addresses from MySQL and create/update 'flat files' common to my package.
2. Read in ip addresses from these 'flat files' and then, if there are any new addresses since the last run, add rules to block the /24 address in iptables
3. Create an html page from the log created by the previous script.

When I come into the office some mornings, and check the 'flat files', 'log file' and 'html page', the script seems to have stopped working at some point. These times are random, and don't coincide with any other tasks.

I then pull up the Analysis Console (ACID) in a browser, and everything seems to kick off again. I will have a single big entry in the logs, that represents every event that has happened since my package seemed to stall.

How could these things be related??? There is nothing unique in any log files et al.

Has anyone ever experieenced this kind of thing? Any feedback is most welcome.

Thanks in advance, if anyone can suggest something.

If you want to take a look, the whole package is here:

http://203.102.255.156/downloads/public/ip_bot-0.9.8-3.tgz

=====

These are the 3 scripts that are run sequentially every 2 minutes:

comp.unix.shell: cron script hangs after a while (seems to)

```
#!/bin/bash
#
# ip_bot/get_ip_mysql
# Get the IP addresses from MySQL
# on a 'Nix box running SNORT, MySQL and ACID
#####

###> We're not insane
umask 0027
shopt -s -o nounset
declare -rx SCRIPT=${0##*/}
sleep 10

source /usr/local/etc/ip_bot/ip_bot.conf

# < ----- Snip here ----- >
##
###
###> Extract the data from MySQL (Original idea)

declare ip_src_table
declare ip_src_table_ref
declare ip_list
declare sig_class_id

### Check to make sure that the path to iptables is correct
if [ ! -f "$MYSQL" ]; then
    printf "%s\n" "You fsckwit, no MySQL here: $MYSQL"
    exit 127
fi

for sig_class_id in $sig_classes; do

### Extract our ip addresses from different criteria in $sig_classes
ip_src_table=`"$MYSQL" --user="$DB_USER" --password="$DB_PASS" \
--exec="use $DB_NAME; \
select ip_src,inet_ntoa(ip_src) from acid_event where sig_class_id =
$sig_class_id"`

if [ $? -ne 0 ]; then
    printf "$SCRIPT MySQL query failed." >&2
    exit 192
fi

### Extract the signature and ip address data for our reference table
ip_src_table_ref=`"$MYSQL" --user="$DB_USER" --password="$DB_PASS" \
--exec="use $DB_NAME; \
select inet_ntoa(ip_src),sig_name from acid_event where sig_class_id =
$sig_class_id"`
```

cron script hangs after a while (seems to)

comp.unix.shell: cron script hangs after a while (seems to)

```
if [ $? -ne 0 ]; then
    printf "$SCRIPT MySQL query failed." >&2
    exit 192
fi

### Get the list of IP's from our MySQL query
ip_list=`echo "$ip_src_table" | awk '{print $2}' | sort | uniq \
sed -e 's/[^0-9*\.\0-9*\.\0-9*\.\0-9*].*$/g`

### Remove any matches within our own private network
ip_list=`echo "$ip_list" | grep -v "$HOME_NET"`

### Create our temp lists from the results from our processed queries
printf "%s\n" "$ip_list" | grep -v -f "$IP_BOT_DIR"/ip_white_list \
>> "$IP_BOT_DIR"/tmp/ip_black_list.tmp
printf "%s\n" "$ip_src_table_ref" | grep -v -f "$IP_BOT_DIR"/ip_white_list \
\
>> "$IP_BOT_DIR"/tmp/ip_black_list_table.tmp

done

###> Put your own 'drop-in' replacement here, for adding the
### IP addresses to "$IP_BOT_DIR"/tmp/ip_black_list.tmp (one per line)
## it's that simple :)
# < ----- Snip here ----- >

### Generate our 'source method' only list of offending IP addresses
cat "$IP_BOT_DIR"/tmp/ip_black_list.tmp | sort >
"$IP_BOT_DIR"/ip_black_list.db

### Sort and individualize our event look-up table
cat "$IP_BOT_DIR"/tmp/ip_black_list_table.tmp | sort | uniq \
> "$IP_BOT_DIR"/ip_black_list_table

### Generate a 'raw' consolidated list of existing IP's and the new IP's
cat "$IP_BOT_DIR"/ip_black_list >> "$IP_BOT_DIR"/tmp/ip_black_list.tmp
cat "$IP_BOT_DIR"/ip_black_list_supp >> "$IP_BOT_DIR"/tmp/ip_black_list.tmp

### Generate a backup of the consolidated list
cp "$IP_BOT_DIR"/ip_black_list "$IP_BOT_DIR"/tmp/ip_black_list.bak

### Finalize our list of IP's by sorting, removing the duplicates and
### then turning them into /24 addresses, and then we're done.
cat "$IP_BOT_DIR"/tmp/ip_black_list.tmp | sed -e 's/\.[0-9]*$/\0/g' \
-e '/^$/d' | sort | uniq > "$IP_BOT_DIR"/ip_black_list

### Zero out the temp file for next time.
:> "$IP_BOT_DIR"/tmp/ip_black_list.tmp

### Zero out our tmp look-up table for next time
:> "$IP_BOT_DIR"/tmp/ip_black_list_table.tmp

cron script hangs after a while (seems to)
```

comp.unix.shell: cron script hangs after a while (seems to)

exit 0

```
=====
=====
=====
```

```
#!/bin/bash
#
# ip_bot/ip_block
#
# This is the script that takes care of inserting
# firewall rules into the iptables chain as defined
# in the $CHAIN variable. This script will use the
# master list: "$IP_BOT_DIR"/ip_black_list to check
# for any new additions that need to be made.
#
```

```
umask 0027
shopt -s -o nounset
declare -rx SCRIPT=${0##*/}
```

```
source /usr/local/etc/ip_bot/ip_bot.conf
source /usr/local/etc/ip_bot/hash_function
```

```
### No need to change these below.
declare -r date=`date +%d.%m.%y`
declare -r time=`date +%r`
declare -i counter=0
declare rules_count
declare line
declare ip_address
declare alert_types
declare ip_ref
```

```
### Check to make sure the path to iptables is correct
if [ ! -f "$IPTABLES" ]; then
  printf "%s\n" "You fsckwit, no iptables here: $IPTABLES"
  exit 127
fi
```

```
### If the custom chain doesn't already exist then create it
$IPTABLES -n -L "$CHAIN" &>/dev/null
if [ $? -ne 0 ]; then
  $IPTABLES -N "$CHAIN"
  $IPTABLES -A FORWARD -j "$CHAIN"
fi
```

```
#####
### Extract the current /24 addresses from our chain ###
#####
```

cron script hangs after a while (seems to)

comp.unix.shell: cron script hangs after a while (seems to)

```
$IPTABLES -n -L "$CHAIN" | \
sed -e 1,2d -e 's/^. *-- //g' -e 's/\24.*$/g' \
> "$IP_BOT_DIR"/tmp/iptables_$CHAIN.tmp

## Find out what we already have in our iptables chain
### This is so that we don't fill our log on the first run

rules_count=`wc -l "$IP_BOT_DIR"/tmp/iptables_$CHAIN.tmp | awk '{print $1}'`

#####
### Create the hash table from our iptables chain ###
#####

while read line; do
  add_hash "$line"
done < "$IP_BOT_DIR"/tmp/iptables_$CHAIN.tmp

#####
### Now take each ip address in ip_black_list ###
### and look it up in our hash table of iptables rules ###
#####

while read line; do
  lookup_hash "$line"
  if [ "$RESULT" == "NO" ]; then

    ### Log some stuff ###
    if [ "$counter" -lt 1 ]; then
      counter=1
      printf "%s\n" >> "$LOGFILE"
      printf "%s\n" "Adding new offending IP addresses" >> "$LOGFILE"
      if [ "$rules_count" -lt 1 ]; then
        printf "%s\n" \
          "This is probably first run from boot - Addresses not logged" >>
"$LOGFILE"
      fi
    fi

    ### Get our IP address and make sure it's a /24 address for iptables
    ip_address=`echo "$line" | sed -e 's/^[0-9]*$/\0/24/g`

    ### Match the ip address against our reference table and get the alert
    type
    ip_ref=`echo "$line" | sed -e 's/^[0-9]*$/./g`
    alert_types=`grep "$ip_ref" "$IP_BOT_DIR"/ip_black_list_table | \
    sed -e 's/^/ >>> /g`

    ### Add the rule
    $IPTABLES -A "$CHAIN" -i eth1 -s "$ip_address" -j DROP &>/dev/null
```

cron script hangs after a while (seems to)

comp.unix.shell: cron script hangs after a while (seems to)

```
### If successful, then log the ip address and alert type
if [ $? -eq 0 ]; then
  if [ "$rules_count" -gt 0 ]; then
    printf "%s\n" \
      "$date $time $SCRIPT: Added $ip_address to $CHAIN chain" >> "$LOGFILE"
    printf "%s\n" "$alert_types" >> "$LOGFILE"
  fi
fi
```

```
fi
done < "$IP_BOT_DIR"/ip_black_list
```

```
: > "$IP_BOT_DIR"/tmp/iptables_$CHAIN.tmp
```

```
exit 0
```

```
=====
=====
=====
```

```
#!/bin/bash
#
# ip_bot/generate_html
# This script simply takes the logfile created
# by ip_bot/ip_block and inserts it into a
# template using ipbot_html-top.tmpl and
# ipbot_html-bottom.tmpl
```

```
umask 0022
shopt -s -o nounset
declare -rx SCRIPT=${0##*/}
```

```
source /usr/local/etc/ip_bot/ip_bot.conf
```

```
declare line
```

```
cat /usr/local/etc/ip_bot/ipbot_html-top.tmpl | \
sed -e "s|</title>|: $TITLE</title>|" | \
sed -e "/IP BLACKLIST REPORT/s/</td>/: $TITLE</td>/" | \
> "$APACHE_PATH"/blacklist.html
```

```
while read line; do
```

```
printf "%s\n" "$line" | \
sed -e '/ip_block/s/^\<b>/g' -e '/ip_block/s/$/</b>/g' -e 's/$/<br>/g' | \
>> "$APACHE_PATH"/blacklist.html
```

```
done < $LOGFILE
```

```
cat /usr/local/etc/ip_bot/ipbot_html-bottom.tmpl | \
>> "$APACHE_PATH"/blacklist.html
```

cron script hangs after a while (seems to)

comp.unix.shell: cron script hangs after a while (seems to)

```
if [ ! -f "$APACHE_PATH"/ip_bot.gif ]; then  
  cp -a /usr/local/etc/ip_bot/ip_bot.gif "$APACHE_DIR"  
fi
```

```
exit 0
```