Sysdig to the Root of the Problem

SCaLE x13 - Los Angeles, CA - February 19-22

What do you do when something breaks?

How do you troubleshoot when a service fails to start?

Approach

- Systematically explore the problem
  - ~~guessing~~
  - hypothesise what is happening
  - verify your hypothesis

Approach

- theory
- tools

Example Approach

- Theory: DHCP address allocation

- Tool: tcpdump - look at what's actually happening on the wire

"Can we do this for processes?"

—enquiring minds want to know

Sysdig

read, write, compress

- record$ sysdig -w file.scap
- replay$ sysdig -r file.scap
- compressed$ sysdig -zw file.scap.z; sysdig -zr file.scap.z

"Can you recreate the bug?"

–asked by every support technician, always

Example 1: Apache fails to start

```
$ sudo sysdig -zw file.scap.z
$ sudo service apache2 start
```

Example 1: Apache fails to start

```
$ sudo sysdig -zr file.scap.z "proc.name=apache2 and evt.failed=true"
3952 19:16:27.418395213 0 apache2 (26658) < futex res=-11(EAGAIN)
29372 19:16:27.428512381 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30266 19:16:27.430537651 0 apache2 (26658) < stat res=-2(ENOENT) path=/usr/
lib/apache2/suexec
30272 19:16:27.430560641 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30278 19:16:27.430565446 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30406 19:16:27.430806180 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30412 19:16:27.430810159 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30524 19:16:27.431052661 0 apache2 (26658) < connect res=-2(ENOENT) tuple=0-
>ffff88001ce8f180 /var/run/nscd/socket
30753 19:16:27.431761879 0 apache2 (26658) < open fd=-2(ENOENT) name=/var/
run/apache2/apache2.pid flags=1(O_RDONLY) mode=0
30761 19:16:27.431769458 0 apache2 (26658) < bind res=-98(EADDRINUSE)
addr=0.0.0.0:80
```

Example 1: Apache fails to start

```
$ sysdig -zr file.scap.z -c lsof
"'fd.sport = 80'"
COMMAND                 PID      TID      USER
FD      TYPE            NAME
nc                      26596    26596    root
3       ipv4            0.0.0.0:80
```

filtering

- possible fields to filter$ sysdig -l
- Comparison operators: =, !=, <, <=, >, >=, contains
- Boolean operators: and, or, not

best practice

- always encapsulate filters in quotes

- encapsulate chisel parameters in quotes

- if multiple chisel parameters, quote individual parameters as well

Example: filtering

```
$ sysdig evt.num < 10
```

Example: filtering

~~$ sysdig evt.num < 10~~

$ sysdig “evt.num < 10”

Example: filtering

~~$ sysdig evt.num < 10~~

```
$ sysdig “evt.num < 10”

$ sysdig -c lsof “proc.name = nc” \
“evt.num = 310”
```

Example: filtering

```
$ sysdig evt.num < 10
$ sysdig “evt.num < 10”

$ sysdig -c lsof “proc.name = nc” \
“evt.num = 310”

$ sysdig -c lsof “‘proc.name = nc’” \
“evt.num = 310
```

chisels

- pre-packaged functionality within sysdig
- uses same filtering syntax
- list chisels$ sysdig -cl

lsof chisel

```
$ sysdig -zr file.scap.z -c lsof
“‘proc.name=mysqld’” “evt.num=250”
```

lsof chisel - point in time

```
$ sysdig -zr file.scap.z -c lsof "proc.name=nc" "evt.num=1"
COMMAND                 PID     TID     USER    FD      TYPE
NAME
nc                      13673   13673   vagrant 0       file
/home/v/foo
nc                      13673   13673   vagrant 1       file
/dev/pts/0
nc                      13673   13673   vagrant 2       file
/dev/pts/0
nc                      13673   13673   vagrant 3       ipv4
0.0.0.0:8081
$ sysdig -zr file.scap.z -c lsof "proc.name=nc"
"evt.num=3477"
COMMAND                 PID     TID     USER    FD      TYPE
NAME
$
```

/proc

- man 5 proc

/proc/net/[tcp|tcp6|udp|udp6]

- Holds a dump of the respective socket table.

Example: ls -l /proc/<mysqld>/fd/

```
lrwx------ 1 root root 64 Jan 30 23:20 0 -> /dev/null
l-wx------ 1 root root 64 Jan 30 23:20 1 -> /var/log/
mysql/error.log
lrwx------ 1 root root 64 Jan 30 23:20 10 -> socket:[9207]
lrwx------ 1 root root 64 Jan 30 23:20 11 -> /tmp/ib3DDpk8
(deleted)
lrwx------ 1 root root 64 Jan 30 23:20 12 -> socket:[9208]
lrwx------ 1 root root 64 Jan 30 23:20 13 -> /var/lib/
mysql/mysql/host.MYI
lrwx------ 1 root root 64 Jan 30 23:20 14 -> /var/lib/
mysql/mysql/host.MYD
lrwx------ 1 root root 64 Jan 30 23:20 15 -> /var/lib/
mysql/mysql/user.MYI
lrwx------ 1 root root 64 Jan 30 23:20 16 -> /var/lib/
mysql/mysql/user.MYD
```

lsof replaced?

- $ lsof +L1 - find all files with link count <1

ps chisel

```
$ sysdig -zr file.scap.z -c ps
TID     PID  USER   VIRT     RES         FDLIMIT   CMD
1       1    root   32.73M   2.85M       1024      init
355     355  root   19.02M   656.00KB    1024      upstart-
udev-br
361     361  root   48.41M   1.49M       1024      systemd-
udevd
508     508  root   22.87M   1.09M       1024      rpcbind
528     528  statd  21.04M   1.36M       1024
rpc.statd
543     543  root   14.90M   632.00KB    1024      upstart-
socket-
547     547  root   9.98M    2.82M       1024      dhclient
```

/proc/[pid]/fd/

/proc/[pid]/fd/

This is a subdirectory containing one entry for each file which the process has open, named by its file descriptor, and which is a symbolic link to the actual file. Thus, 0 is standard input, 1 standard output, 2 standard error, etc.

netstat chisel

```
$ sysdig -zr file.scap.z -c netstat
Proto Server Address    Client Address   State         TID/PID/Program
Name
tcp   127.0.0.1:3306    0.0.0.0:*        LISTEN        1138/1038/mysqld
tcp   127.0.0.1:3306    0.0.0.0:*        LISTEN        1150/1038/mysqld
udp   0.0.0.0:56538     0.0.0.0:*        LISTEN        522/522/dhclient
udp   0.0.0.0:68        0.0.0.0:*        LISTEN        522/522/dhclient
udp   0.0.0.0:111       0.0.0.0:*        LISTEN        524/524/rpcbind
udp   0.0.0.0:691       0.0.0.0:*        LISTEN        524/524/rpcbind
tcp   0.0.0.0:111       0.0.0.0:*        LISTEN        524/524/rpcbind
tcp   127.0.0.1:3306    0.0.0.0:*        LISTEN        1057/1038/mysqld
tcp   127.0.0.1:3306    0.0.0.0:*        LISTEN        1066/1038/mysqld
tcp   10.0.2.15:22      10.0.2.2:50362   ESTABLISHED   9426/9426/sshd
tcp   0.0.0.0:22        0.0.0.0:*        LISTEN        969/969/sshd
tcp   10.0.2.15:22      10.0.2.2:50362   ESTABLISHED   9350/9350/sshd
```

man 1 strace

- In the simplest case strace runs the specified command until it exits. It intercepts and records the system calls which are called by a process and the signals which are received by a process.

system calls

- man 2 syscalls

common system calls

- open/close
- read/write
- connect/accept/bind
- fork
- exec; execve

spy_users chisel

```
$ sysdig -zr file.scap.z -c spy_users 3
13209 04:30:47 vagrant) sudo nc -l 80
    13209 04:30:47 root) nc -l 80
13209 04:30:53 vagrant) sudo service apache2 start
    13209 04:30:53 root) /bin/sh /usr/sbin/service apache2 start
        13209 04:30:53 root) basename /usr/sbin/service
13209 04:30:53 root) env -i LANG=en_US.UTF-8 PATH=/usr/local/sbin:/us...
13209 04:30:53 root) /bin/sh /etc/init.d/apache2 start
    13209 04:30:53 root) run-parts --lsbsysinit --list /lib/lsb/init-...
    13209 04:30:53 root) /usr/bin/tput hpa 60
    13209 04:30:53 root) mktemp
    13209 04:30:53 root) env -i LANG=C PATH=/usr/local/sbin:/usr/loca...
    13209 04:30:53 root) /bin/sh /usr/sbin/apache2ctl configtest
        13209 04:30:53 root) id -u
        13209 04:30:53 root) /usr/sbin/apache2 -t
    13209 04:30:53 root) rm -f /tmp/tmp.OqJNPcbXdg
    13209 04:30:53 root) env -i LANG=C PATH=/usr/local/sbin:/usr/loca...
    13209 04:30:53 root) /bin/sh /usr/sbin/apache2ctl start
        13209 04:30:53 root) id -u
        13209 04:30:53 root) rm -f /var/run/apache2/*ssl_scache*
```

echo_fds chisel

- description: print the data read and written for any FD
- pro-tip: less -R
  - lets **R**aw ansi color bleed through

Example: echo_fds 1

```
$ sysdig -zr file.scap.z -A -c echo_fds
"proc.name=mysql"
<snip/>
------ Read 3.42KB from /etc/mysql/my.cnf (mysql)
#.# The MySQL database server configuration
file..#.# You can copy this to one o
------ Read 21B from /etc/mysql/conf.d/
mysqld_safe_syslog.cnf (mysql)
[mysqld_safe].syslog.
------ Write 16B to /dev/tty (mysql)
Enter password:
------ Read 11B from /dev/tty (mysql)
password01.
<snip/>
```

"OMG! A password!"

–your friendly security officer

"But they don't have the username."

–someone who wasn't paying attention

Example: echo_fds 1

```
$ sysdig -zr file.scap.z “proc.name =
mysql” -c spy_users
1621 01:20:52 vagrant) mysql -u root -p
```

Example: echo_fds - pipes

Example: LAMP stack

Other uses

- performance: https://sysdigcloud.com/aws-storage-latency-sysdig-spectrogram/

- security: https://sysdigcloud.com/fishing-for-hackers/

resources

- https://github.com/draios/sysdig
- irc: #sysdig on freenode
- https://groups.google.com/forum/#!forum/sysdig

Questions?