Is your UNIX/SAS System Healthy?

Victor de Leon, Blue Shield of California, San Francisco, CA

ABSTRACT

Only a few tools exist to monitor the health of UNIX SAS. As SAS becomes more mission critical, new tools need to be developed to ensure systems availability.

INTRODUCTION

The most common threat to SAS availability is an “OUT OF SPACE” condition for the SASWORK disk mount point. This usually happens when a run-away query fills up the disk pool thereby causing abnormal termination of any program simultaneously writing to the pool.

USEFUL UNIX COMMANDS

These are fundamental commands that should be mastered to quickly understand the conditions in your environment.

df –lkh - lists mount points reporting free and used disk capacity

ls –alR – provides a full

top – lists processes currently running

tree – lists pid process tree (parents and children)

lsof - (not a standard utility) lists the filenames that pids are accessing

SOLUTIONS

There are several ways to address system stability:

1. Acquire more disk space – Though this may seem obvious, business cases must convincing enough to fund such acquisition. Actions tend to happen when consequences are clearly identified.
2. More physical mount points (/saswork1 , /saswork2 , . . .) – Grouping
3. Redirect saswork to larger areas – By using “–saswork” parameter, application specific programs can be made more stable if specific or dedicated pools can be assigned
4. Set up threshold warnings – Though these tend to only report static instances, it still provides some indication of a condition that warrants attention.
5. Develop home grown monitoring tools – NORAD

NORAD – Borrowing the acronym, this utility provides a similar function that warns of threat conditions. The necessary elements for predicting failure involve:

1. A directory snapshot – knowing the filenames and sizes and owner ids
2. Mount point capacity – this identifies the bounds of your resources
3. Sampling Time interval – Five minutes seems to be a reasonably effective and efficient time interval.

In short, if you summarize the disk space by owner id and compare the deltas every five minutes, you can calculate how long it would take to fill a disk. Flagging instances where failure could happen with two hours seems to provide an adequate window of resolution.

DEFENDER – This script allows a user to efficiently view the NORAD reports without overtaxing the system.

```
#!/usr/bin/sh

while [ 0 ] ; do
clear
xtime="`date '+%Y%m%d_%H%M%S'`"
echo "NORAD report time is ==> $xtime"
# cat the location of the report ie. /home/vdeleo01/NORAD/status.txt
cat /sasdata/it/apps/norad/prod/report/status.txt
echo " "
echo "          use CTRL+Z  to terminate this report"
sleep 300
done
```

q_work – lists what PIDs are using SASWORK (The top 10 pids using saswork space)

```
options nocenter ps=55 ls=200 ;

/* q_work – print the top 10 pids using saswork
written by Victor de Leon 4/1/2007
*/

options  nosymbolgen  nomlogic   nomacrogen;

%global tot_free mins reps m threat;

%macro dumpdir(dir,dest);


filename intxt pipe "ls -alR &dir";

data &dest (keep=owner dir t_fsize );

attrib yy format=$4.
user owner format=$15.
mmm format=$3.  tttt format=$5.  ttyy format=$5.  dd format=$2.  yy
format=$4.  ;

retain toe ;
retain dir ;
retain fsize t_fsize 0 owner;

infile intxt length=llong;
input @1 dataline $varying200. llong;


if llong = 0 then do;
    output;
    owner =' ';
    t_fsize=0;
    delete;
    end;

if substr(dataline,1,1) eq '/' then do;
    dir = scan(dataline,1,' :');
    delete;
    end;
```

```
if substr(dataline,1,1) eq 't' then delete;

if index(dataline,'Permission denied') > 0 then delete;



wline =dataline;

do x = 1 to 8;
    wline = left(substr(wline,index(wline,' ')));
end;

fn=wline ;


user  = scan(dataline,3,' ');
mmm   = scan(dataline,6,' ');
dd    = scan(dataline,7,' ');
ttyy  = scan(dataline,8,' ');

yy = year(today());
mi = month(today());

zz = index(ttyy,':');
z = length(compress(ttyy));

if  index(ttyy,':')= 0 then do;
yy = input(ttyy,4.);
tttt = '00:00';
end;
else do;
    if  mi <= 6 and
      mmm in ('Jul','Aug','Sep','Oct','Nov','Dec')
    then yy = yy - 1;
    tttt = ttyy;
end;

fsize = put(scan(dataline,5,' '),20.)/1000;

t_fsize = t_fsize + fsize ;

if fn = '.' then owner = user;

run;

%mend dumpdir;

run;

*-----------------------------------------------------------;
%dumpdir(/sas_work,allfiles);


data allfiles;
set allfiles;
attrib fn format=$100.;

fn = scan(dir,2,'/');

if substr(fn,1,3) ne 'SAS' then delete;

pstr = substr(fn,16,5);
```

```
dec = input(pstr,hex5.);

hstr = put(input(substr(pstr,1,1),hex1.),$binary8.)||
put(input(substr(pstr,2,1),hex1.),$binary8.)||
put(input(substr(pstr,3,1),hex1.),$binary8.)||
put(input(substr(pstr,4,1),hex1.),$binary8.)||
put(input(substr(pstr,5,1),hex1.),$binary8.);


y=0;
pid=0;


do x = 0 to 9 ;
    y=9-x;
    nib = input(substr(hstr,1+(x*4),4),binary4.)     ;
    b = nib*4**y;
    pid = pid + b;
end;


proc means data=allfiles nway noprint;
class pid owner;
id pstr;
var t_fsize;
output sum= n= out=allfiles;

proc sort data=allfiles;
by descending t_fsize;

proc print data=allfiles;
format t_fsize comma16. ;
var pid pstr owner  _freq_ t_fsize ;
sum t_fsize;
run;

proc sort data=allfiles;
by pid;
/*-------------------------------------------------------------------*/

filename intxt pipe "ps -fe ";


/* parse the text file for userid, process id and running command   */

data pslist (keep= pid xcmd uid);
attrib pid format=8.;
attrib uid format=$8.;
attrib xcmd format=$70.;
infile intxt length=llong firstobs=2;
retain x y 0 ;
input @1 dataline $varying200. llong;

if index(dataline,'<defunct>') > 0                or
   index(dataline,'-ksh') > 0                     or
   index(dataline,'xterm -ls -display') > 0
then delete;


if index(dataline,'?') > 0 then xcmd = substr(dataline,35);
else xcmd = substr(dataline,40);
```

```
xcmd = left(substr(xcmd,index(xcmd,':')+3));


uid = scan(dataline,1);
pid = input(scan(dataline,2),8.) ;


*xcmd = substr(dataline,x);

/* remove any processes that are associated with this query */

if scan(xcmd,1,' ') not in ('ps','grep','-ksh')
  then output;

run;

proc sort data=pslist;
by pid;


run;

/*-------------------------------------------------------------------*/

filename term_msg terminal ;

data active noact;
merge pslist (in=in1)
      allfiles (in=in2);
by pid;
retain xx 0 ;

file term_msg notitles;

if in1=0 and in2=1 then output noact;
if in1=1 and in2=1 then output active;

run;

/*-----------------------------------------------------------*/
filename getfree pipe "df -lk /sas_work";

data _null_;
infile getfree length=xlong firstobs=2;
input @1 freeline $varying200. xlong;
tot_free = scan(freeline,4,' ') ;
used     = scan(freeline,3,' ') ;

call symput('tot_free',tot_free);
stop;
run;


run;

/*-----------------------------------------------------------*/


proc sort data=noact;
by descending t_fsize;
```

```
proc sort data=active;
by descending t_fsize;

data _null_;
set  noact;
retain aa 0 ;

file term_msg notitles;

    if aa = 0 then do;
    put 'no active processes attached';
    aa = 1;
    end;

put pid @10 pstr @20 owner @30 t_fsize comma16.;
run;

/*-------------------------------------------------------*/

data _null_;
set active (obs=10);
retain aa 0 ;

file term_msg notitles;

    if aa = 0 then do;
    put 'top 10 active processes using sas_work';
    put 'PID      HEX4      USERID           Kbytes         %of sas_work';
    aa = 1;
    end;

pct = (t_fsize/&tot_free)*100;


put pid @10 pstr @20 owner @30 t_fsize comma16. @50 pct 6.2;
run;
```

CONCLUSIONS

Every organization needs to first develop a comprehensive storage policy. These clear objectives then set the requirements for the tools that will need to be built. Capacity planning and utilization reporting can then be driven from the resulting information.

FUTURE ENHANCEMENTS

Back in the lab, SKYNET (or Colossus) is being developed to automate the kill of offending processes. This is no simple matter because the algorithm rather complicated. Questions such as “Do you kill the job that is taking up the most SASWORK space or the job that is writing at the highest I/O transfer rate” tend to get very emotional.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name : Victor de Leon
Company : Blue Shield of California

Address	: 50 Beale
City state ZIP	: San Francisco, CA 94105
Work Phone	: 415-229-6317
Email	: victor.deleon@blueshieldca.com
	: vicdeleon@earthlink.net