Paper 099-2012

Using SAS® to Manage SAS® Work Area Usage

Urvir Palan, Barclays Technology Centre, Pune, India

ABSTRACT

SAS platform administrators always feel the pinch of not having enough space in the SAS work area, even though they plan for the worst. There are multiple approaches to tackle this problem, but one of the better ones is to initiate the alert mechanism as soon as you see the first sign of fire.

INTRODUCTION

This paper talks about the approach I am proposing to be used in our AIX environment to keep track of space usage and alerting the users via email. SAS® data step functions for reading and writing metadata are used for the email functionality.

APPROACH

All production analytics AIX platforms contains multiple work areas to accommodate never-ending demand from analyst and programmer user community. We have deployed load-balancing technique so that whenever a new SAS session is started the least used SAS work area is allocated to the SAS code. The problem starts when there are multiple users on a single work area and one of the SAS process turns rogue. This process loops and eats up all the available space in that SAS work area. All other users whose processes are utilizing same SAS work area are deprived of space and process gets abended. This results in frustration for all the other users of the same wok area, as they have to start from the scratch. Multiple techniques were used in past including providing guidance to users, defining standards of programming and mandating users to follow them etc.

I have devised a three-step preventive mechanism to address this issue.

First step: Alert the Guard

A simple Korn shell script is run from CRON scheduler every 10 minutes capturing the SAS work area usage, user ids and dumping the information into a text file. This file is iterated through a loop and free space in corresponding SAS work area is calculated. A SAS admin is informed via an email.

```
#!/bin/ksh

df -g /wload/XXX/app/SAS* | sort -n | uniq | grep -i SAS | awk '{ print $3 " " $7}' >/tmp/xxx_SASwork_usage

line_by_line(){
  line="$@" # get all args
  if [ $free -lt 60 ]
    then  echo "less space in $file. Current free space is $free GB" >>/tmp/xxx_SASwork
          echo " " >>/tmp/xxx_SASwork
          echo $file > /tmp/SASusers
    ls -la $file | grep -v root | grep -v xxxinst | grep -v xxxadm | awk {'print $3'} >> /tmp/SASusers
  fi
}

while read inputline
do
  free="$(echo $inputline | cut -d " " -f1)"
  file="$(echo $inputline | cut -d " " -f2)"
  line_by_line $inputline
done < /tmp/xxx_SASwork_usage
```

```
if [ -f /tmp/xxx_SASwork ]; then
cat /tmp/xxx_SASwork | mail -s "Less space in XXX SAS Work area; Please check" urvir.palan@example.com
```

Second Step: Find all the user ids and email address from Metadata

I am using SAS data step functions for reading metadata to get the userid, Name and Email address. An extensive exercise was taken to ensue that all SAS users are present in Metadata and their workload id and email address are updated.

```
/*** Step 00 Defining metadata options ***/

OPTIONS metaserver="xxxx.wload.example.com"
metaport=8561
metauser="xxxxsadm"
metapass="yyyyxxxx"
metarepository="Foundation";


/*** Step 01 Getting user name and email id from Metadata ***/

DATA user_id1(keep=Name email user_name);
        LENGTH uri_name Name Id user_id emailuri email $256;
        rc=1;
        rc2=0;
        n=1;
        DO UNTIL (rc<0 );
                uri_name='';
                Name='';
                Id='';
                emailuri='';
                email='';
                uri_userid='';
                rc=METADATA_GETNOBJ("omsobj:person?@Name ? " ",n,uri_name);
                rc2=METADATA_GETATTR(uri_name,"Id",Id);
                rc22=METADATA_GETATTR(uri_name,"Name",Name);
                rc3=METADATA_GETNASN("omsobj:Person?@Id='"!!Id!!"'","EmailAddresses",1,emailuri);
                rc4=METADATA_GETATTR(emailuri,"Address",email);
                n=n+1;
                IF rc>=0 THEN OUTPUT;

        END;
RUN;


/*** Step 02 Getting user id and email id from Metadata ***/

DATA user_id2(keep=user_id name);
        LENGTH uri_userid Name ID emailuri user_id  $256;
        rc=1;
        rc2=0;
        n=1;
        DO UNTIL (rc<0 );
                Name='';
                Id='';
                emailuri='';
                uri_userid='';
                rc=METADATA_GETNOBJ("omsobj:Login?@UserID ? ''",n,uri_userid);
                rc2=METADATA_GETATTR(uri_userid,"Id",Id);
                rc3=METADATA_GETATTR(uri_userid,"UserId",User_Id);
                rc4=METADATA_GETNASN("omsobj:Login?@Id='"!!Id!!"'","AssociatedIdentity",1,emailuri);
                rc5=METADATA_GETATTR(emailuri,"Name",Name);
```

```
                        n=n+1;
                        IF rc>=0 THEN OUTPUT;

            END;
RUN;

/*** Step 03 Joining tables to get all together ***/

PROC SQL;
            CREATE TABLE user_with_email AS
            SELECT a.name,a.email,b.user_id
            FROM user_id1 AS a,
            user_id2 AS b
            WHERE a.name LIKE b.name;
QUIT;


/*** Step 04 Finding users in affected SAS workarea ***/

DATA user_in_aff_SASwork;
            LENGTH id $11;
            INFILE '/tmp/SASusers' DSD MISSOVER TRUNCOVER;
            INPUT id $11.;
            id=strip(id);
            IF id ne '';
RUN;


/*** Step 05 Getting rid of duplicates ***/

PROC SORT
            DATA=user_in_aff_SASwork NODUP;
            BY id;
RUN;


/*** Step 06 Getting email ids of affected SAS work area users ***/

PROC SQL;
            CREATE TABLE xxx_aff_SASwork_users AS
            SELECT DISTINCT a.name,a.email,a.user_id,b.id
            FROM user_with_email AS a,user_in_aff_SASwork AS b
            WHERE a.user_id=b.id ;
quit;
```

Third step: Email users from the affected area to check their SAS usage

Users of affected area are informed via Email. Simple data step iteration is used to send emails to individual users. Below code uses Email Access Method.

```
/*** Step 07 Sending email to affected SAS work area users ***/

FILENAME reports EMAIL "xxx";
DATA work.mail;
            SET xxx_aff_SASwork_users;
            FILE reports;
            PUT '!EM_TO!' email;
            PUT '!EM_CC!' "urvir.palan@example.com";
            PUT '!EM_SUBJECT! Sapce Crunch in SAS Work area';
            /***** BEGIN TEXT IN BODY OF E-MAIL*****/
            PUT 'This is an automatic message please do not reply';
```

```
        PUT /;
        PUT 'We are having a Sapce crunch in SAS Work area on xxx. Can you please check your SAS usage and
        close the sessions which are not needed';
        PUT / 'User id: ' user_id;
        PUT '!EM_SEND!';
        PUT '!EM_NEWMSG!';
        PUT '!EM_ABORT!';
RUN;
```

CONCLUSION

It is never easy to tell a user that his/her code is turning rogue on the basis of errors in the log so it is better to alert them and let them decide if they are really getting the desired output from the code or it is really turning naughty.

REFERENCES

1. http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a002058232.htm
2. http://support.sas.com/documentation/cdl/en/lrmeta/63180/HTML/default/viewer.htm#p1k9zipe59ha2an1pq34gu143lay.htm

ACKNOWLEDGMENTS

I would like to thank Mukund Wadekar for his constant support and encouragement. I will also like to thank Shridhar Iyer and Vikas Jindal for their valuable inputs and support while writing this paper.

CONTACT INFORMATION

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

Name: Urvir Palan
Enterprise: Barclays Technology Centre India
Address: DLF Akruti Business Park, Block 4, Level 5
City, State ZIP: Pune, Maharashtra, India, 411057
Work Phone: +91-20-4155-2835
Fax: +91-20-4155-2500
E-mail: urvir.palan@barclays.com, urvirpalan@gmail.com
Web: www.barclays.com