SMF Recording
With
MVS Logger

IBM

Riaz Ahmad
IBM Washington Systems Center

Jeff McDonough
System z IT Architect
Many z/OS customers have begun to suffer from an overload of data provided by the System Management Facility (SMF). As the systems have grown ever faster and larger, doing more and more work generates more and more data from SMF used to measure, manage, and audit their z/OS environment. With the SMF data volumes increasing, the peak recording rates for this data is more and more often exceeding the capacity for recording and offloading. When this happens there is a significant risk of losing the very data they depend on. This project will validate a solution provided in z/OS Release 1.9, against sample customer data rates and volumes.

The support provided in z/OS Release 1.9 enables SMF to record via System Logger (LOGR) logstreams. CICS users and others utilizing OPERLOG and LOGREC are probably already familiar with LOGR logstreams. LOGR provides a general-purpose facility designed for the recording, retrieving, and managing of log data. There are a number of interesting new possibilities for managing SMF via this logstream support. It will now be possible to merge SMF data from multiple systems in a parallel sysplex in real time, instead of a post processing phase. It will also be possible to separate SMF data into separate streams so a user can isolate their most sensitive or most troublesome SMF record types from the easier to manage data. This separation into multiple streams also means data can be recorded to DASD via parallel processes. This can provide additional relief for successfully recording data at peak rates which can’t be achieved via the older SMF design. SMF logging can take advantage of the features LOGR provides, such as Coupling Facility structure caching, or simply DASD-only logstreams, depending on the user’s requirements.

It is even good news for users who currently don’t have an issue with the volume and rate of SMF data recording; that is, this new support is entirely optional! If a user has no pressing need to change their recording processes, they can continue to use the current support unchanged. In fact, SMF can dynamically switch between both methods with appropriate definitions. This makes it easier for users to try this new support and still be able to turn it off by issuing the SETSMF RECORDING command, if they choose. As always, the SMF configuration is still performed on an image-by-image basis, so (other than such advanced exploitation as merging logstreams mentioned above) the choice of SMF logging method can vary within the user environment – even within a sysplex!

This paper documents a sample customer situation and describes the process followed to implement the SMF-Logger environment. Once the new environment is implemented, some test runs will be run to validate the solution. Also, there will be a description of the flexibility this new support provides, as well as changes the user may have to consider as they make changes to their “tried and true” SMF processes.
Sample User Situation

The sample user situation for this project was taken from a real customer environment. This customer operates several large sysplexes, consisting of multiple, large, current generation System z9 Enterprise Class processors. The workload includes significant batch processing, but has multiple components of high volume, high speed, short duration interactive transactions. The interactive traffic is primarily CICS transactions accessing a DB/2 database backend. During the peak hours of this interactive workload the customer has experienced long intervals when SMF data was produced at a rate exceeding the capacity to offload. In reality, there are limits placed on the offload process by the customers SMF data collection design, which involves collecting data offloaded from the SMF “MANx” datasets into a single daily collection dataset. Each offload is appended to the end of the collection dataset (using DISP=MOD processing) and this effectively limits the offload processes to dump the “MANx” datasets one at a time, due to the enqueue on the daily collection dataset. Although other collection processes might be designed, this is the design this customer has developed and depended on for many years.

The analysis of the environment began with an examination of the volume and type of data being recorded on those days when SMF offload was being overwhelmed. As most DB2/CICS users would expect, the DB2 type 101 and 102 records and the CICS type 110 records dominated the data volume. The record count by type for each hour of the day was charted and then extended to take into the account the average record size for each record type. The results are summarized in the charts below.

![Record Count Chart]

Figure 1 Record count by type
These charts show the peak rates exceed 4 million records per hour and approach 14 GB of data per hour. It also demonstrates, in terms of data volume, the type 110 (CICS) and type 101 (DB2) records dwarf all the other types combined. An additional analysis of these specific types found the average record size for type 101 records was 1525 bytes, and the type 110 records averaged something over 30,000 bytes. These record sizes, counts, and volumes were used to build a test case for this study.

**The Test Environment**

The test environment utilized was four way Parallel Sysplex of z/OS V1R9 LPARs running on a z990 and z9. Neither of the images was constrained in terms of storage, processor power, or workload. There were two coupling facilities, also running on a z990, at CFCC Level 14. Each CF had approximately 2 GB of storage.

To simulate the SMF data load a small program was written to simply write SMF records with type and record lengths as specified. The intention was to generate load and volume similar to the sample user case, so focus was placed on one type of record at 1525 bytes and another at 30,000 bytes. The program has configurable pacing so the rate of SMF data generated could be matched to the sample user case. There was no attempt to suppress the usual system-generated SMF data, but its volume within the tests was considered negligible. In fact RMF data generated was used to produce processor utilization and coupling facility performance information for the test runs.
The Test Plan

1. Base case: generate enough data rate and volume to “overwhelm” an SMF “MANx” dataset configuration and have the “dump” process unable to keep up. (single LPAR)

2. Convert the test environment to SMF-LOGR configuration and run the same SMF data rate and volume against it and observe the results. (Single LPAR, DASD-only logging). Create a logstream “dump” process to create a sequential dataset for SMF post-processing.

3. Reconfigure the SMF-LOGR configuration to merge logstreams from multiple LPARs via the coupling facility structure and observe the results.

4. Test coupling facility structure rebuild processing while SMF processing is ongoing and observe the results.

5. Change the SMF logstreams to take advantage of the option to send different SMF records into separate logstreams and observe the results.

Figure 3  WSC zPlex configuration used for testing
**Step 1: The Base Case**

The sample customer environment utilized multiple SMF MANx datasets, each allocated at 1100 cylinders of 3390-3 space, scattered across multiple volumes but placed in trying to avoid having an active recording dataset on the same volume as filled dataset being dumped. Although many customers are now using Parallel Access Volumes to avoid IOS queue delays and thus may not be restricted with respect to multiple operations on a single volume. This configuration is a good base case for this study. The offload process was set-up to be triggered by the IEFU29 SMF Exit and this procedure would append to a sequential DASD dataset. Other customer may use GDGs for the offload purposes but they still suffer from the same serialization problems as the sample customer using DISP=MOD to append to the existing dataset. The definition of “overwhelming” SMF offload is the sample customer is approaching the point where SMF will stop recording because there are no empty MANx datasets available.

In the sample user case it is easy to see the peak data volume is 14GB/hour which breaks down to 3.8MB/sec. Since the division of data volume was near 50% type 101 and 50% 110 records, the SMF generator program was calibrated to create 2MB/sec of data each. In the first test runs it was determined the test environment was capable of sustaining this data rate and volume without getting overwhelmed. It was not possible to analyze at this point why the sample user environment suffered at this data rate but it was simply decided to increase the data rate until the test environment was overwhelmed. It was found a total data volume about 9.5MB/sec was sufficient to overwhelm the test environment. That is the calibration used as the next phase of the test was carried out.

**Step 2: SMF-LOGR (Single LPAR, DASD-Only Logstream)**

Planning for the SMF-LOGR environment included a discussion of how to modify the processes for the sample user. As noted above, the sample user offloads the SMF MANx datasets to sequential dataset as a daily collection point. At the end of the day, when the last MANx dataset has been offloaded the daily collection dataset is input to the daily reporting process, including an archive to tape. In an SMF-LOGR environment it was reasoned there was no advantage in periodic offloading of the logstream to a sequential dataset. There is no natural point for initiating an offload like there is when a MANx dataset fills, and the data requires approximately the same space residing in the logstream datasets as it does in the daily collection dataset. It was decided the input to the daily process and archiving should read directly from the logstream. This is accomplished using the new SMF utility IFASMFDDL which provides similar function as did the old IFASMFDP “dump” program, but processes logstreams as input. (The IFASMFDP utility has not been changed by this support). Users do need to take into consideration how much DASD will be needed in the LOGR offload pool to contain the collected SMF data and how long it should be retained. The deletion of the “expired” data from the logstream is provided by the RETPD= parameter in the logstream definition. The RETPD specifies
the number of days the logstream data will be retained after it has been written. You should be aware the RETPD checking is done whenever a new LOGR offload is started. If for some reason logging to the logstream is stopped (possibly by reverting to MANx logging) the data may remain in the logstream for much longer than expected from the RETPD. After planning the space requirements and naming conventions the next step was to configure the system for SMF logging to a DASD-only logstream.

```
//AHMAD2  JOB (67YC0000),CLASS=A,REGION=0M,
//    MSGCLASS=0,NOTIFY=AHMAD
//STEP1  EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=* 
//SYSAEND DD SYSOUT=* 
//SYSSN DD 
DATA TYPE(LOGR) REPORT(YES)
DEFINE LOGSTREAM NAME(IFASMF.SYSA.SYSTEM)
   DASDONLY(YES)
   STG_SIZE(500000)
   STG_DATACLASS(MVSLOGR)
   LS_DATACLASS(MVSLOGR)
   LS_SIZE(500000)
   HLQ(LOGGER)
   HIGHOFFLOAD(85)
   LOWOFFLOAD(0)
   AUTODELETE(YES)
   RETPD(1)
```

Figure 4: Definition of DASD-only logstream

```
ACTIVE   /*ACTIVE SMF RECORDING*/
BUFSIZMAX(0800M) /* MAXIMUM BUFFER SIZE */
DSNAME(&SYSNAME..MAN1,&SYSNAME..MAN2,&SYSNAME..MAN3,&SYSNAME..MAN4)
LISTDSN /* LIST DATA SET STATUS AT IPL*/
NOPROMPT /*DON'T PROMPT THE OPERATOR */
LSNAME(IFASMF.SYSA.SYSTEM,TYPE(0:255))
RECORDING(LOGSTREAM)
INTVAL(05) /* SMF GLOBAL RECORDING INTERVAL */
MEMLIMIT(3G) /* LIMIT 1G ABOVE THE BAR */
SYNCVAL(45) /* GLOBAL SYNC VALUE */
REC(PERM) /*TYPE 17 PERM RECORDS ONLY*/
MAXDORM(3000) /* WRITE AN IDLE BUFFER AFTER 30 MIN*/
STATUS(010000) /* WRITE SMF STATS AFTER 1 HOUR*/
JWIT(0030) /* 522 AFTER 30 MINUTES*/
SID(&SYSNAME(1:4)) /* USE SYSNAME AS SID */
SYS(NOTYPE(32,99),
   EXITS(IEFACTRT,IEFUTL,IEFUSI,IEFU83,IEFU84,IEFU29),
   INTERVAL(SMF,SYNC),NODETAIL)
/* WRITE ALL RECORDS EXCEPT TYPE 32 (TSO RECORDS), TAKE THE IEFACRT RT EXIT ONLY. */
SUBSYS(STC,EXITS(IEFACTRT,IEFUSI,IEFU83,IEFU84,IEFU29),
   INTERVAL(SMF,SYNC), TYPE(0,30,41,70:79,88:90,100:103,151,245))
SUBSYS(OMVS,NOEXITS,INTERVAL(SMF,SYNC),
   TYPE(0,30,70:79,88:90,103,245))
/* FOR STARTED TASKS, WRITE 30, RMF'S, TAKE ONLY IEFU29 AND ACTRT EXIT. NOTE: IEFU29 EXECUTES IN THE MASTER ASID WHICH IS A STC ADDRESS SPACE SO IEFU29 MUST BE ON FOR STC. USE ALL OTHER SYS PARMETERS AS A DEFAULT */
```
This case configured the simplest, most straightforward conversion of SMF MANx logging to LOGR; that is, a single DASD-only logstream. Again the SMF generator jobs were executed with the same data volume which overwhelmed the MANx logging, and carefully observed the system log. Other than an occasional message that LOGR had allocated another offload dataset there was really nothing to watch except the statistics from our data generator programs. The test runs continued for 12 min 57 sec, and generated a total of 7.4GB of data, or 9.5MB/sec.

After this data was generated the IFASMFDL utility was tested by reading all the data from the logstream and writing it to a sequential dataset as we might for archiving and post-processing. Reading the logstream and writing it to ordinary sequential DASD took 8 min 45 sec. To understand if the performance could be improved, the process was repeated using a sequential dataset striped across four volumes. The copy process took 6 min 16 sec for this case. This test was to retrieve the logged data and understand the performance characteristics, remembering that the logstreams are written to DASD and will have to be read by LOGR to provide to the IFASMFDL utility.

At this point the test was successful in demonstrating the SMF-LOGR configuration could exceed the maximum capability of MANx logging and the primary goal was achieved. However, there is significant additional flexibility provided by the SMF-LOGR feature and additional tests were performed to explore those capabilities.

**Step 3: Merge SMF Logstreams from Multiple LPARS**

One of the earliest capabilities provided by MVS System Logger was the ability to utilize the Coupling Facility to merge logstreams from multiple LPARs. Both OPERLOG and LOGREC recordings are examples many customers have used for the past several years. With the implementation of SMF-LOGR there are possibilities for logstreams which never existed before. Prior to this support, SMF was simply a single LPAR consideration – but now you can take advantage of LOGR to create a single stream of log data, merged in real-time via the coupling facility. The next task was to configure two LPARs to log their SMF data to the same logstream.

After the configuration was complete, the SMF generator jobs were executed but this time they were split across two LPARs of a four-way parallel sysplex. This generated approximately the same data volume for the same run duration, but simply created half of it on each system. Again there was nothing exciting to watch, and when the run completed the IFASMFDL ran against the single logstream and offloaded all the data from both systems in a single run.
Step 4: Test Coupling Facility Rebuild Processing

The next test was to perform coupling facility structure rebuild processing for merged logstreams and to observe its affects on SMF logging. Being able to rebuild coupling facility structures is an important aspect of maintaining high availability and recoverability in a parallel sysplex. The coupling facility structure was a fairly large percentage of the available CF storage (1.5GB of the 2.0GB total per CF) and the test was to see if SMF would suffer any ill effects from the rebuild processing. In the original set-up SMF was configured for 800MB (BUFZISEMAX=0800M) of internal storage buffer, as did the sample user, and the test was to observe if the specified value for BUFSIZEMAX would be sufficient to maintain regular processing during the rebuild operation.

Once the configuration was in place, the SMF generator jobs executed, and while they were in execution Coupling Facility structure rebuild was initiated to relocate the structure from one CF to the other. After the move was successful, structure rebuild was initiated again to relocate the structure back to the original CF according to the specified PREFLIST. The rebuilds completed in approximately 1 min 27 sec, including “directed offload” to insure data is retrieved from the originating structure.
SMF Logger

Figure 5: Coupling Facility structure rebuild
(These times are of course dependent on the configuration and will vary in other configurations). The SYSLOG was inspected for any indication of issue during the rebuild indicating a problem. A message would be expected if the buffer space utilization inside SMF had exceeded 25% of the 800MB allocation, but there were no messages issued in this interval.

**Step 5: Taking Advantage of New Logstream Options**

As mentioned above, the new SMF-LOGR support allows flexible options allowing configurations never possible before. The previous test demonstrated merging logstreams from multiple systems. The final test is to create multiple logstreams from the SMF data. This may provide users with a way to segregate data they would normally extract from SMF in a separate run, such as, extracting only DB2 data for analysis/auditing or RMF data for post-processing. These multiple logstreams are defined by SMF record types so you can send any grouping of SMF record types to a specific logstream. The default logstream then becomes the destination for all record types not specified in another logstream. Record types can even be repeated in different logstream definitions, causing the same records to be written to each logstream where it is referenced. This allows tremendous flexibility for user configuration, limited only by the user’s requirements. In addition to the flexibility it provides, the multiple logstreams also can improve the SMF-LOGR performance, because each logstream has its own DATASPACE and log writing task in SMF and in LOGR. This provides additional parallelism in processing the SMF data.

This test will define two logstreams, and segregate half of the data into one logstream and allow the other half to flow to the default logstream. This is another test providing nothing of interest to watch except to notice when LOGR has allocated the next offload dataset. However, it does demonstrate more of the flexibility provided by the SMF-LOGR support.
Setting up the SMF/Logger Environment

CFRM Policy

For logging SMF data to coupling facility logstreams, use IXCMIAPU administrative Data utility and update your CFRM Policy. An example is shown below:

```
//AHMAD1  JOB (67YC0000),CLASS=A,REGION=0M,
//      MSGCLASS=G,NOTIFY=AHMAD
//STEP01   EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=* 
//SYSABEND DD   SYSOUT=* 
//SYSIN    DD   *

DATA TYPE(CFRM) REPORT(NO)
DEFINE POLICY NAME(POLCF12) REPLACE(YES)

CF NAME(CF1)
   TYPE(002094)
   MFG(IBM)
   PLANT(02)
   SEQUENCE(000000023A6A)
   PARTITION(5)
   CPCID(00)
   DUMPSPACE(10000)

CF NAME(CF2)
   TYPE(002094)
   MFG(IBM)
   PLANT(02)
   SEQUENCE(000000023A6A)
   PARTITION(6)
   CPCID(00)
   DUMPSPACE(10000)

STRUCTURE NAME(LOGGER_SMF)
   SIZE(525056)
   REBUILDPERCENT(1)
   PREFLIST(CF1,CF2)
```

Figure 6: CFRM Policy

After the CFRM policy job has successfully completed, you would activate the newly created CFRM Policy in your sysplex.

```
SETXCF START,POLICY,POLNAME=polcynametype=CFRM
```
Logger Policy

The MVS Logger logstreams are defined using IXCMIAPU. It is recommended to use Logger STAGING DATA SETS initially when the SMF data recording to the logstream. By not specifying STG_SIZE, the Logger will create the default size which will be equal to the space occupied by the CF structure housing the logstream. Next determine the size of the logstream offload data set specified by the LS_SIZE statement. Also, determine how many logstreams to define to record all SMF record types. This sample job will define two logstreams:

- IFASMF.allsys.default
- IFASMF.allsys.data

```
//AHMAD2   JOB (67YCO000),CLASS=A,REGION=0M,
//     MSGCLASS=O,NOTIFY=AHMAD
//STEP1    EXEC PGM=IXCMIAPU
//SYSPRINT DD   SYSOUT=*  
//SYSABEND DD   SYSOUT=*  
//SYSIN    DD   *
DATA TYPE(LOGR) REPORT(YES)
DEFINE STRUCTURE NAME(LOGGER_SMF)
  LOGSNUM(2)
DEFINE LOGSTREAM NAME(IFASMF.allsys.data)
  STRUCTNAME(LOGGER_SMF)
  LOGGERDUPLEX(UNCOND)
  DULPEXMODE(UNCOND)
  STG_DUPLEX(YES)
  STG_DATACLAS(MVSLOGR)
  LS_DATACLAS(MVSLOGR)
  LS_SIZE(1500000)
  HLQ(LOGGER)
  HIGHTOFFLOAD(85)
  LOWOFFLOAD(0)
  AUTODELETE(YES)
  RETPD(1)
DEFINE LOGSTREAM NAME(IFASMF.allsys.default)
  STRUCTNAME(LOGGER_SMF)
  LOGGERDUPLEX(UNCOND)
  DULPEXMODE(UNCOND)
  LS_DATACLAS(MVSLOGR)
  STG_DUPLEX(YES)
  STG_DATACLAS(MVSLOGR)
  LS_SIZE(1500000)
  HLQ(LOGGER)
  HIGHTOFFLOAD(85)
  LOWOFFLOAD(0)
  AUTODELETE(YES)
  RETPD(1)
```

Figure 7: MVS Logger policy

The first logstream is the DEFAULT logstream and the second logstream is called named logstream. By specifying which SMF record types will be written to the named logstream,
record types not specified in the named logstream will be automatically directed to the DEFAULT logstream. You need to determine how to manage the offloaded data for each logstream created. There are a number of ways to manage this offloaded data but we chose the retention of data by using AUTODELETE and RETPD facility of MVS Logger. Note we did not specify AVGBUFSIZE and MAXBUFSIZE parameters on the STRUCTURE statement. The reason being the System Logger only uses the AVGBUFSIZE specified initially to set the entry-to-element ratio for the structure. After that, system logger will automatically manage the entry-to-element ratio for a structure dynamically, based on actual structure usage. We opted to take the default value of 65532 bytes for MAXBUFSIZE.

Customize SMFPRMxx

Next, customize the SMFPRMxx member in SYS1.PARMLIB. In this member you will specify the names of the logstreams you have created and the SMF recording method. With this new support there are two recording methods, DATASET or LOGSTREAM. You may consider taking your existing SMFPRMxx and modify it with new set of statements required for this support. There is a sample SMFPRMxx shown below:

```plaintext
ACTIVE                       /*ACTIVE SMF RECORDING*/
BUFSIZE(0800M)              /* MAXIMUM BUFFER SIZE */
DSNAME(&SYSNAME..MAN1,&SYSNAME..MAN2,&SYSNAME..MAN3,&SYSNAME..MAN4)
LISTDSN                     /* LIST DATA SET STATUS AT IPL*/
NOPROMPT                    /*DON'T PROMPT THE OPERATOR */
DEFAULTLSNAME(IFASMF.ALLSYS.DEFAULT)
LSNAME(IFASMF.ALLSYS.DATA,TYPE(100:255))
RECORDING(LOGSTREAM)

INTVAL(05)                  /* SMF GLOBAL RECORDING INTERVAL */
MEMLIMIT(3G)                /* LIMIT 1G ABOVE THE BAR */
SYNCVAL(45)                 /* GLOBAL SYNC VALUE */
REC(perm)                   /*TYPE 17 PERM RECORDS ONLY*/
MAXDORM(3000)               /* WRITE AN IDLE BUFFER AFTER 30 MIN*/
STATUS(010000)              /* WRITE SMF STATS AFTER 1 HOUR*/
JWT(0030)                   /* 522 AFTER 30 MINUTES*/
SID(&SYSNAME(1:4))          /* USE SYSNAME AS SID */
SYS( NOTYPE(32,99),
    EXITS(IEFACTRT,IEFUTL,IEFUSI,IEFUSI,IEFU83,IEFU84,IEFU29),
    INTERVAL(SMF,SYNC),NODETAIL)
/* WRITE ALL RECORDS EXCEPT TYPE 32 (TSO RECORDS), TAKE THE IEFACTRT EXIT ONLY. */
SUBSYS(STC,EXITS(IEFACTRT,IEFUSI,IEFU83,IEFU84,IEFU29),
    INTERVAL(SMF,SYNC),
    TYPE(0,30,41,70:79,88:90,100:103,151,245))
SUBSYS(OMVS,NOEXITS,INTERVAL(SMF,SYNC),
    TYPE(0,30,70:79,88:90,103,245))
/* FOR STARTED TASKS, WRITE 30, RMF'S, TAKE ONLY IEFU29 AND ACTRT EXIT. NOTE: IEFU29 EXECUTES IN THE MASTER ASID WHICH IS A STC ADDRESS SPACE SO IEFU29 MUST BE ON FOR STC. USE ALL OTHER SYS PARMETERS AS A DEFAULT */
```

Figure 8: SMFPRMxx member of SYS1.PARMLIB
Activate the newly created SMFPRMxx member by issuing SET SMF=xx command

SET SMF=RA
IEE252I MEMBER SMFPRMRA FOUND IN SYS1.PARMLIB
IEE974I 13.03.01 SMF DATA SETS 139
NAME VOLSER SIZE(BLKS) %FULL STATUS
P-SYS.B.MANW SMF011 33000 34 ACTIVE
S-SYS.B.MANX SMF012 33000 0 ALTERNATE
S-SYS.B.MANY SMF011 33000 0 ALTERNATE
S-SYS.B.MANZ SMF012 33000 0 ALTERNATE
IXC582I STRUCTURE LOGGER_SMF ALLOCATED BY SIZE/RATIOS. 140
PHYSICAL STRUCTURE VERSION: C1522A03 7F43A99C
STRUCTURE TYPE: LIST
CFNAME: CF1
ALLOCATION SIZE: 1500160 K
POLICY SIZE: 1500000 K
POLICY INITSIZE: 0 K
POLICY MINSIZE: 0 K
IXLCONN STRSIZE: 0 K
ENTRY COUNT: 43594
ELEMENT COUNT: 2833700
ENTRY COUNT: 43594
ELEMENT COUNT: 2833700
ENTRY:ELEMENT RATIO: 1 : 65
ALLOCATE SIZE IS WITHIN CFRM POLICY DEFINITIONS
IXL014I IXLCONN REQUEST FOR STRUCTURE LOGGER_SMF 141
WAS SUCCESSFUL. JOBNAME: IXGLOGR ASID: 0016
CONNECTOR NAME: IXGLOGR_SYSB CFNAME: CF1
IXL015I STRUCTURE ALLOCATION INFORMATION FOR 142
STRUCTURE LOGGER_SMF, CONNECTOR NAME IXGLOGR_SYSB
CFNAME ALLOCATION STATUS/FAILURE REASON
-------- ---------------------------------
CF1 STRUCTURE ALLOCATED AC005800
CF2 PREFERRED CF ALREADY SELECTED AC005800
IEF196I IGD101I SMS ALLOCATED TO DDNAME (SYS56112)
IEF196I DSN (LOGGER.IFASMF.SYB.SYSTEM.SYB )
IEF196I STORCLAS (LOGRSC) MGMTCLAS ( ) DATACLAS (MVSLOGR)
IEF196I VOL SER NOS FOR DATA COMPONENT= ZLOG11
IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS56113
IEF196I IGD101I SMS ALLOCATED TO DDNAME (SYS56114)
IEF196I DSN (LOGGER.IFASMF.SYB.DEFAULT.SYB )
IEF196I STORCLAS (LOGRSC) MGMTCLAS ( ) DATACLAS (MVSLOGR)
IEF196I VOL SER NOS FOR DATA COMPONENT= ZLOG1C
IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS56115
IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS56115
IEF196I IGD101I SMS ALLOCATED TO DDNAME (SYS56114)
IEF196I IGD101I SMS ALLOCATED TO DDNAME (SYS56114)
IEF196I DSN (LOGGER.IFASMF.SYB.DEFAULT.SYB )
IEF196I STORCLAS (LOGRSC) MGMTCLAS ( ) DATACLAS (MVSLOGR)
IEF196I VOL SER NOS FOR DATA COMPONENT= ZLOG1C
IEF196I VOL SER NOS FOR DATA COMPONENT= ZLOG1C
IEF196I VOL SER NOS FOR DATA COMPONENT= ZLOG1C
IFA716I THERE ARE NO RECORDS FOR DEFAULT LOGSTREAM TO COLLECT 153
DEFAULTLSNAME(IFASMF.SYB.DEFAULT) PARAMETER IS IGNORED.
IFA711I LOGSTREAM PARAMETERS ARE IN EFFECT
IEE968I NOTIFICATION OF SUBSYS OMVS FAILED -
IEE968I SUBSYSTEM IS NOT OPERATIONAL
IEE536I SMF VALUE RA NOW IN EFFECT
START DUMPSMF,DSNAME=SYSB.MANW
IEF196I has issued command 'START DUMPSMF,DSNAME=SYSB.MANW'
IEF196I has issued command 'START DUMPSMF,DSNAME=SYSB.MANW'
$HASP100 DUMPSMF ON STCINRDR
IEF695I START DUMPSMF WITH JOBNAME DUMPSMF IS ASSIGNED TO USER
STCRACF, GROUP SYS1
$HASP373 DUMPSMF STARTED
IEF403I DUMPSMF - STARTED - TIME=13.03.06
Display the SMF by issuing `D SMF` to see the logstream name and the connect status.

```
D SMF
IF4714I 13.04.58 SMF STATUS 184
  LOGSTREAM NAME               BUFFERS        STATUS
  A-IFASMF.ALLSYS.DATA              4984       CONNECTED
  A-IFASMF.ALLSYS.DEFAULT           7571       CONNECTED
```

Display the CF structure to get the detailed information about the CF structure.

```
D XCF,STR,STRNM=LOGGER_SMF
IXC360I  13.09.59  DISPLAY XCF 189
STRNAME: LOGGER_SMF
  STATUS: ALLOCATED
  EVENT MANAGEMENT: MESSAGE-BASED
  TYPE: LIST
  POLICY INFORMATION:
  POLICY SIZE : 1500000 K
  POLICY INITSIZE: N/A
  POLICY MINSIZE : 0 K
  FULLTHRESHOLD : 80
  ALLOWAUTOALT : NO
  REBUILD PERCENT: 1
  DUPLEX : DISABLED
  ALLOWREALLOCATE: YES
  PREFERENCE LIST: CF1   CF2
  ENFORCEORDER : NO
  EXCLUSION LIST IS EMPTY

ACTIVE STRUCTURE
-----------------
  ALLOCATION TIME: 10/09/2007 13:03:01
  CFNAME         : CF1
  COUPLING FACILITY: 002084.IBM.02.000000023A6A
  PARTITION: 05   CPCID: 00
  ACTUAL SIZE    : 1500160 K
  STORAGE INCREMENT SIZE: 256 K
  USAGE INFO      TOTAL     CHANGED    %
  ENTRIES:       43594         511    1
  ELEMENTS:    2833700       59697    2
  PHYSICAL VERSION: C1522A03 7F43A99C
  LOGICAL VERSION: C1522A03 7F43A99C
  SYSTEM-MANAGED PROCESS LEVEL: 8
  DISPOSITION : DELETE
  ACCESS TIME : 0
  MAX CONNECTIONS: 32
  # CONNECTIONS : 1

  CONNECTION NAME   ID  VERSION   SYSNAME   JORNAME   ASID   STATE
                   ---- -------- -------- -------- ---- ----------------
  IXGLOGR_SYSB     01 00010001 SYSB     IXGLOGR  0016 ACTIVE

DIAGNOSTIC INFORMATION:  STRNUM: 00000041 STRSEQ: 00000002
  MANAGER SYSTEM ID: 02000497
  NAME/MGR #QUEUED 1STQESN LASTQESN CMPESN NOTIFYESN
  SYSB 00000000 00000000 00000000 00000002 00000002
  MGR SYS 00000000 00000000 00000000 00000002 00000000

EVENT MANAGEMENT: MESSAGE-BASED
  MANAGER SYSTEM NAME:  SYSB
```
Additional function is also available in the new IFASMFDL utility. This utility reads logstreams instead of MANx datasets. IFASMDL provides for definition of multiple output datasets in a single pass, so a single logstream can be read to create sequential post-processing datasets, for example, to separate datasets for CICS, DB2, RMF, etc.

```
//AHMAD1   JOB (????,????),'AHMAD-WSC',MSGCLASS=O,CLASS=A,
//      NOTIFY=AHMAD,REGION=0M TYPRUN=HOLD
//STEP1    EXEC  PGM=IFASMFDL
//OUTDD1   DD DSN=JBMCDON.SMFT99.ACUM,DISP=SHR
//OUTDD2   DD DSN=JBMCDON.SMFT255.ACUM,DISP=SHR
//SYSPRINT DD SYSOUT=*  
//SYSIN    DD *
LSNAME(IFASMF.ALLSYS.DATA,OPTIONS(ALL))
OUTDD(OUTDD1,TYPE(0:99),START(0000),END(2400))
OUTDD(OUTDD2,TYPE(100:255),START(0000),END(2400))
DATE(2007282,2007282)
```

Figure 9: IFASMFDL JOB for post processing

**SMF SWITCH Processing**

When the SMF data set which is currently being recorded on becomes full, SMF automatically closes the full data set, making it available for dumping. SMF locates the new data set to open by starting at the top of the list of data sets specified on the DSNAME parameter of the SMFPRMxx member and looking for the first completely empty data set.

If the first available data set is not completely empty, SMF will begin to store the records in its buffers, even though there might be enough room in the data set for the records it is trying to write, and even though other SMF data sets might be completely empty.

Full means the record which SMF is currently preparing to write out will not fit into the space left on the current SMF data set. It is possible a data set might become "full" when it is less than 100% filled.

To prepare an SMF data set for dumping before it becomes full, the operator uses the SWITCH SMF (I SMF) command. The SMF Exit IEFU29 is driven after the SWITCH command completes and a command may be issued from this exit to start the SMF dump job. When switching the SMF data sets, an inactive data set cannot become active unless it is empty. Therefore, before issuing the SWITCH command, the operator should use the DISPLAY SMF command to verify there is at least one alternate data set. If the operator does not make this check, data might be lost.

With the logstream implementation, a new SMF exit IEFU29L will be driven when a SWITCH command is issued in the new environment and all the SMF data is flushed from the structure to an offload data set.
Migration and Coexistence Considerations

SMF MANx data sets can still be defined in the SMFPRMxx SYS1.PARMLIB member. As a matter of fact, this is a good idea, as it allows fall back to SMF datasets if the exploitation of log streams encounters unexpected problems.

While you can identify both SMF data sets and log streams in the SMFPRMxx member, only one recording mechanism can be in use at a time. The MVS command SETSMF RECORDING(DATASET|LOGSTREAM) allows you to easily switch between recording types. This will facilitate both exploitation of the new function as well as fall back.

It should be noted a robust exploitation of the new recording mechanism may require “business process updates”. That is, you may want to change the end location of your data to keep the data completely partitioned throughout, in other words, from time of creation to time of archival or deletion. As an example, one would not want to change the location of billing data without concurrence of the users of the data.

In terms of coexistence, either DASDONLY or CF-based logstreams can be used. If multiple systems record to the same logstream, you need to have unique SMF IDs (SID) for each system in the sysplex.

Conclusion

Although the tests were not able to demonstrate the same limitations of SMF logging as the sample user case at the same data rate, the tests were able to increase the data rate in our environment until the same limitations were encountered. After enabling the SMF-Logger feature and establishing a DASD-only log stream the tests demonstrated data volume could be sustained at a rate which could not be sustained with native SMF logging.

Additional options enabled by the SMF-LOGR feature were then explored, taking advantage of the coupling facility to merge logstreams as well as the options to separate them. The ability to create multiple output dataset from the IFASMFDL logstream dump program was also demonstrated.

Overall, the SMF –Logger support provides the capability expected and more than just improved performance for SMF logging.