Bit Bucket x'40'

Tom Conley, pinnacle@pinncons.com
Mike Shorkend, mike@shorkend.com
James Lund, james-lund@tamu.edu
Ed Jaffe, edjaffe@phoenixsoftware.com

SHARE
Columbus
Summer 2022
31061

It's a Small zFS, After All...

(Tom Conley)

zFS Grows, and Grows, and Grows…

- Getting a lot of USS filesystem pool space issues
- zFS just grows constantly without freeing old space
- zfsadm shrink to the rescue, right?
- Not exactly…

zfsadm shrink (not exactly the Easy button)

- 1st try, zfsadm shrink -aggregate ZFS.FILESYS.RO
  - `IOEZ00240E IOEZADM:  Missing required parameter '-size'.`
  - zfsadm shrink requires you to specify size of zFS after shrink
- 2nd try, zfsadm shrink -aggregate ZFS.FILESYS.RO -size 1
  - Decided to try "1" for size, thinking that would release all freespace
  - `IOEZ00872E Error shrinking aggregate ZFS.FILESYS.RO, error code=141 reason code=EF176C22.`
  - TSO BPXMTEXT EF176C22 shows:

    ```
    Description: Aggregate could not be shrunk. It is mounted in
    R/O mode.

    Action: The aggregate must be mounted read-write to be shrunk.
    ```
  - zfsadm shrink forces you to remount RO filesystems to RW
  - You'll also have to remount after shrink to get back to RO

zfsadm shrink (not exactly the Easy button)

- 3rd try zfsadm shrink -aggregate ZFS.FILESYS.RW -size 1
  - IOEZ00872E Error shrinking aggregate ZFS.FILESYS.RW, error code=121 reason code=EF176C4A.
  - TSO BPXMTEXT EF176C4A shows:

    Description: The new size is smaller than the minimum aggregate size.

    Action: Use the zfsadm fsinfo command to determine the size of the aggregate, the amount of free space, and the amount of space in use. Try the command again with a new size that is larger than the amount of space currently being used.
  - So much for thinking "1" would release all the freespace
  - zfsadm shrink requires you to calculate *FINAL* size of zFS shrink

Nobody Told Me There'd be Math

- So I run fsinfo on ZFS.FILESYS.RW (relevant fields italicized and underlined: 770720K, 16396):

```
File System Name: ZFS.FILESYS.RW

   <snip>
   Size:                    770720K          Free 8K Blocks:          16396
   <snip>
```

- Calculate size parm for zFS after shrink
  - Multiply Free 8K Blocks by 8 (16396 * 8 = 131168)
  - Subtract that result from Size (770720 - 131168 = 639552)
  - Use 639552 in zfsadm shrink command

Nobody Told Me There'd be Math

- 4th try, zfsadm shrink -aggregate ZFS.FILESYS.RW -size 639552
  - IOEZ00873I Aggregate ZFS.FILESYS.RW successfully shrunk.
  - Finally, we have shrunk the zFS, but…
- FSINFO shows that not everything was shrunk, even after all that

```
File System Name: ZFS.FILESYS.RW

    *** owner information ***
    Owner:                  S0W1            Converttov5:            OFF,n/a
    Size:                   640576K         Free 8K Blocks:         136
    <snip>
```

- Testing shows multiple iterations required to release all freespace

Nobody Told Me There'd be Math

- After messing with fsinfo, discovered aggrinfo
- zfsadm aggrinfo -aggregate PINNACLE.JVA710.ZFS
  - `PINNACLE.JVA710.ZFS (R/W COMP): 162671 K free out of total 626400`
- 626400 - 126671 = 463729
- zfsadm shrink PINNACLE.JVA710.ZFS -size 463729
  - `PINNACLE.JVA710.ZFS (R/W COMP): 2112 K free out of total 465784`
- zfsadm shrink PINNACLE.JVA710.ZFS -size 463672
  - `PINNACLE.JVA710.ZFS (R/W COMP): 2048 K free out of total 465720`
  - `PINNACLE.JVA710.ZFS (R/W COMP): 1024 K free out of total 464696`
  - `PINNACLE.JVA710.ZFS (R/W COMP): 16 K free out of total 463672`
- zfsadm shrink PINNACLE.JVA710.ZFS -size 463656
  - `PINNACLE.JVA710.ZFS (R/W COMP): 0 K free out of total 463656`
- So 5, (only 5) runs later, we've finally released all zFS freespace

This One Time, at Band Camp…

- I ran zfsadm shrink against Java filesystem
- On 2nd iteration of zfsadm shrink, it failed outright
  - PINNACLE.JAVA.V80.DIR (R/W COMP): 1312 K free out of total 525904
  - zfsadm shrink -aggregate PINNACLE.JAVA.V80AP.DIR -size 524592
  - IOEZ00872E Error shrinking aggregate PINNACLE.JAVA.V80.DIR, error code=119 reason code=EF176C47.
- TSO BPXMTEXT EF176C47 says:
  - Description: Unknown error during shrink operation.
    Action: Check the system log for message IOEZ00904E to get additional error information. If the problem cannot be resolved, contact the service representative.
- IOEZ00904E says:
  - IOEZ00904E Aggregate PINNACLE.JAVA.V80.DIR shrink halted with errorcode=119. Moved 27 blocks and 0 fragments.

This One Time, at Band Camp…

- The Fine Manual says:
  - If the error code is 119 (EFBIG) or 133 (ENOSPC) and active increase was not allowed, try the shrink command with a larger new size. If active increase was allowed and you still want to shrink the aggregate, remove unwanted files and directories from the aggregate and try the shrink command again. For other error codes, try to determine the reason for the failure and reissue the shrink command. If the problem persists, contact your service representative.
- So I opened a case, and after much wrangling, survey says!
  - IBM: Run zfsadm shrink again, add the -noai parameter
  - Me: -noai? <in my best Astro or Scooby Doo voice>
- -noai says:
  - The new total size is not to be increased if more space is needed. For more information about active increase, see Usage notes for zfsadm shrink.

This One Time, at Band Camp…

- The Usage Notes for zfsadm shrink say:
  - Most of the shrink operation allows other applications to access file and directory blocks during the shrink operation. This might cause additional blocks to be allocated. If this allocation causes more space to be needed in the aggregate than the new total size specified in -size, zFS will actively increase the new total size. The shrink command ends with an error if the size is actively increased back to the original size of the aggregate. You can prevent active increase by specifying -noai. If -noai is specified, and an active increase is needed, the shrink command ends with an error.
- Butbutbut…
  - Filesystem was mounted at /u/ibmuser/zfstest, and I wasn't using it
  - Made no sense to me, but I did it anyway, and VIOLA!!
    - zfsadm shrink -aggregate PINNACLE.JAVA.V80.DIR -size 524592 -noai;
    - IOEZ00873I Aggregate PINNACLE.JAVA.V80.DIR successfully shrunk.
    - PINNACLE.JAVA.V80.DIR (R/W COMP): 0 K free out of total 524592

IBM on the Case

- I opened case with IBM about IOEZ00904E error
  - Also mentioned multiple iteration issue
- After much back and forth, here's the solution
- Use –noai switch if you want to release all freespace in one run
- Doc implies failure due to filesystem being updated "active increase"
- My filesystem was not being updated
- zfsadm shrink without –noai reserves 128 8K pages of space
  - Juuuuust in case you have an "active increase"
  - zfsadm shrink without –noai WILL NOT RELEASE 1024K!!
- I was told this behavior was documented, can't find it anywhere
- Also doesn't explain why multiple iterations find more freespace
- Stay tuned, there's an APAR or z/OS Idea in here somewhere…

You'll Get 0-7K and LIKE IT!!

- I tried to shrink zFS that reported 6K freespace:
  - PINNACLE.MESSEDUP.ZFS (R/W COMP): 6 K free out of total 2139816
  - zfsadm shrink -aggregate PINNACLE.MESSEDUP.ZFS -size 2139810;
  - IOEZ00872E Error shrinking aggregate PINNACLE.MESSEDUP.ZFS, error code=119 reason code=EF176C49.
- TSO BPXMTEXT EF176C49 says:
  - Description: The space used in the aggregate is larger than the specified new size
  - Action: Use the zfsadm fsinfo command to determine the size of the aggregate, the amount of free space, and the amount of space in use. Try the command again with a new size that is larger than the amount of space currently being used.
- The Fine Manual documents that the smallest increment that can be freed by zfsadm shrink is 8K
- If you see freespace of 0-7K in aggrinfo, you are DONESKI!

zfsadm shrink (not exactly the Easy button)

- zfsadm shrink is far too difficult to use
  - Filesystem must be mounted R/W
  - -size must be manually calculated
  - Only one zFS can be processed per zfsadm shrink command
- Tom, I have hundreds of zFS filesystems, what do I do?
- aggrinfo will give you all mounted filesystems

```
zfsadm aggrinfo
IOEZ00370I A total of 46 aggregates are attached.
<snip>
ZFS.ROCKET (R/W COMP): 1086943 K free out of total 1728000
HUH100.ZFS (R/W COMP): 28079 K free out of total 106560
IEL530.ZFS (R/O COMP): 863 K free out of total 2160
<snip>
```

zfsadm shrink (not exactly the Easy button)

- I'm writing a Rexx exec to handle releasing zFS filesystem space
  - Parse aggrinfo for mount, total, and free
  - Remount R/O filesystem as R/W if necessary
  - Calculate -size and perform zfsadm shrink
  - Remount R/W filesystem as R/O if necessary
- How do you handle zFS filesystems that aren't mounted?
  - Perhaps do catalog search to find cataloged but unmounted zFS's
  - Create temp mount point
  - Mount zFS filesystem at temp mount point and perform zfsadm shrink
  - Unmount, lather, rinse, repeat

z/OS Ideas for zfsadm shrink

- ZOS-I-3390 - zfsadm shrink should provide means to mount R/O filesystem R/W, then switch back to R/O after zfsadm shrink completes
- ZOS-I-3391 - zfsadm shrink should provide a means to automatically specify -size without running aggrinfo or calculating from fsinfo
- ZOS-I-3392 - zfsadm shrink should support a wildcard for -aggregate
- ZOS-I-3393 - zfsadm shrink should provide the ability to mount an unmounted zFS filesystem
- Benefits of implementing these Ideas
  - Drastically reduce amount of effort required to release zFS space
  - Possible to release freespace for all zFS filesystems in one command
- Please vote for these Ideas if you agree with them

Acknowledgements (Knowing and Unknowing)

- Walter Manrique, IBM
- Marna Walle, IBM

Two Hidden RMF Goodies

ERBSCAN/ERBSHOW

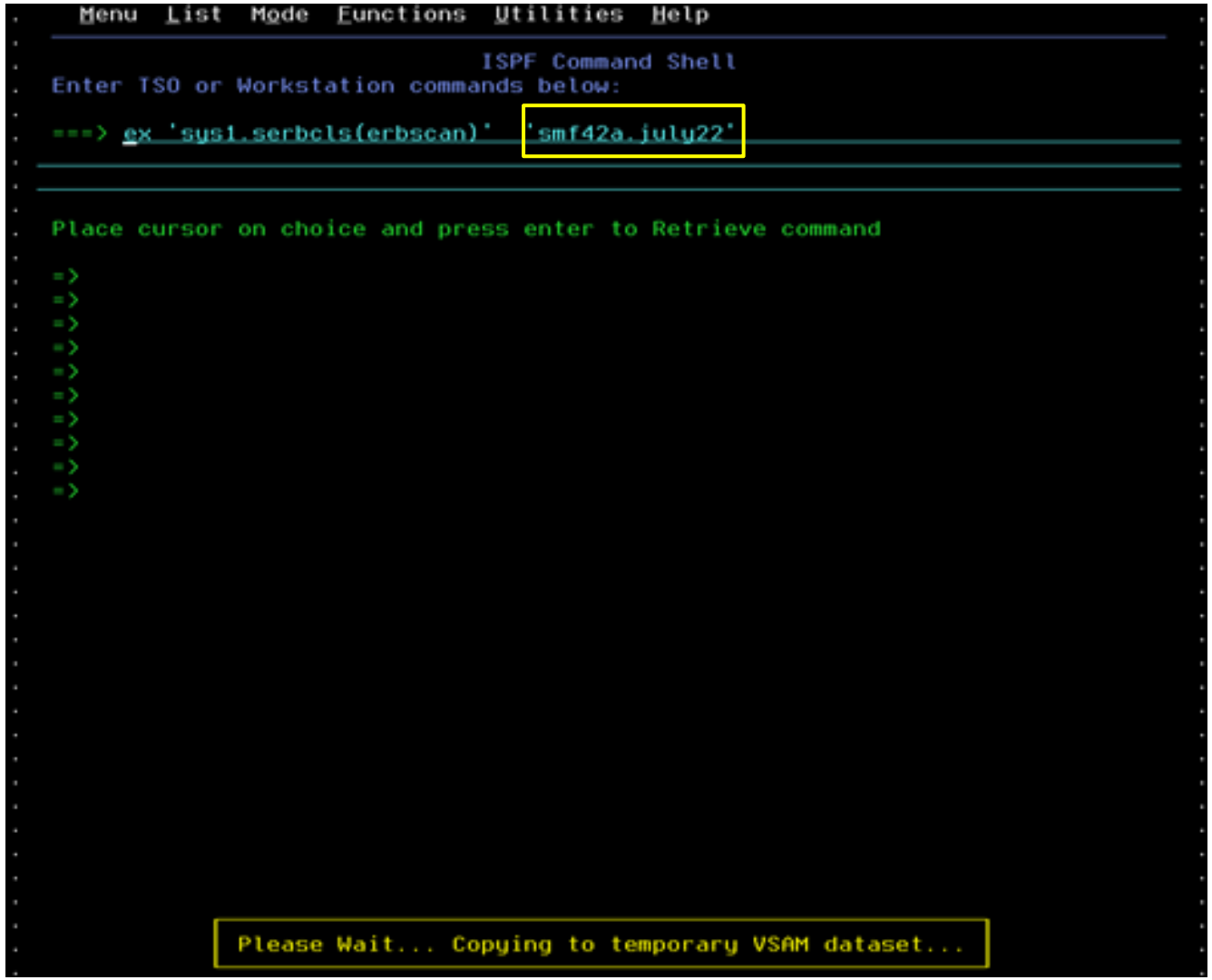
RMF III Batch Reporting

(Mike Shorkend)

ERBSCAN/ERBSHOW

- Allows you to read RMF SMF records within ISPF
- ERBSCAN is a rexx program that takes a dataset with SMF records as a parameter
- After analyzing the SMF data, it places you in an edit session with a list of the records it found.
- You can then use the ERBSHOW provided edit macro to drill down a specific record
- These two rexx programs can be found in the SERBCLS dataset which should be concatenated to your SYSEXEC card

Invoking ERBSCAN



- Create a small input dataset
- Does not work great with TSO PROFILE NOPREF

Invoking ERBSCAN

```
VIEW       SYS22229.T153205.RA000.S03.R0308112                                  Columns 00001 00124
Command ===> _                                                                  Scroll ===> PAGE
****** ***************************************************** Top of Data ********************************************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 1z/OS V2R4 RMF ERBMFSCN Version 9 (30 Apr 2012) - SCAN SMF dataset
000002
000003  SMF dataset characteristics:
000004  RECFM    : VBS
000005  LRECL    : 32760
000006  BLKSIZE  : 32760
000007  DATASETS : 1
000008  DSNAME(S): S03.SMF99.JUNE22
000009  DATE/TIME: 2022 August 17     15:32:05.381
000010
000011
000012 1Rec-Num Type    RecLn SMFDate  SMFTime  RMFDate  RMFTime  Int-Len   SMFId/Vers Samples SRCL End-Token        S-C Sysplex  S
000013  ------- ------- ----- ---------- -------- -------- -------- --------- ---------- ------- ---- ---------------- --- -------- -
000014        1 002        18 2022.170 11:50:01                               SYSD
000015        2 099.001 10386 2022.168 12:00:01                     SP7.2.4   SYSI                                              S
000016        3 099.002   896 2022.168 12:00:01 DBMSTEST          SP7.2.4   SYSI                                              S
000017        4 099.002   896 2022.168 12:00:01 DB2DDCOL          SP7.2.4   SYSI                                              S
000018        5 099.002  1920 2022.168 12:00:01 DB2PROD           SP7.2.4   SYSI                                              S
000019        6 099.002   896 2022.168 12:00:01 DDFWORK           SP7.2.4   SYSI                                              S
000020        7 099.002  1076 2022.168 12:00:01 DDFWORKP          SP7.2.4   SYSI                                              S
000021        8 099.003   356 2022.168 12:00:01 DDFWORKP          SP7.2.4   SYSI                                              S
000022        9 099.002  2024 2022.168 12:00:01 HOTBATCH         SP7.2.4   SYSI                                              S
000023       10 099.003   628 2022.168 12:00:01 HOTBATCH         SP7.2.4   SYSI                                              S
000024       11 099.002   896 2022.168 12:00:01 ISPW             SP7.2.4   SYSI                                              S
000025       12 099.002  1672 2022.168 12:00:01 NEWWORK          SP7.2.4   SYSI                                              S
000026       13 099.002   896 2022.168 12:00:01 NEWWORKV         SP7.2.4   SYSI                                              S
000027       14 099.003   260 2022.168 12:00:01 NEWWORKV         SP7.2.4   SYSI                                              S
000028       15 099.002  1512 2022.168 12:00:01 ONLPRD           SP7.2.4   SYSI                                              S
000029       16 099.003   492 2022.168 12:00:01 ONLPRD           SP7.2.4   SYSI                                              S
000030       17 099.002   896 2022.168 12:00:01 ONLPRH           SP7.2.4   SYSI                                              S
000031       18 099.002                                                                                                       S
000032       19 099.003  Enter 'ERBSHOW <recnum>' in the EDIT command line to analyze the specified                           S
000033       20 099.002  record in detail.                                                                                    S
000034       21 099.003                                                                                                       S
```

Using ERBSHOW

```
VIEW       SYS22229.T153528.RA000.S03.R0308255                                  Columns 00001 00124
Command ===> erbshow 10_                                                        Scroll ===> PAGE
****** ********************************************************* Top of Data ***********************************************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 1z/OS V2R4 RMF ERBMFSCN Version 9 (30 Apr 2012) - SCAN SMF dataset
000002
000003  SMF dataset characteristics:
000004  RECFM    : VBS
000005  LRECL    : 32760
000006  BLKSIZE  : 32760
000007  DATASETS : 1
000008  DSNAME(S): S03.SMF99.JUNE22
000009  DATE/TIME: 2022 August 17     15:35:28.381
000010
000011
000012 1Rec-Num Type    RecLn SMFDate  SMFTime  RMFDate  RMFTime  Int-Len   SMFId/Vers Samples SRCL End-Token        S-C Sysplex  S
000013  ------- ------- ----- ---------- -------- ---------- -------- --------- ------- ---- ---------------- --- -------- -
000014        1 002        18 2022.170 11:50:01                              SYSD
000015        2 099.001 10386 2022.168 12:00:01                    SP7.2.4   SYSI                                              $
000016        3 099.002   896 2022.168 12:00:01 DBMSTEST         SP7.2.4   SYSI                                              $
000017        4 099.002   896 2022.168 12:00:01 DB2DDCOL         SP7.2.4   SYSI                                              $
000018        5 099.002  1920 2022.168 12:00:01 DB2PROD          SP7.2.4   SYSI                                              $
000019        6 099.002   896 2022.168 12:00:01 DDFWORK          SP7.2.4   SYSI                                              $
000020        7 099.002  1076 2022.168 12:00:01 DDFWORKP         SP7.2.4   SYSI                                              $
000021        8 099.003   356 2022.168 12:00:01 DDFWORKP         SP7.2.4   SYSI                                              $
000022        9 099.002  2024 2022.168 12:00:01 HOTBATCH         SP7.2.4   SYSI                                              $
000023       10 099.003   628 2022.168 12:00:01 HOTBATCH         SP7.2.4   SYSI                                              $
000024       11 099.002   896 2022.168 12:00:01 ISPW             SP7.2.4   SYSI                                              $
000025       12 099.002  1672 2022.168 12:00:01 NEWWORK         SP7.2.4   SYSI                                              $
000026       13 099.002   896 2022.168 12:00:01 NEWWORKV        SP7.2.4   SYSI                                              $
000027       14 099.003   260 2022.168 12:00:01 NEWWORKV        SP7.2.4   SYSI                                              $
000028       15 099.002  1512 2022.168 12:00:01 ONLPRD          SP7.2.4   SYSI                                              $
000029       16 099.003   492 2022.168 12:00:01 ONLPRD          SP7.2.4   SYSI                                              $
000030       17 099.002   896 2022.168 12:00:01 ONLPRH          SP7.2.4   SYSI                                              $
000031       18 099.002                                                                                                      $
000032       19 099.003  Enter 'ERBSHOW <recnum>' in the EDIT command line to analyze the specified                          $
000033       20 099.002  record in detail.                                                                                   $
000034       21 099.003                                                                                                      $
```

Using ERBSHOW

```
VIEW       SYS22229.T153618.RA000.S03.R0308280
Command ===> _____________________________________________________________
****** ***************************************************** Top of Data **********
000001 Record Number 10: SMF Record Type 99(3)
000002 ======================================
000003
000004 -> SMF record header
000005 ====================
000006
000007    SMF record length      : 628
000008    SMF segment descriptor : '0000'X
000009    SMF system indicator   : '01011110'B
000010    SMF record type        : 99
000011    SMF record time        : 12:00:01
000012    SMF record date        : 22.168
000013    SMF system id          : SYSI
000014    SMF subsystem id       :
000015    SMF record subtype     : 3
000016    Self Defining Length   : 16
000017
000018 -> SMF 99 Self Defining Section
000019 ==============================
000020
000021    Number of triplets     : 2
000022
000023    Section  1 offset      : '0000002C'X
000024    Section  1 length      : '0020'X
000025    Section  1 number      : 1
000026
000027    Section  2 offset      : '0000004C'X
000028    Section  2 length      : '0228'X
000029    Section  2 number      : 1
000030
000031 -> SMF 99 SubType 3 Product Section
000032 ===================================
000033
000034      #1: +0000:  0026F3F1 E2D9D440 40404040 E2D7F74B  *  31SRM      SP7.*
000035          +0010:  F24BF440 E2E8E2C9 40404040 00000000  *2.4 SYSI         *
000036
```

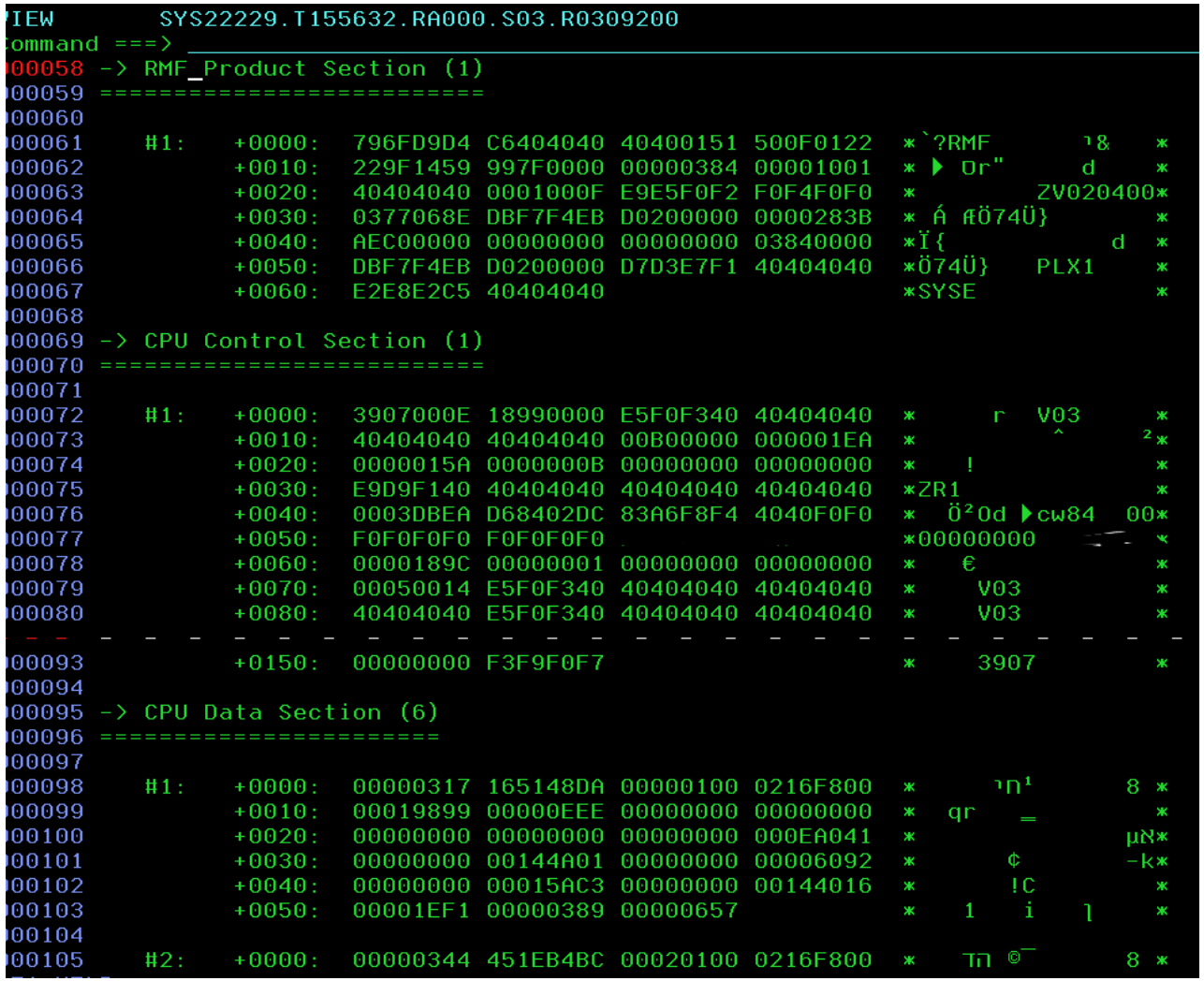
- For supported types there will be several pages of output
- Supported tpes are 70 to 79 and 99

Using ERBSHOW

```
VIEW       SYS22229.T153618.RA000.S03.R0308280
Command ===> ____________________________________________________________
000138 -> SMF 99.3 Proportional Aggregate Speed Plot (0)
000139 ================================================
000140
000141 -> SMF 99.3 Queue Delay Plot (2)
000142 ================================
000143
000144      #1: +0000:  00000001 5BE2D9D4 E2F0F5F0 00000007  *    $SRMS050    *
000145          +0010:  00000001 00000000 00000194 00010008  *           m    *
000146          +0020:  80000000 00000000 00000000 00000000  *Ã              *
000147      #1: +0000:  00000117 00000000                    *                *
000148
000149      #2: +0000:  00000001 5BE2D9D4 E2F0F4C4 00000007  *    $SRMS04D    *
000150          +0010:  00000001 00000000 0000019C 00010008  *           €    *
000151          +0020:  80000000 00000000 00000000 00000000  *Ã              *
000152      #2: +0000:  00000117 00000000                    *                *
000153
000154 -> SMF 99.3 Queue Ready User Average Plot (2)
000155 ============================================
000156
000157      #1: +0000:  00000001 5BE2D9D4 E2F0F5F0 00000001  *    $SRMS050    *
000158          +0010:  00000000 00000000 000001E4 00000008  *           U    *
000159
000160      #2: +0000:  00000001 5BE2D9D4 E2F0F4C4 00000001  *    $SRMS04D    *
000161          +0010:  00000000 00000000 000001E4 00010008  *           U    *
000162      #2: +0000:  00000001 00000000                    *                *
000163
000164 -> SMF 99 Self Defining Section
000165 ===============================
000166
000167    Number of triplets     : 4
000168
000169    Section  1 offset      : '000001EC'X
000170    Section  1 length      : '0020'X
000171    Section  1 number      : 2
000172
000173    Section  2 offset      : '00000000'X
000174    Section  2 length      : '0000'X
```

- Shows the SMF sections with the offsets in character and hex.

Using ERBSHOW - SMF type 70



RMFM3B - Batch reporting for monitor III

- JCL procedure provided in PROCLIB
- Creates a particular monitor III report every interval until stopped
- Writes exceptions via WTO
- 3 canned reports: WFEX, CPC and SYSSUM
- You can write your own(I have not tried)
- Best run as a started task
- Catch exceptions with automation product.

RMFM3B - Batch reporting for monitor III

- JCL procedure provided in PROCLIB
- Creates a particular monitor III report every interval until stopped
- Writes exceptions via WTO
- 3 canned reports: WFEX, CPC and SYSSUM
- You can write your own(I have not tried)
- Best run as a started task
- Catch exceptions with automation

RMFM3B - Exceptions

```
16.03.25 JOO27147  +RMF100I 3B: Processing WFEX Report...
16.03.25 JOO27147  +RMF101I 3B: 00000007 WFEX Exception(s) Encountered
16.04.03 JOO27147  +RMF100I 3B: Processing WFEX Report...
16.04.03 JOO27147  +RMF101I 3B: 00000016 WFEX Exception(s) Encountered
16.05.07 JOO27147  +RMF100I 3B: Processing WFEX Report...
```

RMFM3B - Reports

```
 HARDCOPY      RMF V2R4    Workflow/Exceptions            Line 1 of 3
Command ===>
Switched to option set STANDARD on SYSD.
Samples: 120      System: SYSD  Date: 08/17/22  Time: 16.01.00  Range: 120   Sec
--------------------------- Speed (Workflow) --------------------------------
           Speed of 100 = Maximum, 0 = Stopped         Average CPU Util:  13 %
Name        Users Active       Speed          Name        Users Active       Speed
*SYSTEM       267      9          27          *DEV           23      1          88
ALL TSO        42      0          51          *MASTER*        1      0         100
ALL STC       210      7          25
ALL BATCH       3      2          30
ALL ASCH                    Not avail
ALL OMVS       12      0           9
*PROC         146      8          20
------------------------------ Exceptions -----------------------------------
Name        Reason         Critical val. Possible cause or action
ALL OMVS    PROC-CONSOLE     0.3 users
DBSDDBM1    PROC-JES2MON    16.7 % delay
M12D4300    PROC-E62OM020   31.7 % delay
```

RMFM3B - What next

- Not much information available
- Some information in RMF manuals
- The best source is here RMF2WTO.pdf
  (thanks to Marna Walle for finding it)

Procrastination…

It's Making Me Wait for Access

(James Lund)

The Setup…

- Managed Services company tasked with installing and migrating to z/OS 2.4 (from 2.2)

- Day of PROD migration, everything looks good
  - **Single IPL on a Sunday morning**
  - **All errant log msgs addressed**
  - **Customer seems happy with everything, and it was within migration window**

- Migration +5 weeks
  - **Might be good to do a "clean" IPL**
  - **Little did we know…**

Here Comes the Panic Attack!

- Random tasks abending

- Health Checker messages on the console

```
*HZSPROC  *HZS0003E CHECK(CA_1,CA1_VRFY_SECURITY_PROFILE_TAPE):
*TMSH0161E RACF External Security is active and CA 1 external security
*option is set to YES, but resource class CA@APE has not been defined.
*HZSPROC  *HZS0003E CHECK(IBMRACF,RACF_SENSITIVE_RESOURCES):
*IRRH204E The RACF_SENSITIVE_RESOURCES check has found one or
*more potential errors in the security controls on this system.
```

- Curious messages on some 3PP startup

Here Comes the Panic Attack!

It's a Mistake… Oh, It's a Mistake!

OK, we did *NOT* just run a month with some SAF piece missing

- Pre-conversion? Would have shown up in first IPL
- Why are just *some* apps and tasks affected?
- Only user-defined classes and 3PP?
- Cached rules?

Something obviously changed in system or RACF class definitions

ICHRRCDE?

Robin, to the diagnostic tools!

RACF Active Classes

RACF ISPF option 5.1:

```
 ICHPLST RACF  COMMAND OUTPUT------------------------- LINE 00000011 COL 001 080
ACTIVE CLASSES = DATASET USER GROUP ACCTNUM ACICSPCT APPL BCICSPCT CAMSM
                 CAMSX CCICSCMD CDT CONSOLE CSFKEYS CSFSERV DASDVOL DCICSDCT
                 DIGTCERT DIGTCRIT DIGTNMAP DIGTRING ECICSDCT EJBROLE
                 FACILITY FCICSFCT FIELD GCICSTRN GCSFKEYS GDASDVOL GEJBROLE
                 GTERMINL GXCSFKEY GXFACILI HCICSFCT ILMADMIN JCICSJCT
                 JESSPOOL KCICSJCT LOGSTRM MCICSPPT NCICSPPT NETSPAN OPERCMDS
                 PCICSPSB PERFGRP PMBR PROGRAM PROPCNTL PTKTDATA PTKTVAL
                 QCICSPSB RCICSRES SCDMBR SCICSTST SECDATA SERVAUTH STARTED
                 SURROGAT TCICSTRN TEMPDSN TERMINAL TSOAUTH TSOPROC UCICSTST
                 UNIXPRIV VCICSCMD WBEM WCICSRES WRITER XCSFKEY XFACILIT
```

RACF Active Classes?

RACF ISPF option 5.1:

```
 ICHPLST RACF  COMMAND OUTPUT------------------------- LINE 00000011 COL 001 080
ACTIVE CLASSES = DATASET USER GROUP ACCTNUM ACICSPCT APPL BCICSPCT CAMSM
                 CAMSX CCICSCMD CDT CONSOLE CSFKEYS CSFSERV DASDVOL DCICSDCT
                 DIGTCERT DIGTCRIT DIGTNMAP DIGTRING ECICSDCT EJBROLE
                 FACILITY FCICSFCT FIELD GCICSTRN GCSFKEYS GDASDVOL GEJBROLE
                 GTERMINL GXCSFKEY GXFACILI HCICSFCT ILMADMIN JCICSJCT
                 JESSPOOL KCICSJCT LOGSTRM MCICSPPT NCICSPPT NETSPAN OPERCMDS
                 PCICSPSB PERFGRP PMBR PROGRAM PROPCNTL PTKTDATA PTKTVAL
                 QCICSPSB RCICSRES SCDMBR SCICSTST SECDATA SERVAUTH STARTED
                 SURROGAT TCICSTRN TEMPDSN TERMINAL TSOAUTH TSOPROC UCICSTST
                 UNIXPRIV VCICSCMD WBEM WCICSRES WRITER XCSFKEY XFACILIT
```

Where are all the 3PP classes? EJES, CA@*, SD@*, etc.

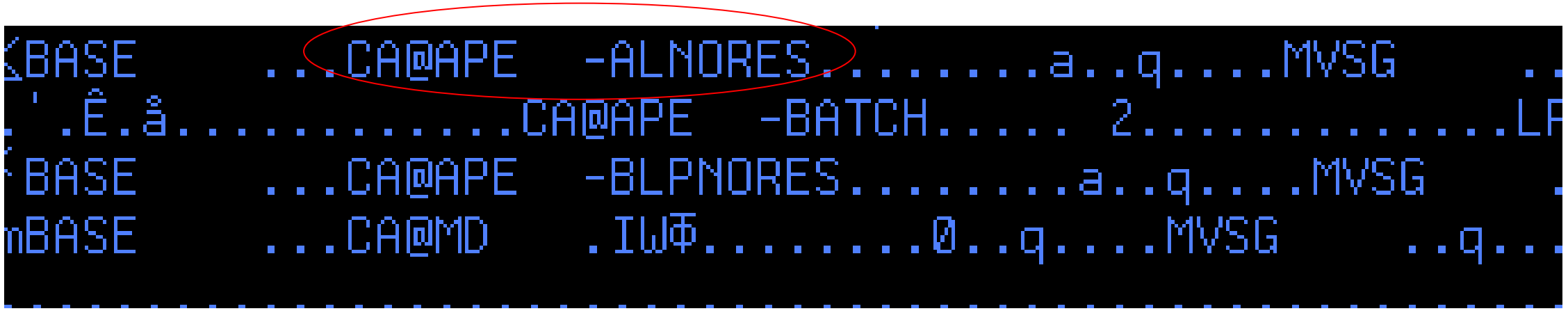
"MS, what happened to the user classes between migration and today? Did y'all do some clean up work?"

Response: "Nothing has changed" YEAH RIGHT!

RACF-L to the Rescue!

So, missing entries in RACF displays.

MS: "Hey, we see classes in cursory ISPF 3.4 BROWSE of RACF databases!"



RACF ignorant, so let's bring in the experts…

Diag Commands!

- SR CLASS(CDT) – Missing classes showing? Nope
  - What's this CDT thing? New to me…
- IRRUT200 with MAP option – Missing classes showing with a profile count? YES!

- https://www.rshconsulting.com/racftips/RSH_Consulting__RACF_Tips__July_2016.pdf
- Article on "Hidden" Profiles, dealing with orphaned RACF class profiles

- EUREKA! This explains our observations!
  - We had deleted classes in RACF – CDT? ICHRRDDE?
  - Confirmed by an overlooked observation…

Gee, I Missed That…

- Looking again at the Active Class report:

```
 ICHPLST RACF  COMMAND OUTPUT------------------------ LINE 00000011 COL 001 080
ACTIVE CLASSES = DATASET USER GROUP ACCTNUM ACICSPCT APPL BCICSPCT CAMSM
                 CAMSX CCICSCMD CDT CONSOLE CSFKEYS CSFSERV DASDVOL DCICSDCT
                 DIGTCERT DIGTCRIT DIGTNMAP DIGTRING ECICSDCT EJBROLE
                 FACILITY FCICSFCT FIELD GCICSTRN GCSFKEYS GDASDVOL GEJBROLE
                 GTERMINL GXCSFKEY GXFACILI HCICSFCT ILMADMIN JCICSJCT
                 JESSPOOL KCICSJCT LOGSTRM MCICSPPT NCICSPPT NETSPAN OPERCMDS
                 PCICSPSB PERFGRP PMBR PROGRAM PROPCNTL PTKTDATA PTKTVAL
                 QCICSPSB RCICSRES SCDMBR SCICSTST SECDATA SERVAUTH STARTED
                 SURROGAT TCICSTRN TEMPDSN TERMINAL TSOAUTH TSOPROC UCICSTST
                 UNIXPRIV VCICSCMD WBEM WCICSRES WRITER XCSFKEY XFACILIT
```

- What's all this, Gov'nah?! A user class!
- We have mixed class definitions in RACF!
  - Old ICHRRCDE CDT module – "cleaned up" by MS?
  - New CLASS(CDE) entries (CA-MSM/CSM)
    - Defunct CDT2DYN utility – creates REXX exec of commands to eliminate need for ICHRRCDE
    - Luckily, previous sysprog used, but never implemented.

CDT2DYN Output

```
RDEFINE CDT CA@APE    CDTINFO(
                                CASE(UPPER)
                                DEFAULTRC( 4 )
                                DEFAULTUACC( NONE )
                                FIRST( ALPHA NATIONAL )
                                GENLIST( DISALLOWED )
                                KEYQUALIFIERS( 0 )
                                MACPROCESSING( NORMAL )
                                MAXLENGTH( 8 )
                                MAXLENX( 8 )
                                OPERATIONS( NO )
                                OTHER( ALPHA NATIONAL NUMERIC SPECIAL )
                                POSIT( 20 )
                                PROFILESALLOWED( YES )
                                RACLIST( DISALLOWED )
                                SECLABELSREQUIRED( NO )
                                SIGNAL( NO )
                                )
```

SMF is Love!

- Now, to prove to MS that they messed up!
  - ICHRRCDE:
    - Add? Delete? Replace?
    - Where? Can't assume SYS1.LINKLIB only

- SMF 42 - DFSMS statistics and configuration
  - Subtype 21 – PDS(E) Deletion
  - Subtype 24 – PDS(E) Add/Replace
  - Subtype 25 – PDS(E) Rename

- Used MXG as data repository for 8 prior weeks

The Evidence is Undeniable!

Sub24 -

04APR2022:19:26:39.62 MSID added CDT to user LOADLIB

Sub21 -

04APR2022:19:26:39.63 MSID deleted CDT from z/OS 2.2 LOADLIB

Sub25 -

No RENAMES

Lessons Learned

- Don't use mixed class defs – CLEAN UP YOUR MESS!
  - RACF dynamic CDE is your best bet
- If you want something done right…
  - (Ask me later about parameter migration oversights!)

Acknowledgements (Knowing and Unknowing)

- Juan G. Mautalen, República Argentina
- Robert S. Hansel, RSH Consulting

Ode to a Small Lump of Green Putty I Found in My Armpit One Midsummer Morning

(James Lund)

Ode to Mainframe Legacy – Over 60 Years of Computing (James Lund)

IBM zEnterprise BC12

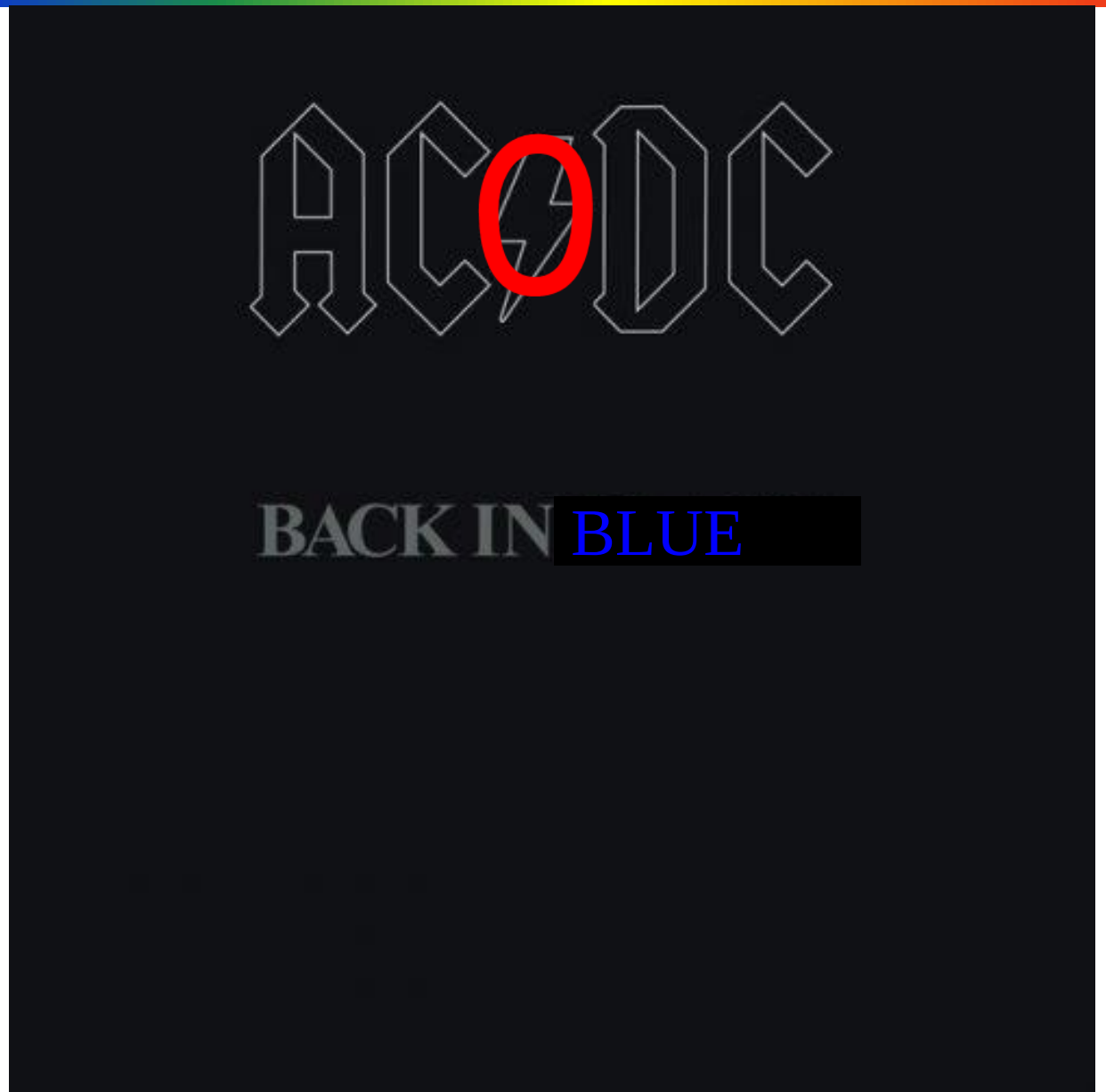
IBM eServer zSeries 890 and IBM System z10BC

IBM eServer zSeries 890 and IBM System z10BC

ACODC

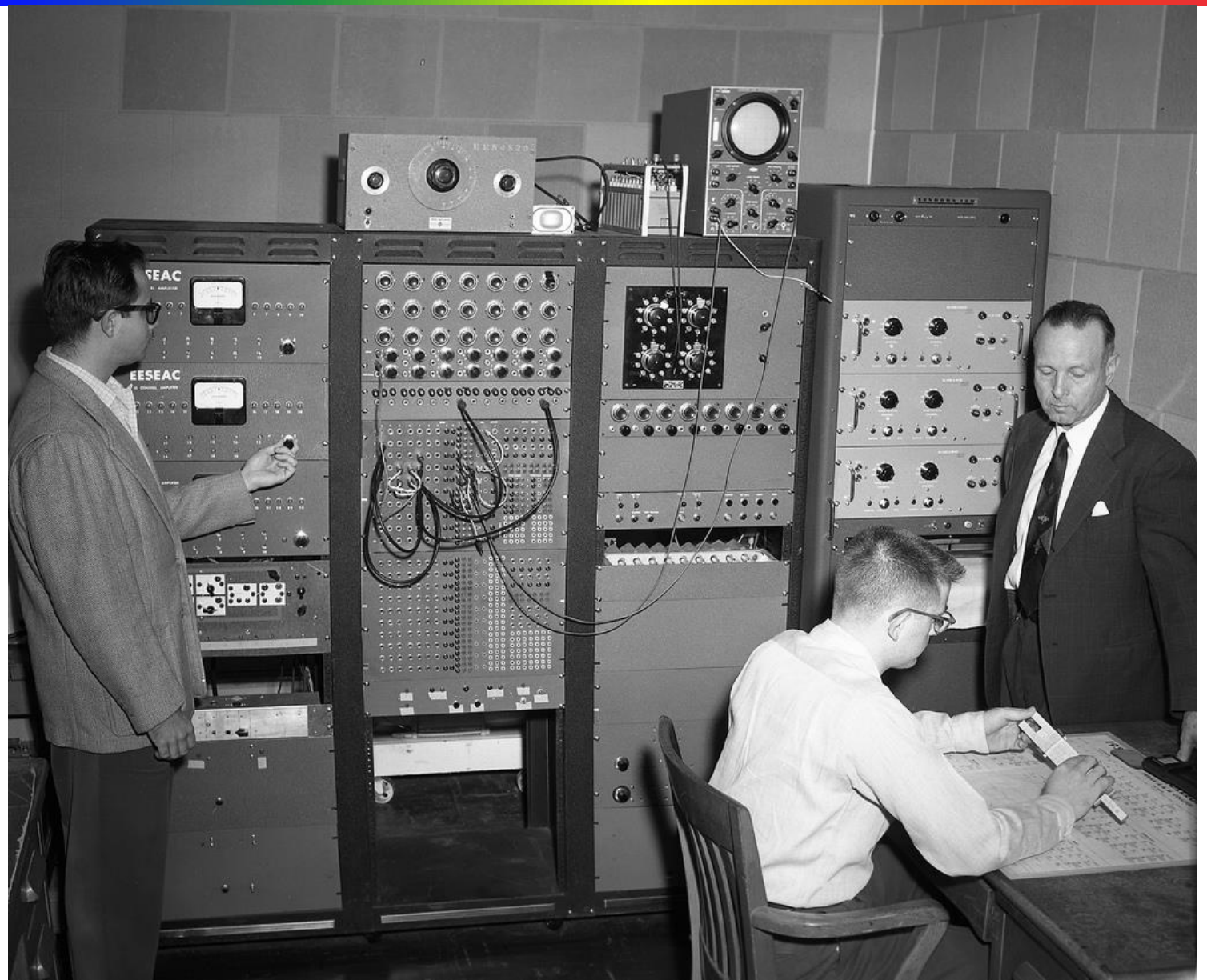
ROCK ON!

ROCK ON!



And So IT Begins…

1949 EESEAC – Engineering Experiment Station Electronic Analog Computer



1949 EESEAC – Engineering Experiment Station Electronic Analog Computer

1954 EESEAC – Add Electronic Function Multiplier/Divider

AN ELECTRONIC FUNCTION MULTIPLIER

FOR

THE EESEAC ANALOG COMPUTER

By

W. J. Lindsay

A Thesis

Submitted to the Graduate School of the

Agricultural and Mechanical College of Texas in

partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major Subject: Electrical Engineering

August 1954

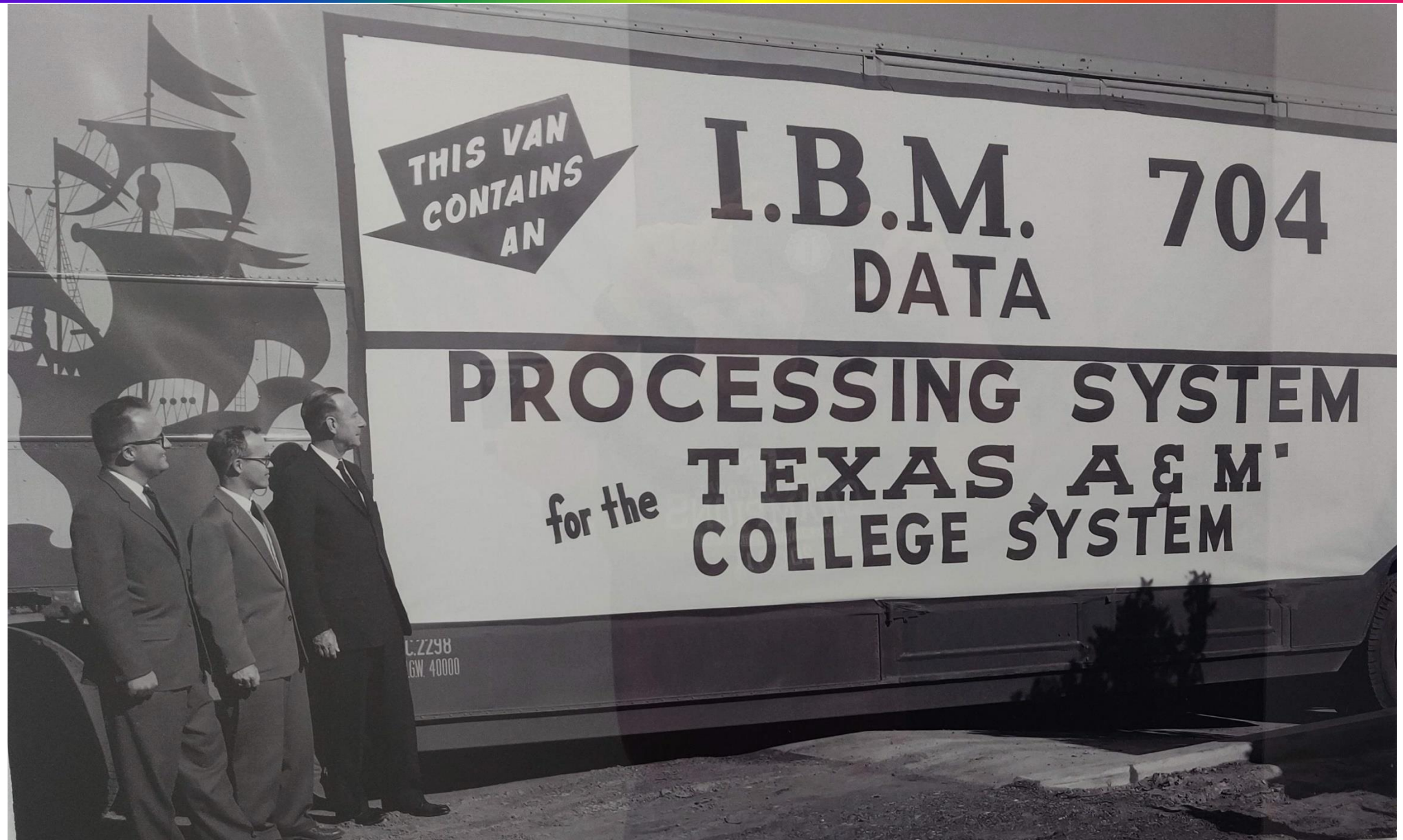
1950 IBM 407 Accounting Machine – Plug Board Programmable

1950 IBM 407 Accounting Machine and IBM 514 Summary Punch

1956 IBM 650 Magnetic Drum Data Processing Machine

1956 IBM 650 Magnetic Drum Data Processing Machine

1957 IBM 704


THIS VAN CONTAINS AN
I.B.M. 704
DATA
PROCESSING SYSTEM
for the TEXAS A&M
COLLEGE SYSTEM

1958 TEES and SHARE

IBM®

SHARE

REFERENCE MANUAL

TC MR. JOHN R. PEARSON, SUPERVISOR
PROGRAMMING AND PROCEDURES
ENGINEERING SERVICES DIVISION
TELECOMPUTING CORPORATION
HOLLOMAN AFB, NEW MEXICO

TE MR. ROBERT L. SMITH, JR.
DATA PROCESSING CENTER
TEXAS ENGINEERING EXP. STATION
COLLEGE STATION, TEXAS

TR MR. MARVIN HOWARD
ROOM 1305, BUILDING F
RAMO-WOOLDRIDGE DIVISION OF,
THOMPSON RAMO WOOLDRIDGE INC.
P. O. BOX 90534 AIRPORT STATION
LOS ANGELES 45, CALIFORNIA

IBM 026 Card Punch and IBM 056 Card Verifier

IBM 072 Alphabetic Collator

IBM 082 Sorter

Long Line of Mainframe Technology

- 1963 7090
- 1965 7094
- 1966 System/360-65
- 1971 System/370-158
- 1988 3090-200E/600E/400J
- 1993 Amdahl 5995
- 1999 MP2003-246
- 2000 9121-411
- 2001 (MP3000) 7060-H70 RPQ
- ...

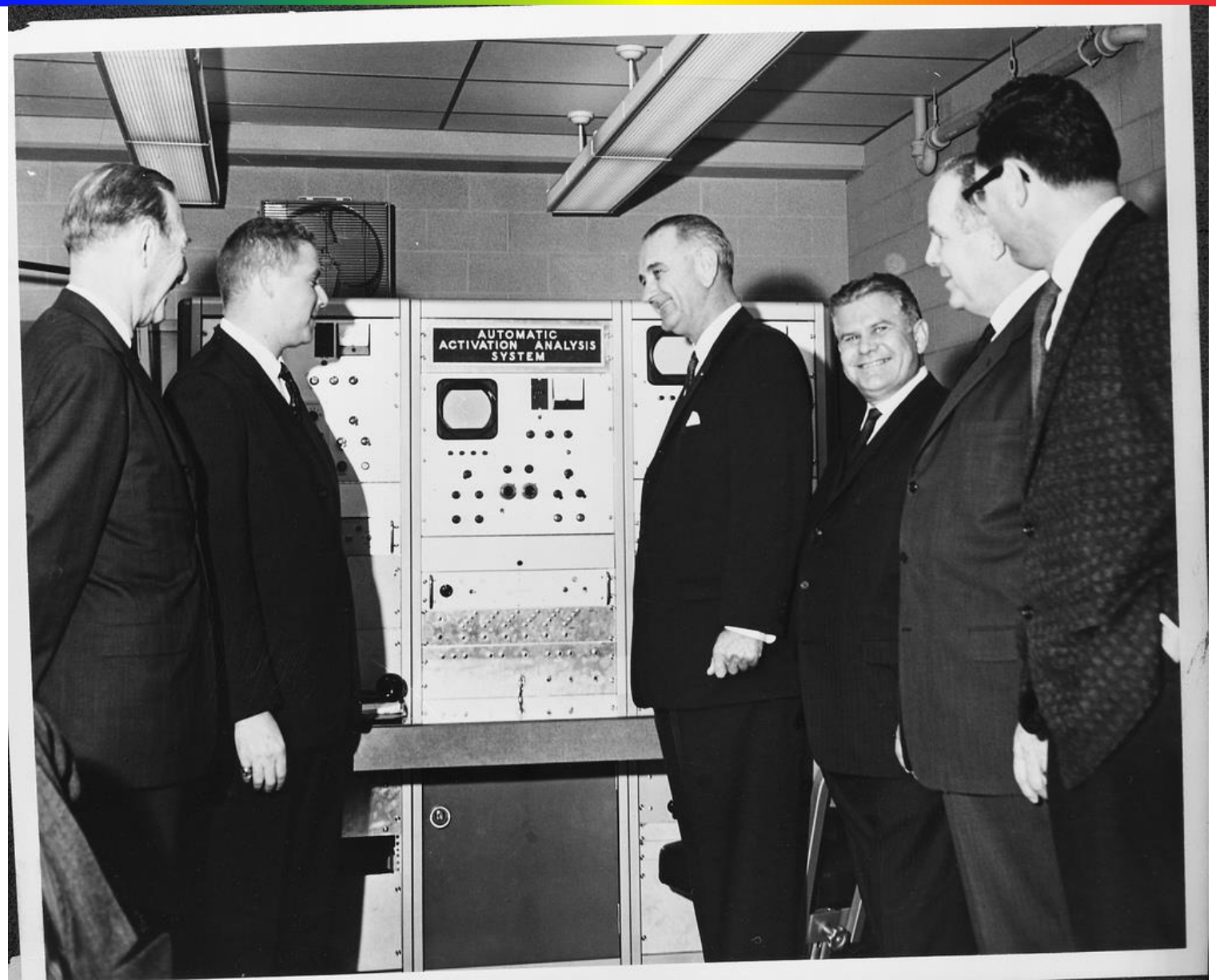
YAWN

Geepers....

How long is this list?!?

President Lyndon Johnson

Work with NASA



Computer Seminars – can you find the two non-Military?

6th. ANNUAL SEMINAR ON
ELECTRONIC COMPUTERS
TEXAS A&M UNIVERSITY
COLLEGE STATION, TEXAS
1968

YAWN

Another list?

WE GET IT…MOVE ON.

YAWN

Another list?

WE GET IT…MOVE ON.

I MEAN IT…MOVE ON!

IBM zEnterprise BC12

Retirement Scheduled for February 2023

Acknowledgements (Knowing and Unknowing)

- Cushing Library Historical Images Collection
- TAMU Sysprogs and Operators
- The SHARE Community

You all made it happen! Thank you!

How Low Can Low-Core Go?

(Ed Jaffe)

Restart New PSW Has a New Look

- For quite a few years now, when IPLing z/OS in z/Architecture mode, the first 8 bytes of the z/OS PSA have been populated with:

```
IPCS OUTPUT STREAM ----------------------------------------------------------
Command ===>                                                  SCROLL ===> CSR
 ****************************** TOP OF DATA ***********************************

 LIST 00. ASID(X'004A') LENGTH(X'08') AREA
 ASID(X'004A') ADDRESS(00.) KEY(00) PREFIXED
 00000000. 000A0000 000130E1                    |........                |
 ****************************** END OF DATA *************************************
```

- Now look!!!

```
IPCS OUTPUT STREAM ---------------------------------------- Line 0 Cols 1 78
Command ===>                                                  SCROLL ===> CSR
 ****************************** TOP OF DATA ***********************************

 LIST 00. ASID(X'00A1') LENGTH(X'08') AREA
 ASID(X'00A1') ADDRESS(00.) KEY(08)
 00000000. 00000000 00000000                    |........                |
 ****************************** END OF DATA *************************************
```

Wha???

- We missed catching a bug because of this.
  - Of course, the bug was our fault – not IBM's.
- But, the question remains: Why do this? Bug fix? New feature?

Restart New PSW Has a New Look

- I asked Jim Mulder about this and he said simply:

  In z/OS 2.5, we changed the old ESA/390 interrupt new PSW locations from being 0E1 disabled wait states to the more customary zeros value for reserved storage area.

  Jim Mulder

- Peter Relson gave me some more general information later.
- Bottom line: it seems to be in the category of general cleanup.
- It's worth noting they did this for all ESA/390 new PSWs in PSA and not just the Restart New PSW at location zero.

- So if you didn't already know, consider this your Heads Up!! ☺

ZAD SLIP Report

- Because there was an issue with our code addressing location zero by accident, I wanted to know why nobody noticed it in our daily IEAVTSZR reports we generate using SLIP ZAD.
  - https://www.ibm.com/docs/en/zos/2.5.0?topic=traps-slip-zero-address-detection-zad

z/OS / 2.5.0 /

Feedback Product list

SLIP zero address detection (ZAD)

Last Updated: 2022-07-27

A Zero Address Detection (ZAD) event is a PER program interrupt caused by execution of an instruction that accesses (stores or fetches) storage using an operand address that was formed from a general register containing zero, when the PSW PER bit is on and Zero Address Detection is enabled. You enable Zero Address Detection by a SLIP PER trap of type ZAD. A ZAD trap can help to detect errors that involve unintentionally referencing the PSA. All instruction ranges are monitored for ZAD events; you can limit ZAD events only by limiting the address spaces and jobs for which PER is active. You can limit the ZAD events that match your SLIP traps by using other SLIP filtering keywords.

- The general consensus is that the report has become ugly.

ZAD SLIP Report

- We've started using node.js for some of our back-office infrastructure (Paul Scott from PSI has done a few presentations around the world on this – including here at SHARE), we've been integrating with Zowe which also uses node.js, and we started using PFA which is a Java application.
- Java and node.js generate a lot of ZAD hits. Roland Koo confirms they do NOT share the same compiler "back end."

```
PFA3     009B  00000000_115ADFAC       1  *PATHNAM  0007CFAC     58606224EC68  L      R6,548(,R6)
                /usr/lpp/java/J8.0/lib/s390/default/libj9prt29.so
PFA3     009B  00000000_115AE1B6       1  *PATHNAM  0007D1B6     58606224EC68  L      R6,548(,R6)
                /usr/lpp/java/J8.0/lib/s390/default/libj9prt29.so
PFA3     009B  00000000_787E1F66       1  *PATHNAM  001E1F66     58003074A701  L      R0,116(,R3)
                /usr/lpp/java/J8.0/lib/s390/default/libj9jit29.so
PFA3     009B  00000000_79F2D0BE       1  *PATHNAM  0002D0BE     5006A0045090  ST     R0,4(R6,R10)
                /usr/lpp/java/J8.0/lib/s390/default/libj9vm29.so
PFA3     009B  00000000_79F2E8EE       1  *PATHNAM  0002E8EE     5009E0045060  ST     R0,4(R9,R14)
                /usr/lpp/java/J8.0/lib/s390/default/libj9vm29.so
PFA3     009B  00000000_7A11E484       1  *PATHNAM  0021E484     E37210010094  LLC    R7,1(R2,R1)
                /usr/lpp/java/J8.0/lib/s390/default/libj9vm29.so
PFA3     009B  00000000_7A11E48A       1  *PATHNAM  0021E48A     E33210020094  LLC    R3,2(R2,R1)
                /usr/lpp/java/J8.0/lib/s390/default/libj9vm29.so
PFA2     009C  00000000_11556FAC       1  *PATHNAM  0007CFAC     58606224EC68  L      R6,548(,R6)
                /usr/lpp/java/J8.0/lib/s390/default/libj9prt29.so
PFA2     009C  00000000_115571B6       1  *PATHNAM  0007D1B6     58606224EC68  L      R6,548(,R6)
                /usr/lpp/java/J8.0/lib/s390/default/libj9prt29.so
```

If you're looking for a specific component, you can use FIND, but we tend to "wade" through looking for anything that looks like it's ours.

Scary!

ZAD SLIP Report

- You can set a ZAD SLIP with PVTMOD=('z/OS UNIX path') and ACTION=IGNORE for the most annoying ones and put them in IEASLPxx or COMMNDxx (we do that latter) to remove them from consideration.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
MVS60       SYS2.PARMLIB(COMMND00) - 01.99                  Columns 00001 00080
Command ===>                                                Scroll ===> CSR
000041 COM='SETPROG LPA,ADD,MOD=IEAVTSZE,DSN=SYS1.LINKLIB,FIXED'
000042 COM='SLIP SET,ZAD,ASIDSA=SA,A=AEXIT,AEXIT=IEAVTSZE,ID=ZAD1,PL=50,OK,END'
000043 COM='SLIP SET,ZAD,PVTMOD=('libnode.83.so'),A=IGNORE,ID=ZAD2,END'
000044 COM='SLIP SET,ZAD,PVTMOD=('libj9prt29.so'),A=IGNORE,ID=ZAD3,END'
000045 COM='SLIP SET,ZAD,PVTMOD=('libj9vm29.so'),A=IGNORE,ID=ZAD4,END'
000046 COM='SLIP SET,ZAD,PVTMOD=('libwrappers.so'),A=IGNORE,ID=ZAD5,END'
000047 COM='SLIP SET,ZAD,PVTMOD=('libzNativeServices.so'),A=IGNORE,ID=ZAD6,END'
```

- I hate doing this because I try to help IBM in every way I can, but the compiler developers in Toronto were told about this long ago and seem to have their heads in the sand about it, so I don't wanna make their lack of interest my problem... or yours!

Lock Around the Clock

(Ed Jaffe)

Diagnosing an Early Unlock Condition

- One of the most-difficult bugs to diagnose is when a lock gets released too early.
- Typically this is discovered in one of two ways:
  - Some sort of logical corruption – by far the worst of the two
  - An abend attempting to invoke an MVS service that requires a lock held
- You want to know who released the lock, when and why?
- Answering these questions is essentially an impossible task with a typical SVC dump because SETLOCK requests are not traced in the SYSTRACE.
- Some products might create CTRACE entries showing lock activity. For them, activating the proper CTRACE options can shed light on the problem.
- But what alternative to do the rest of us have?

Diagnosing an Early Unlock Condition

- By luck (or design), most popular lock requests have unique EPAs.
- The address of a given lock request is obtained from a hard-wired offset from the start of the lock interface table.
- Examples:
  - SETLOCK OBTAIN TYPE=CPU

```
+         L     14,PSALITA-PSA(0,0)   LOCK INTERFACE TABLE ADDRESS    @P1A 01-SETLOCK
+         LM    11,13,1284(14)        LOAD ADDRESS OF THE LOCKWORD,   @L2C 01-SETLOCK
+*                                     PARM LIST POINTER, AND LOCK RTN
+*                                     ENTRY POINT.
+         BALR  14,13                 BRANCH ENTER LOCK ROUTINE             01-SETLOCK
```

  - SETLOCK OBTAIN TYPE=LOCAL

```
+         L     14,PSALITA-PSA(0,0)   LOCK INTERFACE TABLE ADDRESS    @P1A 02-SETLOCK
+         L     13,576+8(14,0)        LOAD ADDRESS OF LOCK RTN        @L8C 02-SETLOCK
+         BALR  14,13                 BRANCH ENTER LOCK ROUTINE             02-SETLOCK
```

Diagnosing an Early Unlock Condition

- SETLOCK OBTAIN TYPE=CMS

```
+         L     14,PSALITA-PSA(0,0)  LOCK INTERFACE TABLE ADDRESS   @P1A 01-SETLOCK
+         LM    11,13,540(14)        LOAD ADDRESS OF THE LOCKWORD,  @L2C 01-SETLOCK
+*                                    PARM LIST POINTER, AND LOCK RTN
+*                                    ENTRY POINT.
+         BALR  14,13                BRANCH ENTER LOCK ROUTINE            01-SETLOCK
```

- The SETLOCK RELEASE entry points are likewise mostly unique from each other and from their SETLOCK OBTAIN counterparts.
- There are many different MVS lock calls (as I will show in the slides that follow) and each has its own Lock Interface Table entry.
- The vast majority of software developers need only concern themselves with a few of them e.g., CPU, LOCAL, etc.

Diagnosing an Early Unlock Condition - Conditional Obtains

```
.*
.*****************************************************************
.*                                                               *
.*       CONDITIONAL OBTAIN LIT OFFSETS (DECIMAL)                *
.*                                                               *
.*****************************************************************
.*
&COBT(1)     SETA 0          DISPATCHER LOCK                      @L8A
&COBT(2)     SETA 48         RSMDS LOCK                           @L8A
&COBT(3)     SETA 96         IOSUCB LOCK                          @L8A
&COBT(4)     SETA 144        IOSLCH LOCK - NO LONGER SUPPORTED    @L8A
&COBT(5)     SETA 192        IOSYNCH LOCK                         @L8A
&COBT(6)     SETA 240        TPNCB LOCK - NO LONGER SUPPORTED     @L8A
&COBT(7)     SETA 288        TPDNCB LOCK - NO LONGER SUPPORTED    @L8A
&COBT(8)     SETA 336        TPACBDEB LOCK                        @L8A
&COBT(9)     SETA 384        ASM LOCK                             @L8A
&COBT(10)    SETA 432        SALLOC LOCK                          @L8A
&COBT(11)    SETA 480        SRM lock                             @LPC
&COBT(12)    SETA 528        CMS LOCK                             @L8A
&COBT(13)    SETA 564        LOCAL LOCK                           @L8A
&COBT(14)    SETA -1         SPIN - USED FOR RELEASE ONLY         @L8A
&COBT(15)    SETA 612        CMSEQDQ LOCK                         @L8A
&COBT(16)    SETA -1         CMSALL - COND OBTAIN NOT SUPPORTED   @L8A
&COBT(17)    SETA -1         ALL CMS LOCKS HELD- FOR RELEASE ONLY @L8A
&COBT(18)    SETA 684        CMSSMF LOCK                          @L8A
&COBT(19)    SETA 720        CML LOCK                             @L8A
&COBT(20)    SETA 756        TRACE LOCK - SHARED ENTRY            @L8A
&COBT(21)    SETA 804        TRACE LOCK - EXCLUSIVE ENTRY         @L8A
&COBT(22)    SETA 852        VSMPAG LOCK                          @L8A
&COBT(23)    SETA 900        RSM LOCK - SHARED ENTRY              @L8A
&COBT(24)    SETA 948        RSM LOCK - EXCLUSIVE ENTRY           @L8A
&COBT(25)    SETA 996        RSMAD LOCK                           @L8A
&COBT(26)    SETA 1044       RSMXM LOCK                           @L8A
&COBT(27)    SETA 1092       RSMST LOCK                           @L8A
&COBT(28)    SETA 1140       ASMGL LOCK                           @L8A
&COBT(29)    SETA 1188       VSMFIX LOCK                          @L8A
&COBT(30)    SETA 1236       RSMGL LOCK                           @L8A
&COBT(31)    SETA 1284       CPU LOCK                             @L8A
&COBT(32)    SETA -1         ALL - USED FOR RELEASE ONLY          @L8A
&COBT(33)    SETA -1         (REG) - USED FOR RELEASE ONLY        @L8A
&COBT(34)    SETA 1356       RSMCM LOCK                           @L8A
&COBT(35)    SETA 1420       IOS LOCK - SHARED ENTRY              @L8A
&COBT(36)    SETA 1468       IOS LOCK - EXCLUSIVE ENTRY           @L8A
&COBT(37)    SETA 0          BMFLSD LOCK                          @L7A
&COBT(38)    SETA 48         ETRSET LOCK                          @L6A
&COBT(39)    SETA 96         XCFDS LOCK                           @L5A
&COBT(40)    SETA 144        XCFRES LOCK                          @L5A
&COBT(41)    SETA 192        XCFQ LOCK - SHARED ENTRY             @L5A
&COBT(42)    SETA 240        XCFQ LOCK - EXCLUSIVE ENTRY          @L5A
&COBT(43)    SETA 292        IOSULUT LOCK - SHARED ENTRY          @L9A
&COBT(44)    SETA 340        IOSULUT LOCK - EXCLUSIVE ENTRY       @L9A
&COBT(45)    SETA 388        IXLSCH LOCK                          @LBC
&COBT(46)    SETA 436        IXLSHR LOCK - SHARED ENTRY           @LBC
&COBT(47)    SETA 484        IXLSHR LOCK - EXCLUSIVE ENTRY        @LBA
&COBT(48)    SETA 532        IXLDS LOCK                           @LBC
&COBT(49)    SETA 580        IXLSHELL LOCK                        @LBC
&COBT(50)    SETA 628        IXLREQST LOCK                        @01A
&COBT(51)    SETA 676        WLMQ LOCK - SHARED ENTRY             @LCA
&COBT(52)    SETA 724        WLMQ LOCK - EXCLUSIVE ENTRY          @LCA
&COBT(53)    SETA 772        WLMRES LOCK                          @LCA
&COBT(54)    SETA 820        REGSRV LOCK - SHARED ENTRY           @LCA
&COBT(55)    SETA 868        REGSRV LOCK - EXCLUSIVE ENTRY        @LCA
&COBT(56)    SETA 916        CONTEXT LOCK                         @LCA
&COBT(57)    SETA 964        SSD LOCK                             @LDA
&COBT(58)    SETA 1012       CMSLATCH LOCK                        @LEA
.* Note, there are only 3 CMSLATCH lock LIT entries, instead of 4.
&COBT(59)    SETA 1048       GRSINT LOCK - SHARED ENTRY           @LFA
&COBT(60)    SETA 1096       GRSINT LOCK - EXCLUSIVE ENTRY        @LFA
&COBT(61)    SETA 1144       MISC lock                            @LJC
&COBT(62)    SETA 1192       DONOTUS2 lock                        @LJC
&COBT(63)    SETA 1240       DONOTUS3 lock                        @LJC
&COBT(64)    SETA 1288       DONOTUS4 lock                        @LJC
&COBT(65)    SETA 1336       DONOTUS5 lock                        @LJC
&COBT(66)    SETA 1384       HCWDRLK2 lock                        @LGA
&COBT(67)    SETA 1432       HCWDRLK1 lock                        @LGA
&COBT(68)    SETA 1480       SRMENQ LOCK - SHARED ENTRY           @LHA
&COBT(69)    SETA 1528       SRMENQ LOCK - EXCLUSIVE ENTRY        @LHA
.* Entries 70-74 are SETLOCKP only
&COBT(75)    SETA 1528+6*48  RSMQ LOCK                            @LKA
&COBT(76)    SETA 1528+7*48  SSDGROUP lock                        @LMA
```

Diagnosing an Early Unlock Condition - Unconditional Obtains

```
.*
.*****************************************************************
.*                                                               *
.*       UNCONDITIONAL OBTAIN LIT OFFSETS (DECIMAL)              *
.*                                                               *
.*****************************************************************
.*
&UOBT(1)     SETA 12          DISPATCHER LOCK                     @L8A
&UOBT(2)     SETA 60          RSMDS LOCK                          @L8A
&UOBT(3)     SETA 108         IOSUCB LOCK                         @L8A
&UOBT(4)     SETA 156         IOSLCH LOCK - NO LONGER SUPPORTED   @L8A
&UOBT(5)     SETA 204         IOSYNCH LOCK                        @L8A
&UOBT(6)     SETA 252         TPNCB LOCK - NO LONGER SUPPORTED    @L8A
&UOBT(7)     SETA 300         TPDNCB LOCK - NO LONGER SUPPORTED   @L8A
&UOBT(8)     SETA 348         TPACBDEB LOCK                       @L8A
&UOBT(9)     SETA 396         ASM LOCK                            @L8A
&UOBT(10)    SETA 444         SALLOC LOCK                         @L8A
&UOBT(11)    SETA 492         SRM LOCK                            @L8A
&UOBT(12)    SETA 540         CMS LOCK                            @L8A
&UOBT(13)    SETA 576         LOCAL LOCK                          @L8A
&UOBT(14)    SETA -1          SPIN - USED FOR RELEASE ONLY        @L8A
&UOBT(15)    SETA 624         CMSEQDQ LOCK                        @L8A
&UOBT(16)    SETA 648         CMSALL                              @L8A
&UOBT(17)    SETA -1          ALL CMS LOCKS HELD- FOR RELEASE ONLY @L8A
&UOBT(18)    SETA 696         CMSSMF LOCK                         @L8A
&UOBT(19)    SETA 732         CML LOCK                            @L8A
&UOBT(20)    SETA 768         TRACE LOCK - SHARED ENTRY           @L8A
&UOBT(21)    SETA 816         TRACE LOCK - EXCLUSIVE ENTRY        @L8A
&UOBT(22)    SETA 864         VSMPAG LOCK                         @L8A
&UOBT(23)    SETA 912         RSM LOCK - SHARED ENTRY             @L8A
&UOBT(24)    SETA 960         RSM LOCK - EXCLUSIVE ENTRY          @L8A
&UOBT(25)    SETA 1008        RSMAD LOCK                          @L8A
&UOBT(26)    SETA 1056        RSMXM LOCK                          @L8A
&UOBT(27)    SETA 1104        RSMST LOCK                          @L8A
&UOBT(28)    SETA 1152        ASMGL LOCK                          @L8A
&UOBT(29)    SETA 1200        VSMFIX LOCK                         @L8A
&UOBT(30)    SETA 1248        RSMGL LOCK                          @L8A
&UOBT(31)    SETA 1284        CPU LOCK                            @L8A
&UOBT(32)    SETA -1          ALL - USED FOR RELEASE ONLY         @L8A
&UOBT(33)    SETA -1          (REG) - USED FOR RELEASE ONLY       @L8A
&UOBT(34)    SETA 1368        RSMCM LOCK                          @L8A
&UOBT(35)    SETA 1432        IOS LOCK - SHARED ENTRY             @L8A
&UOBT(36)    SETA 1480        IOS LOCK - EXCLUSIVE ENTRY          @L8A
&UOBT(37)    SETA 12          BMFLSD LOCK                         @L7A
&UOBT(38)    SETA 60          ETRSET LOCK                         @L6A
&UOBT(39)    SETA 108         XCFDS LOCK                          @L5A
&UOBT(40)    SETA 156         XCFRES LOCK                         @L5A
&UOBT(41)    SETA 204         XCFQ LOCK - SHARED ENTRY            @L5A
&UOBT(42)    SETA 252         XCFQ LOCK - EXCLUSIVE ENTRY         @L5A
&UOBT(43)    SETA 304         IOSULUT LOCK - SHARED ENTRY         @L9A
&UOBT(44)    SETA 352         IOSULUT LOCK - EXCLUSIVE ENTRY      @L9A
&UOBT(45)    SETA 400         IXLSCH LOCK                         @LBC
&UOBT(46)    SETA 448         IXLSHR LOCK - SHARED ENTRY          @LBC
&UOBT(47)    SETA 496         IXLSHR LOCK - EXCLUSIVE ENTRY       @LBC
&UOBT(48)    SETA 544         IXLDS LOCK                          @LBC
&UOBT(49)    SETA 592         IXLSHELL LOCK                       @LBA
&UOBT(50)    SETA 640         IXLREQST LOCK                       @01A
&UOBT(51)    SETA 688         WLMQ LOCK - SHARED ENTRY            @LCA
&UOBT(52)    SETA 736         WLMQ LOCK - EXCLUSIVE ENTRY         @LCA
&UOBT(53)    SETA 784         WLMRES LOCK                         @LCA
&UOBT(54)    SETA 832         REGSRV LOCK - SHARED ENTRY          @LCA
&UOBT(55)    SETA 880         REGSRV LOCK - EXCLUSIVE ENTRY       @LCA
&UOBT(56)    SETA 928         CONTEXT LOCK                        @LCA
&UOBT(57)    SETA 976         SSD LOCK                            @LDA
&UOBT(58)    SETA 1024        CMSLATCH LOCK                       @LEA
&UOBT(59)    SETA 1060        GRSINT LOCK - SHARED ENTRY          @LFA
&UOBT(60)    SETA 1108        GRSINT LOCK - EXCLUSIVE ENTRY       @LFA
&UOBT(61)    SETA 1156        MISC lock                           @LJC
&UOBT(62)    SETA 1204        DONOTUS2 lock                       @LJC
&UOBT(63)    SETA 1252        DONOTUS3 lock                       @LJC
&UOBT(64)    SETA 1300        DONOTUS4 lock                       @LJC
&UOBT(65)    SETA 1348        DONOTUS5 lock                       @LJC
&UOBT(66)    SETA 1396        HCWDRLK2 lock                       @LGA
&UOBT(67)    SETA 1444        HCWDRLK1 lock                       @LGA
&UOBT(68)    SETA 1492        SRMENQ LOCK - SHARED ENTRY          @LHA
&UOBT(69)    SETA 1540        SRMENQ LOCK - EXCLUSIVE ENTRY       @LHA
.* Entries 70-74 are SETLOCKP only
&UOBT(75)    SETA 1540+6*48   RSMQ LOCK                           @LKA
&UOBT(76)    SETA 1540+7*48   SSDGROUP lock                       @LMA
```

Diagnosing an Early Unlock Condition - Enabled Releases

```
.*
.*****************************************************************
.*                                                               *
.*       RELEASE LIT OFFSETS (DECIMAL)                           *
.*                                                               *
.*****************************************************************
.*
&REL(1)      SETA 24          DISPATCHER LOCK                    @L8A
&REL(2)      SETA 72          RSMDS LOCK                         @L8A
&REL(3)      SETA 120         IOSUCB LOCK                        @L8A
&REL(4)      SETA 168         IOSLCH LOCK - NO LONGER SUPPORTED  @L8A
&REL(5)      SETA 216         IOSYNCH LOCK                       @L8A
&REL(6)      SETA 264         TPNCB LOCK - NO LONGER SUPPORTED   @L8A
&REL(7)      SETA 312         TPDNCB LOCK - NO LONGER SUPPORTED  @L8A
&REL(8)      SETA 360         TPACBDEB LOCK                      @L8A
&REL(9)      SETA 408         ASM LOCK                           @L8A
&REL(10)     SETA 456         SALLOC LOCK                        @L8A
&REL(11)     SETA 504         SRM LOCK                           @L8A
&REL(12)     SETA 552         CMS LOCK                           @L8A
&REL(13)     SETA 588         LOCAL LOCK                         @L8A
&REL(14)     SETA 600         SPIN - RELEASE ALL SPIN LOCKS HELD @L8A
&REL(15)     SETA 636         CMSEQDQ LOCK                       @L8A
&REL(16)     SETA 660         CMSALL                             @L8A
&REL(17)     SETA 672         ALL CMS LOCKS HELD                 @L8A
&REL(18)     SETA 708         CMSSMF LOCK                        @L8A
&REL(19)     SETA 744         CML LOCK                           @L8A
&REL(20)     SETA 780         TRACE LOCK - SHARED ENTRY          @L8A
&REL(21)     SETA 828         TRACE LOCK - EXCLUSIVE ENTRY       @L8A
&REL(22)     SETA 876         VSMPAG LOCK                        @L8A
&REL(23)     SETA 924         RSM LOCK - SHARED ENTRY            @L8A
&REL(24)     SETA 972         RSM LOCK - EXCLUSIVE ENTRY         @L8A
&REL(25)     SETA 1020        RSMAD LOCK                         @L8A
&REL(26)     SETA 1068        RSMXM LOCK                         @L8A
&REL(27)     SETA 1116        RSMST LOCK                         @L8A
&REL(28)     SETA 1164        ASMGL LOCK                         @L8A
&REL(29)     SETA 1212        VSMFIX LOCK                        @L8A
&REL(30)     SETA 1260        RSMGL LOCK                         @L8A
&REL(31)     SETA 1296        CPU LOCK                           @L8A
&REL(32)     SETA 1308        ALL - RELEASE ALL LOCKS HELD       @L8A
&REL(33)     SETA 1332        (REG) - RELEASE LOCKS SPECIFIED    @L8A
&REL(34)     SETA 1380        RSMCM LOCK                         @L8A
&REL(35)     SETA 1444        IOS LOCK - SHARED ENTRY            @L8A
&REL(36)     SETA 1492        IOS LOCK - EXCLUSIVE ENTRY         @L8A
&REL(37)     SETA 24          BMFLSD LOCK                        @L7A
&REL(38)     SETA 72          ETRSET LOCK                        @L6A
&REL(39)     SETA 120         XCFDS LOCK                         @L5A
&REL(40)     SETA 168         XCFRES LOCK                        @L5A
&REL(41)     SETA 216         XCFQ LOCK - SHARED ENTRY           @L5A
&REL(42)     SETA 264         XCFQ LOCK - EXCLUSIVE ENTRY        @L5A
&REL(43)     SETA 316         IOSULUT LOCK - SHARED ENTRY        @L9A
&REL(44)     SETA 364         IOSULUT LOCK - EXCLUSIVE ENTRY     @L9A
&REL(45)     SETA 412         IXLSCH LOCK                        @LBC
&REL(46)     SETA 460         IXLSHR LOCK - SHARED ENTRY         @LBC
&REL(47)     SETA 508         IXLSHR LOCK - EXCLUSIVE ENTRY      @LBC
&REL(48)     SETA 556         IXLDS LOCK                         @LBC
&REL(49)     SETA 604         IXLSHELL LOCK                      @LBA
&REL(50)     SETA 652         IXLREQST LOCK                      @01A
&REL(51)     SETA 700         WLMQ LOCK - SHARED ENTRY           @LCA
&REL(52)     SETA 748         WLMQ LOCK - EXCLUSIVE ENTRY        @LCA
&REL(53)     SETA 796         WLMRES LOCK                        @LCA
&REL(54)     SETA 844         REGSRV LOCK - SHARED ENTRY         @LCA
&REL(55)     SETA 892         REGSRV LOCK - EXCLUSIVE ENTRY      @LCA
&REL(56)     SETA 940         CONTEXT LOCK                       @LCA
&REL(57)     SETA 988         SSD LOCK                           @LDA
&REL(58)     SETA 1036        CMSLATCH LOCK                      @LEA
&REL(59)     SETA 1072        GRSINT LOCK - SHARED ENTRY         @LFA
&REL(60)     SETA 1120        GRSINT LOCK - EXCLUSIVE ENTRY      @LFA
&REL(61)     SETA 1168        MISC lock                          @LJC
&REL(62)     SETA 1216        DONOTUS2 lock                      @LJC
&REL(63)     SETA 1264        DONOTUS3 lock                      @LJC
&REL(64)     SETA 1312        DONOTUS4 lock                      @LJC
&REL(65)     SETA 1360        DONOTUS5 lock                      @LJC
&REL(66)     SETA 1408        HCWDRLK2 lock                      @LGA
&REL(67)     SETA 1456        HCWDRLK1 lock                      @LGA
&REL(68)     SETA 1504        SRMENQ LOCK - SHARED ENTRY         @LHA
&REL(69)     SETA 1552        SRMENQ LOCK - EXCLUSIVE ENTRY      @LHA
.* Entries 70-74 are SETLOCKP only
&REL(75)     SETA 1552+6*48   RSMQ LOCK                          @LKA
&REL(76)     SETA 1552+7*48   SSDGROUP lock                      @LMA
```

Diagnosing an Early Unlock Condition - Disabled Releases

```
.*
.*****************************************************************
.*                                                               *
.*       RELEASE DISABLED LIT OFFSETS (DECIMAL)                  *
.*                                                               *
.*****************************************************************
.*
&RELD(1)     SETA 36          DISPATCHER LOCK                     @L8A
&RELD(2)     SETA 84          RSMDS LOCK                          @L8A
&RELD(3)     SETA 132         IOSUCB LOCK                         @L8A
&RELD(4)     SETA 180         IOSLCH LOCK - NO LONGER SUPPORTED   @L8A
&RELD(5)     SETA 228         IOSYNCH LOCK                        @L8A
&RELD(6)     SETA 276         TPNCB LOCK - NO LONGER SUPPORTED    @L8A
&RELD(7)     SETA 324         TPDNCB LOCK - NO LONGER SUPPORTED   @L8A
&RELD(8)     SETA 372         TPACBDEB LOCK                       @L8A
&RELD(9)     SETA 420         ASM LOCK                            @L8A
&RELD(10)    SETA 468         SALLOC LOCK                         @L8A
&RELD(11)    SETA 516         SRM LOCK                            @L8A
&RELD(12)    SETA -1          CMS LOCK - DISABLED NOT SUPPORTED   @L8A
&RELD(13)    SETA -1          LOCAL LOCK - DISABLED NOT SUPPORTED @L8A
&RELD(14)    SETA 600         SPIN - RELEASE ALL SPIN LOCKS HELD  @L8A
&RELD(15)    SETA -1          CMSEQDQ LOCK - DISABLED NOT SUPPORTED@L8A
&RELD(16)    SETA -1          CMSALL - DISABLED NOT SUPPORTED     @L8A
&RELD(17)    SETA -1          ALL CMS LOCKS HELD- DISABLED NOT    @L8A
.*                            SUPPORTED
&RELD(18)    SETA -1          CMSSMF LOCK - DISABLED NOT SUPPORTED @L8A
&RELD(19)    SETA -1          CML LOCK - DISABLED NOT SUPPORTED   @L8A
&RELD(20)    SETA 792         TRACE LOCK - SHARED ENTRY           @L8A
&RELD(21)    SETA 840         TRACE LOCK - EXCLUSIVE ENTRY        @L8A
&RELD(22)    SETA 888         VSMPAG LOCK                         @L8A
&RELD(23)    SETA 936         RSM LOCK - SHARED ENTRY             @L8A
&RELD(24)    SETA 984         RSM LOCK - EXCLUSIVE ENTRY          @L8A
&RELD(25)    SETA 1032        RSMAD LOCK                          @L8A
&RELD(26)    SETA 1080        RSMXM LOCK                          @L8A
&RELD(27)    SETA 1128        RSMST LOCK                          @L8A
&RELD(28)    SETA 1176        ASMGL LOCK                          @L8A
&RELD(29)    SETA 1224        VSMFIX LOCK                         @L8A
&RELD(30)    SETA 1272        RSMGL LOCK                          @L8A
&RELD(31)    SETA 1296        CPU LOCK                            @L8A
&RELD(32)    SETA 1320        ALL - RELEASE ALL LOCKS HELD        @L8A
&RELD(33)    SETA 1344        (REG) - RELEASE LOCKS SPECIFIED     @L8A
&RELD(34)    SETA 1392        RSMCM LOCK                          @L8A
&RELD(35)    SETA 1456        IOS LOCK - SHARED ENTRY             @L8A
&RELD(36)    SETA 1504        IOS LOCK - EXCLUSIVE ENTRY          @L8A
&RELD(37)    SETA 36          BMFLSD LOCK                         @L7A
&RELD(38)    SETA 84          ETRSET LOCK                         @L6A
&RELD(39)    SETA 132         XCFDS LOCK                          @L5A
&RELD(40)    SETA 180         XCFRES LOCK                         @L5A
&RELD(41)    SETA 228         XCFQ LOCK - SHARED ENTRY            @L5A
&RELD(42)    SETA 276         XCFQ LOCK - EXCLUSIVE ENTRY         @L5A
&RELD(43)    SETA 328         IOSULUT LOCK - SHARED ENTRY         @L9A
&RELD(44)    SETA 376         IOSULUT LOCK - EXCLUSIVE ENTRY      @L9A
&RELD(45)    SETA 424         IXLSCH LOCK                         @LBC
&RELD(46)    SETA 472         IXLSHR LOCK - SHARED ENTRY          @LBC
&RELD(47)    SETA 520         IXLSHR LOCK - EXCLUSIVE ENTRY       @LBC
&RELD(48)    SETA 568         IXLDS LOCK                          @LBC
&RELD(49)    SETA 616         IXLSHELL LOCK                       @LBA
&RELD(50)    SETA 664         IXLREQST LOCK                       @01A
&RELD(51)    SETA 712         WLMQ LOCK - SHARED ENTRY            @LCA
&RELD(52)    SETA 760         WLMQ LOCK - EXCLUSIVE ENTRY         @LCA
&RELD(53)    SETA 808         WLMRES LOCK                         @LCA
&RELD(54)    SETA 856         REGSRV LOCK - SHARED ENTRY          @LCA
&RELD(55)    SETA 904         REGSRV LOCK - EXCLUSIVE ENTRY       @LCA
&RELD(56)    SETA 952         CONTEXT LOCKS                       @LCA
&RELD(57)    SETA 1000        SSD LOCKS                           @LDA
&RELD(58)    SETA -1          CMSLATCH LOCK - DISABLED NOT SUPPORTED   -
                                                                  @LEA
&RELD(59)    SETA 1084        GRSINT LOCK - SHARED ENTRY          @LFA
&RELD(60)    SETA 1132        GRSINT LOCK - EXCLUSIVE ENTRY       @LFA
&RELD(61)    SETA 1180        MISC lock                           @LJC
&RELD(62)    SETA 1228        DONOTUS2 lock                       @LJC
&RELD(63)    SETA 1276        DONOTUS3 lock                       @LJC
&RELD(64)    SETA 1324        DONOTUS4 lock                       @LJC
&RELD(65)    SETA 1372        DONOTUS5 lock                       @LJC
&RELD(66)    SETA 1420        HCWDRLK2 lock                       @LGA
&RELD(67)    SETA 1468        HCWDRLK1 lock                       @LGA
&RELD(68)    SETA 1516        SRMENQ LOCK - SHARED ENTRY          @LHA
&RELD(69)    SETA 1564        SRMENQ LOCK - EXCLUSIVE ENTRY       @LHA
.* Entries 70-74 are SETLOCKP only
&RELD(75)    SETA 1564+6*48   RSMQ LOCK                           @LKA
&RELD(76)    SETA 1564+7*48   SSDGROUP lock                       @LMA
```

Diagnosing an Early Unlock Condition

- By examining your assembly listings, you can determine the offset into the Lock Interface Table used for your lock OBTAIN and RELEASE.
- This is SETLOCK RELEASE TYPE=LOCAL:

```
0000065C 58E0 02FC        000002FC   59406+          L      14,PSALITA-PSA(0,0)
00000660 58DE 0254        00000254   59407+          L      13,588+8(14,0)
00000664 05ED                        59408+          BALR   14,13
```

- Issue 'ip w 2fc?+254?' in a dump:

```
ASID(X'003B') FEF8A0. IEANUC01.IEAVESLL+A0 IN READ ONLY NUCLEUS
```

- You would do the same for SETLOCK OBTAIN TYPE=LOCAL.
- Now that you know which addresses to trace, you can devise a plan of attack.

Diagnosing an Early Unlock Condition

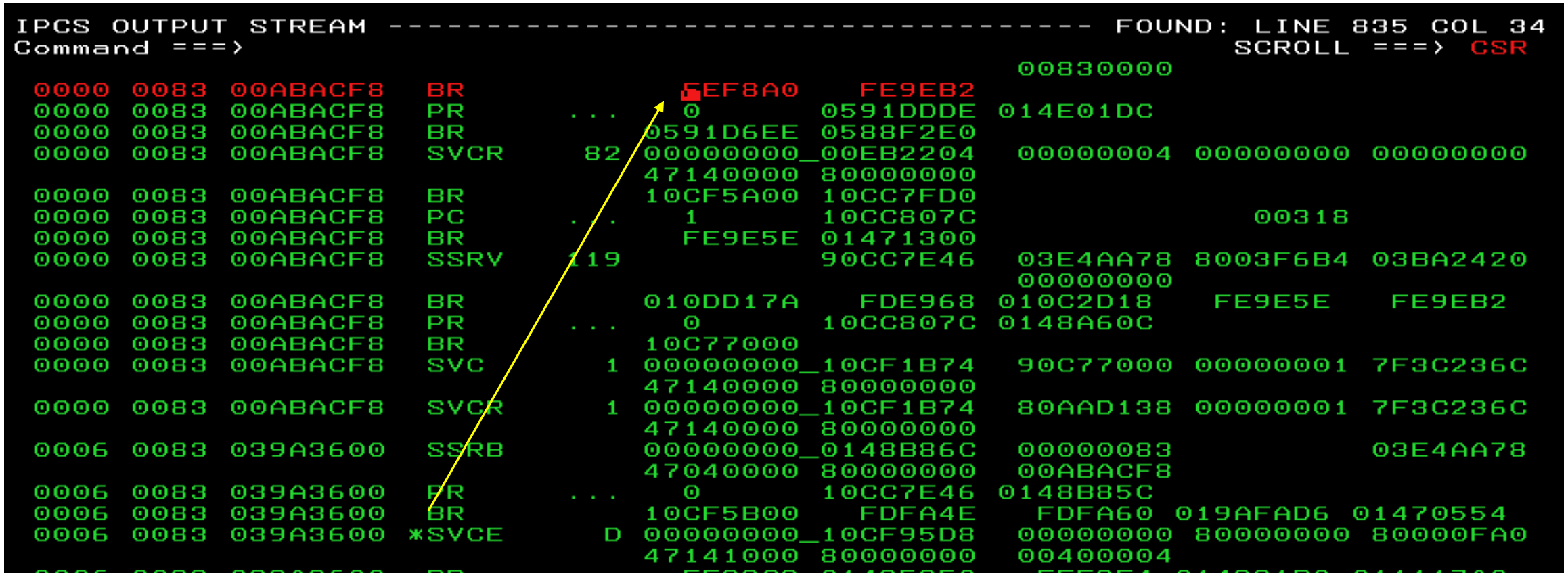
- Let's assume you're diagnosing a dump caused by calling a service without the local lock held. As far as the code is concerned, the lock was obtained and later mysteriously released.
- My approach would be to ask the reporting agency to enable branch tracing and reproduce the problem.

```
TRACE ST,BR=ON
IEE839I ST=(ON,0006M,00030M) AS=ON   BR=ON   EX=ON   MO=OFF MT=(ON,064K) 904
          ISSUE DISPLAY TRACE CMD FOR SYSTEM AND COMPONENT TRACE STATUS
          ISSUE DISPLAY TRACE,TT CMD FOR TRANSACTION TRACE STATUS
```

- Once you have the dump, you can determine from the LIT the addresses of the relevant lock obtain and release routines.
- Those addresses will appear in the SYSTRACE when branch tracing is active. Working backwards from the abend, you should be able to figure out who is calling the SETLOCK RELEASE erroneously.

Diagnosing an Early Unlock Condition

- A contrived example (the SVCE is not due to a missing lock).



- This technique of looking for a specific entry point address can be used to analyze the use of any branch-entered MVS service.

Acknowledgements Knowing and Unknowing

- Jim Mulder (IBM)
- Peter Relson (IBM)
- Roland Koo (IBM)
- Paul Scott (Phoenix Software)
- Peter Kania (Phoenix Software)

See you in Atlanta!