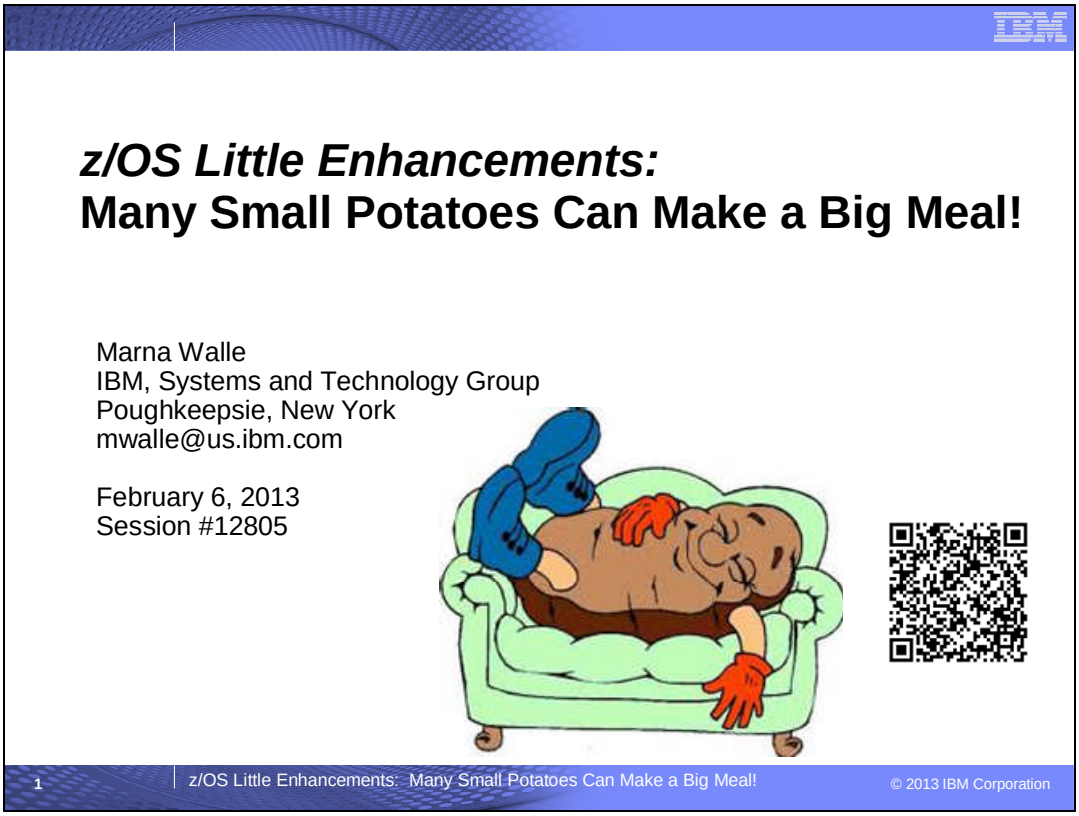


Abstract
There are a lot of little functions to make life easier for z/OS system programmers that you probably didn't even know about. These are not items that will make headlines or probably Redbooks, but if you need them, they could save time for a busy system programmer. The speaker will cover several z/OS functions in prior releases that you may not have even heard about such as: BEGINPARALLEL in IEFSSNxx, Timed Event Data Report, IDCAMS delete enhancements, and SDSF sorting. Most of these are probably available on the z/OS release you have right now!

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States, other countries, or both.

Not all common law marks used by IBM are listed on this page. Failure of a mark to appear does not mean that IBM does not use the mark nor does it mean that the product is not actively marketed or is not significant within its relevant market.

Those trademarks followed by ® are registered trademarks of IBM in the United States; all others are trademarks or common law marks of IBM in the United States.

For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

*, AS/400®, e business(logo)®, DBE, ESCO, eServer, FICON, IBM®, IBM (logo)®, iSeries®, MVS, OS/390®, pSeries®, RS/6000®, S/30, VM/ESA®, VSE/ESA, WebSphere®, xSeries®, z/OS®, zSeries®, z/VM®, System i, System i5, System p, System p5, System x, System z, System z9®, BladeCenter®

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:
Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Notice Regarding Specialty Engines (e.g., zIIPs, zAAPs and IFLs):

Any information contained in this document regarding Specialty Engines ("SEs") and SE eligible workloads provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the "Authorized Use Table for IBM Machines" provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html ("AUT").

No other workload processing is authorized for execution on an SE.

IBM offers SEs at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal ! Agenda

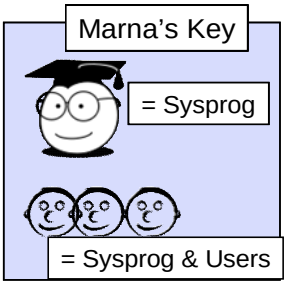
- **z/OS V1.13:**
  - **z/OS UNIX: Prevent Content Overlay during MOUNT**
  - **DFSMS: IEBPDSE Batch Program**
  - **SDSF: Cursor-sensitive Sort**
- **z/OS V1.12:**
  - **BCP: IEFSSNxx BEGINPARALLEL**
  - **BCP: Timed Event Data Report**
  - **BCP: Some PROGxx Goodies**
  - **DFSMS: IDCAMS DELETE All Members**
- **z/OS V1.11:**
  - **TSO/E: LOGONHERE reconnect support**
  - **BCP: D ALLOC and SETALLOC commands**
  - **z/OS UNIX: ALTROOT BPXPRMxx support**
- **Older than dirt on potatoes:**
  - **DFSMS: STGADMIN.DPDSRN.*oldname***
  - **z/OS UNIX: Dynamic Service Activation**
  - **BCP: IEARELEC to remove inactive EMCS consoles**
- **Summary**

IBM

z/OS R13 Enhancements

Small Enhancements of System Programmer Interest

- **z/OS UNIX:** Prevent Content Overlay during MOUNT
- **DFSMS:** IEBPDSE Batch Program
- **SDSF:** Cursor-sensitive Sort



z/OS R13 Enhancements

z/OS UNIX: Prevent Content Overlay during MOUNT

- **What:** Have you ever "lost" updates to files when they've been over-mounted? Do you want the system to let you know when you're mounting over something? Now you can have the system warn or even deny a mount on a mountpoint that is not empty.
- **How to use:** Specify WARN or DENY (default is NOWARN, same as today's behavior) on BPXPRMxx or on SETOMVS NONEMPTYMOUNTPT statement. `D OMVS,OPTIONS` shows current setting.
- **Considerations:** WARN produces a syslog message (not back to user). DENY goes back to the user.
  - Advanced consideration: Nonprivileged User Mount (in R13) must occur on an empty mountpoint (will act like a "DENY").

z/OS R13 Enhancements

z/OS UNIX: Prevent Content Overlay during MOUNT

- **WARN Usage Example:**

```
SETOMVS NONEMPTYMOUNTPT=WARN
BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
BPXO062I NONEMPTYMOUNTPT WAS CHANGED FROM NOWARN TO WARN.
```

Then do a mount:

```
READY
MOUNT FILESYSTEM('IBMUSER.PRODUCT.ZFS')
  MOUNTPOINT('/samples/') TYPE(ZFS)    MODE(RDWR)
READY
```

User saw no warning here!

However, in the syslog:

```
0290  BPXF263I FILE SYSTEM 901
0290  IBMUSER.PRODUCT.ZFS
0290  HAS BEEN MOUNTED ON A NONEMPTY DIRECTORY
```

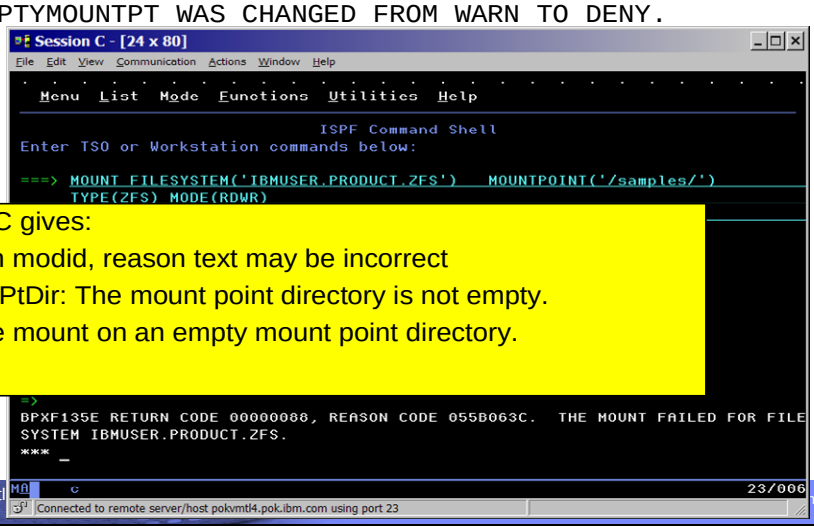
z/OS R13 Enhancements

z/OS UNIX: Prevent Content Overlay during MOUNT

- **DENY Usage Example:**

```
SETOMVS NONEMPTYMOUNTPT=DENY
BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
BPXO062I NONEMPTYMOUNTPT WAS CHANGED FROM WARN TO DENY.
```

Then do a mount:



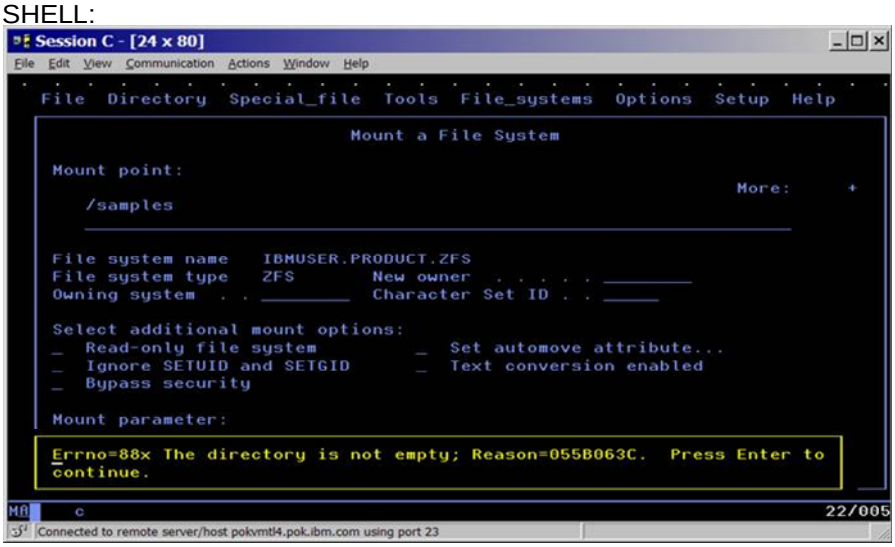
A bpxmtext 063C gives:
Notice: unknown modid, reason text may be incorrect
JrNonEmptyMntPtDir: The mount point directory is not empty.
Action: Retry the mount on an empty mount point directory.
***

z/OS R13 Enhancements

z/OS UNIX: Prevent Content Overlay during MOUNT

- **DENY Usage Example:**

Or from ISHELL:



z/OS UNIX: Prevent Content Overlays During MOUNT

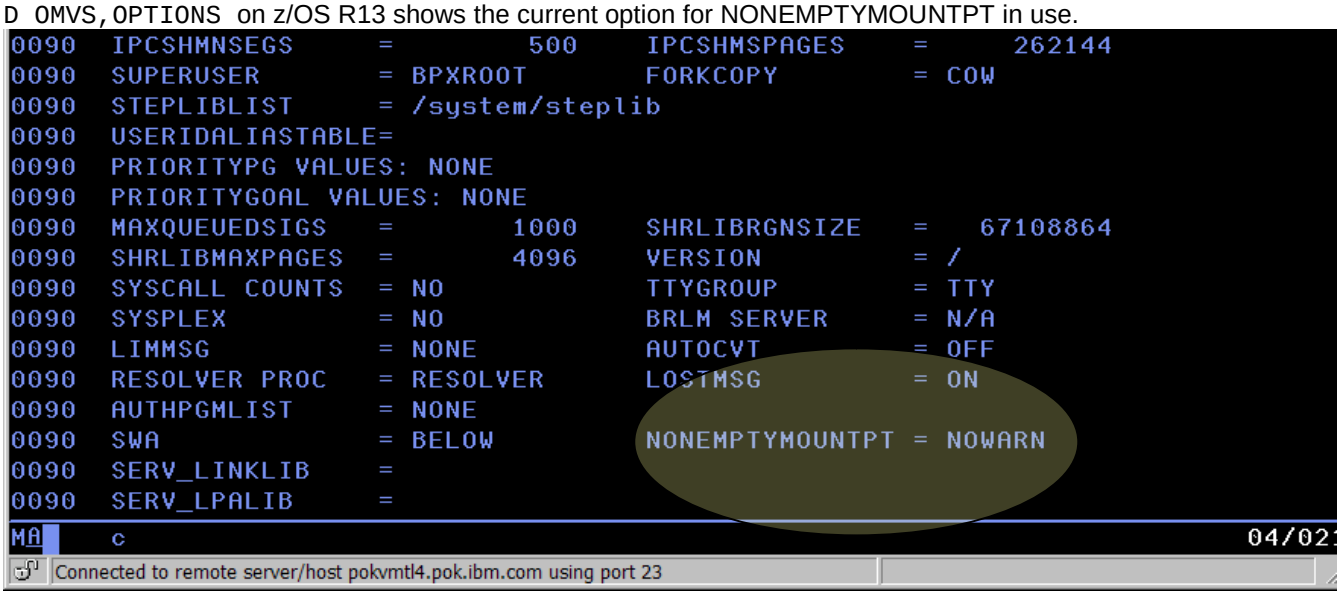
The BPXPRMxx parmlib statement NONEMPTYMOUNTPT can be used to control how the system mounts the file systems on the non-empty mount points.

- The NOWARN option specifies that the mount is to take place without any warning message when the mount point is a non-empty directory. The contents of that directory are hidden for the duration of the mount.
- The WARN option specifies that the mount is to take place with a warning message when the mount point is a non-empty directory. The contents of that directory are hidden for the duration of the mount.
- The DENY option specifies that mounting is not to take place when the mount point is a non-empty directory.

During OMVS initialization, if the mount point is contained in an NFS file system, the NONEMPTYMOUNTPT setting is not honored.

If you use the Nonprivileged User Mount function (introduced in z/OS R13), those mounts must be at an empty mountpoint. That is, a nonprivileged user may never mount on a non-empty mountpoint.

`D OMVS,OPTIONS` on z/OS R13 shows the current option for NONEMPTYMOUNTPT in use.



z/OS R13 Enhancements

DFSMS: IEBPDSE Batch Program

- **What:** Want to check the structural integrity of a PDSE? It's a good idea to verify a PDSE's integrity *before* it's deployed, and a problem would be propagated. A new utility can help you do that.
- **How to use:** Invoke PGM=IEBPDSE, with SYSLIB indicating your PDSEs to verify.
- **Considerations:** If the DUMP parameter option is specified, the PDSE validation utility issues an ABEND in the PDSE address space, which results on an SVC dump. Utility output is meant to tell you if the PDSE was valid or not – it does not correct the problem, nor is intended for you to correct the PDSE yourself. A dump would help IBM Service to diagnose the problem.
- **Usage Example:**

```
//STEPCHK EXEC PGM=IEBPDSE,PARM='DUMP'
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=SYS1.SIEALNKE,DISP=SHR
//       DD DSN=MWALLE.UTIL.JOBS,DISP=SHR
```

Successful output:

```
******************************** TOP OF DATA ***
IGW700I PDSE Directory Validation Successful
DSN:SYS1.SIEALNKE
ADPages:36 IXRecords:1054
NDPages:6 IXRecords:315
AD ND Tree Nodes:315
IGW700I PDSE Directory Validation Successful
DSN:MWALLE.UTIL.JOBS
ADPages:65 IXRecords:3985
NDPages:29 IXRecords:1590
AD ND Tree Nodes:1590
******************************** BOTTOM OF DATA *
```


z/OS R13 Enhancements

DFSMS: IEBPDSE Batch Program

- **Usage Example:**

Unsuccessful output:

```
******************************** TOP OF DATA ***
IGW702I PDSE Directory Validation Unsuccessful
DESC:<ND> Structure is corrupted
LTK:
00000000000000000000000000000000000000000000000000000000000000000000000000000000*
ERROR NUM:3
DSN:IBMUSER.PDSEVAL.PDSE1
VOLSER:338001
RC:8 RS:01198018 R14:040130A8
RPN:0
VPTVFN:N/A
IGW699I PDSE Directory Validation Unsuccessful
DESC:PDSE structure is corrupted
ERROR NUM:101
DSN:IBMUSER.PDSEVAL.PDSE1
VOLSER:338001
ADPages:1 IXRecords:14
NDPages:0 IXRecords:0
RC:8 RS:01198018
IGW702I PDSE Directory Validation Unsuccessful
DESC:<ND> Structure is corrupted
LTK:
00000000000000000000000000000000000000000000000000000000000000000000000000000000*
ERROR NUM:23
DSN:IBMUSER.PDSEVAL.PDSE1
VOLSER:338001
RC:8 RS:01198018 R14:04012F8E
RPN:0
     VPTVFN:N/A
     ******************************** BOTTOM OF DATA *
```

DFSMS: IEBPDSE (PDSE Validation) Program

You can use IEBPDSE to validate a PDSE data set and determine whether it is valid or corrupted.

IEBPDSE uses the following input:

- A PDSE data set, to be validated.

IEBPDSE produces the following output:

- A message data set that contains informational messages (for example if the data set was found to be corrupted), the results of the validation check, and error messages.

IEBPDSE is controlled by job control statements. Utility control statements are not used.

A PARM keyword may be specified: **PARM [DUMP| NODUMP]** If the DUMP option is specified, the PDSE validation utility issues an ABEND in the PDSE address space, which results on an SVC dump.

IEBPDSE Return Codes

IEBPDSE returns a code in register 15 to indicate the results of program execution. The return codes and their meanings are:

Codes Meaning

00 (X'00') Successful completion.
04 (X'04') The input PDSE is slightly damaged. Processing continues.
08 (X'08') The input PDSE is corrupted. The utility ends.
12 (X'0C') The input PDSE could not be opened. The utility ends.
16 (X'10') The input data set is not a PDSE. The utility ends.

z/OS R13 Enhancements

SDSF: Cursor Sensitive Sort

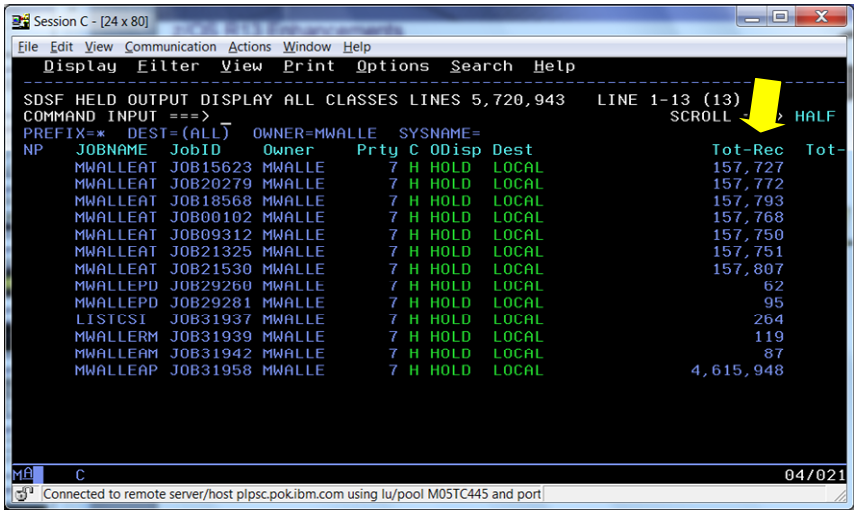
- **What:** Under ISPF, you now have "point and shoot" support. Meaning, you can sort on a column by tabbing the cursor to the column title and pressing Enter.
- **How to use:**
  - Tab to column title, hit Enter. Easy!
  - Enter toggles through ascending, descending, then original order.
- **Considerations:**
  - Can control it with SDSF command `SET CSORT OFF` or `ON`.
  - Make sure you have "Tab to point-and-shoot fields" enabled under ISPF Settings in Option 0.

z/OS R13 Enhancements

SDSF: Cursor Sensitive Sort

- **Usage Example:**

Before:

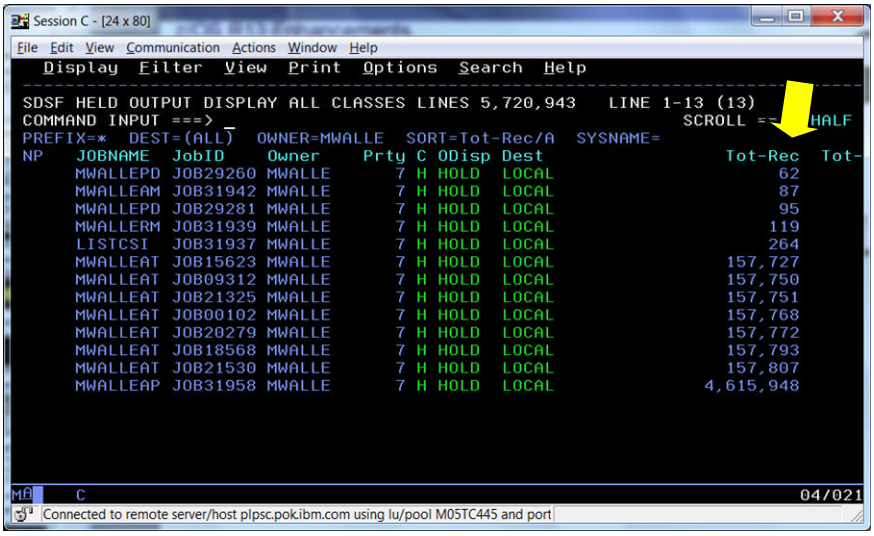


z/OS R13 Enhancements

SDSF: Cursor Sensitive Sort

- **Usage Example:**

First enter (ascending):

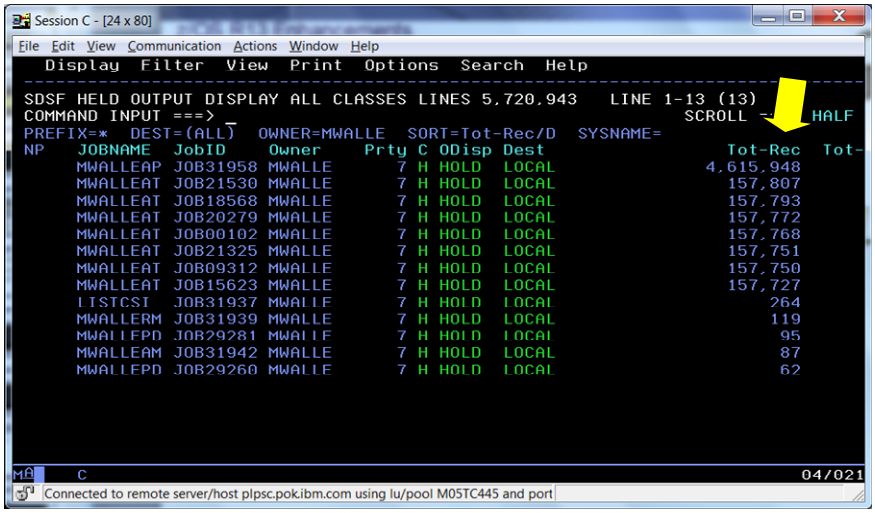


z/OS R13 Enhancements

SDSF: Cursor Sensitive Sort

- **Usage Example:**

Second enter (descending):

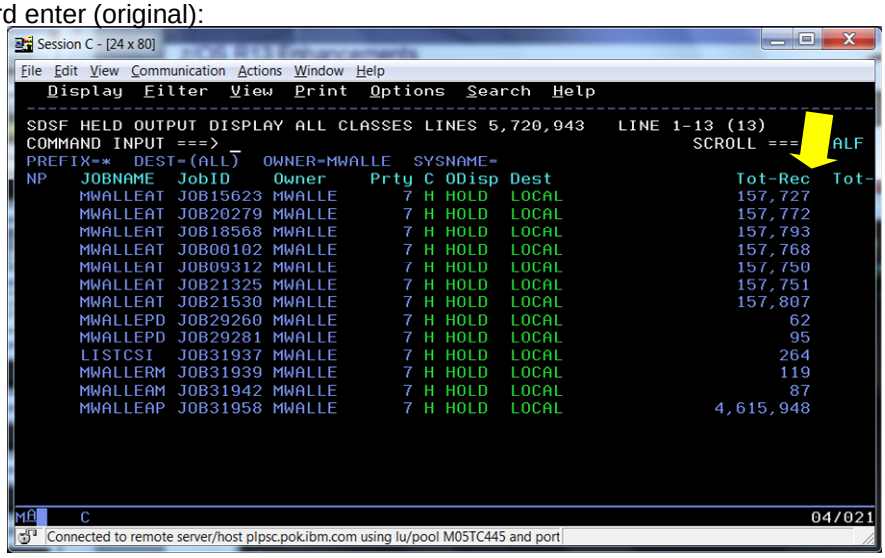


z/OS R13 Enhancements

SDSF: Cursor Sensitive Sort

- **Usage Example:**

Third enter (original):



15 | z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal! |

SDSF: Cursor-sensitive sort

You can now sort a tabular panel by placing the cursor on a column title and pressing Enter. Under ISPF, you can use the Tab key to move the cursor to the column titles. This is a quick alternative to typing the SORT command.

This function is so easy to use! You can see in the slide above an example of a tabular column (Tot-Rec) that I want to sort in ascending order, then descending order, and finally put back into the original order.

z/OS R12 Enhancements

Small Enhancements of System Programmer Interest

- **BCP:** IEFSSNxx BEGINPARALLEL

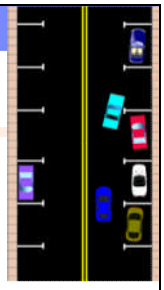
- **BCP:** Timed Event Data Report

- **BCP:** Some PROGxx Goodies

- **DFSMS:** IDCAMS DELETE All Members

z/OS R12 Enhancements

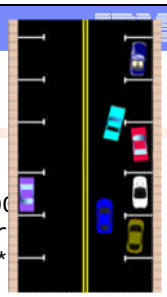
BCP: BEGINPARALLEL in IEFSSNxx



- **What:** In order to help with Mean Time To Recovery (MTTR), we want to reduce initialization paths where possible. With this in mind, you can specify that Subsystem Initialization Routines be run in parallel.
- **How to use:** Code **BEGINPARALLEL** in IEFSSNxx at the point in which you would like the SSI routines to be executed in parallel. Everything before **BEGINPARALLEL** will be executed serially, as before.
- **Considerations:**
  - The order that the parallelized initialization routines are run is now unpredictable. Therefore, the routines must have not any execution order dependencies.
  - SMS should be started before the BEGINPARALLEL statement. Also, Communications Server's VMCF and TNF subsystems should be specified before BEGINPARALLEL. Check with the subsystem product documentation to see if it should be placed before or after BEGINPARALLEL.
  - How much benefit you get will depend on how many initialization routines you have, the complexity of the routines, serialization requirements of routines, and available CPs. If a subsystem doesn't specify an initialization routine, then there is **no** effect on that subsystem.
  - Duplicate specifications of BEGINPARALLEL (within multiple concatenated IEFSSNxx members, for instance) result in subsequent specifications being rejected.

z/OS R12 Enhancements

BCP: BEGINPARALLEL in IEFSSNxx



- **Usage Example*:**

```
BROWSE    SYS1.PARMLIB(IEFSSNS2) - 01.25            Line 0000
 Command ===>                                          Scr
******************************** Top of Data ***************
SUBSYS SUBNAME(JES2) PRIMARY(YES) START(NO)
SUBSYS SUBNAME(SMS)  INITRTN(IGDSSIIN)
  INITPARM('ID=99,PROMPT=DISPLAY')
SUBSYS SUBNAME(BFL)  INITRTN(BFLSSI)
SUBSYS SUBNAME(JESA)
SUBSYS SUBNAME(JESB)
SUBSYS SUBNAME(TNF)  INITRTN(MVPTSSI)
SUBSYS SUBNAME(VMCF) INITRTN(MVPXSSI)  INITPARM(&SYSNAME.EIP)
BEGINPARALLEL
SUBSYS SUBNAME(CSQB) INITRTN(CSQ3INI) INITPARM('CSQ3EPX,!MQSB0,S')
SUBSYS SUBNAME(CSQD) INITRTN(CSQ3INI) INITPARM('CSQ3EPX,!MQSD0,S')
SUBSYS SUBNAME(CSQF) INITRTN(CSQ3INI) INITPARM('CSQ3EPX,!MQSF0,S')
SUBSYS SUBNAME(CSQH) INITRTN(CSQ3INI) INITPARM('CSQ3EPX,!MQSH0,S')
```

* on nine of our test systems in Poughkeepsie, we start 108 subsystem initialization routines in parallel each week!

BCP: IEFSSNxx BEGINPARALLEL

Before z/OS V1R12, the order in which the subsystems were defined depended on the order in which they were specified in the IEFSSNxx parmlib member on the SSN parameter. Beginning with z/OS V1R12, you can specify the BEGINPARALLEL statement that allows the initialization routines for any subsystem that supports parallel processing to be invoked in parallel. For the SMS subsystem or any subsystem that does not support parallel processing, be sure to specify the BEGINPARALLEL statement after you specify the subsystem definitions. For the SMS subsystem, if the BEGINPARALLEL statement is encountered before the SUBSYS statement, the system issues message IEF009E about potential errors.

```
SUBSYS      SUBNAME(subname)

            [CONSNAME(consname)]

            [INITRTN(initrtn)

            [INITPARM(initparm)]]

            [PRIMARY({NO|YES})

            [START({YES|NO})]]

BEGINPARALLEL
```

BEGINPARALLEL

The statement that indicates that the subsystem initialization routines specified in SUBSYS statements that follow the BEGINPARALLEL statement are invoked in parallel to reduce the amount of time it takes for all subsystems to initialize.

For subsystems that do not support parallel processing, you must ensure that the SUBSYS statement appears before the BEGINPARALLEL statement. For example, for the SMS subsystem definition IGDSSIN, be sure to specify the subsystem definitions before you specify the BEGINPARALLEL statement. For SMS, if you specify BEGINPARALLEL before the subsystem definition, the system issues message IEF009E about potential problems. Note that all initialization routines specified before the BEGINPARALLEL keyword are invoked serially, and all routines specified after the BEGINPARALLEL keyword are invoked in parallel. The BEGINPARALLEL keyword must be specified after the SMS entry.

For the z/OS Communications Server TNF and VMCF subsystems, you must specify the subsystem definitions before you specify the BEGINPARALLEL statement if the INITRTN parameter is included on the subsystem definitions. For information about starting the TNF and VMCF subsystems, see *z/OS Communications Server: IP Configuration Guide.*

If you do not specify BEGINPARALLEL, the subsystem initialization routines run serially, and you do not obtain any performance benefits of parallel processing. See the configuration or installation documentation of the subsystem for information about whether the subsystem initialization routines can support running in parallel.

You only need to specify one BEGINPARALLEL statement. If you specify more than one statement, the system issues a message indicating that it is using the first BEGINPARALLEL statement it finds and ignores any other statements. This is also true if you specify multiple concatenated IEFSSxx members with BEGINPARALLEL statements across a sysplex. This is what message you'll see:

```
ASA011I ERROR IN PARMLIB MEMBER=IEFSSNM2 ON LINE 1, POSITION 1: 213
DUPLICATE SPECIFICATION OF 'BEGINPARALLEL'.
FIRST SPECIFICATION IS USED.
DETECTING MODULE IS IEFJPACT. INPUT LINE: BEGINPARALLEL
```

z/OS R12 Enhancements

BCP: Timed Event Data Report

- **What:** A new authorized service, Timed Event Data Service (IEATEDS), allows a program to record events to a Timed Event Data Table: `REQUEST=REGISTER` and `REQUEST=RECORD.` (Hint: use and document a good description on your event!)
- IBM-supplied REXX exec, IEAVFTED, can be used to produce a Timed Event Data Report from the Timed Event Data Table in either a TSO or IPCS environment.
- As of z/OS R12, some components (XCF, GRS, JES2, and others) use the IEATEDS service. This should help you gather some Mean Time To Recovery statistics during IPL (such as the benefit of using the BEGINPARALLEL statement in IEFSSNxx).
  - –Other methods still help with gathering IPL statistics.
- **How to use:** IEAVFTED is compiled REXX, and requires the full REXX compiler run-time libraries (from the REXX Library product).
  - –Note that the REXX Alternate Runtime Library incorporated into the z/OS product is *not* sufficient!
- Several options on IEAVFTED, but most common may be the simple invocation from TSO/E of `===>` **`IEAVFTED DA('output_data_set')`**
  - –Where '**`output_data_set`**' must be the name of a pre-allocated data set with an LRECL of 512 and a RECFM of V or VB
- Reference information: *z/OS MVS Programming: Authorized Assembler Services Reference, Volume 2 (EDT-IXG)*

z/OS R12 Enhancements

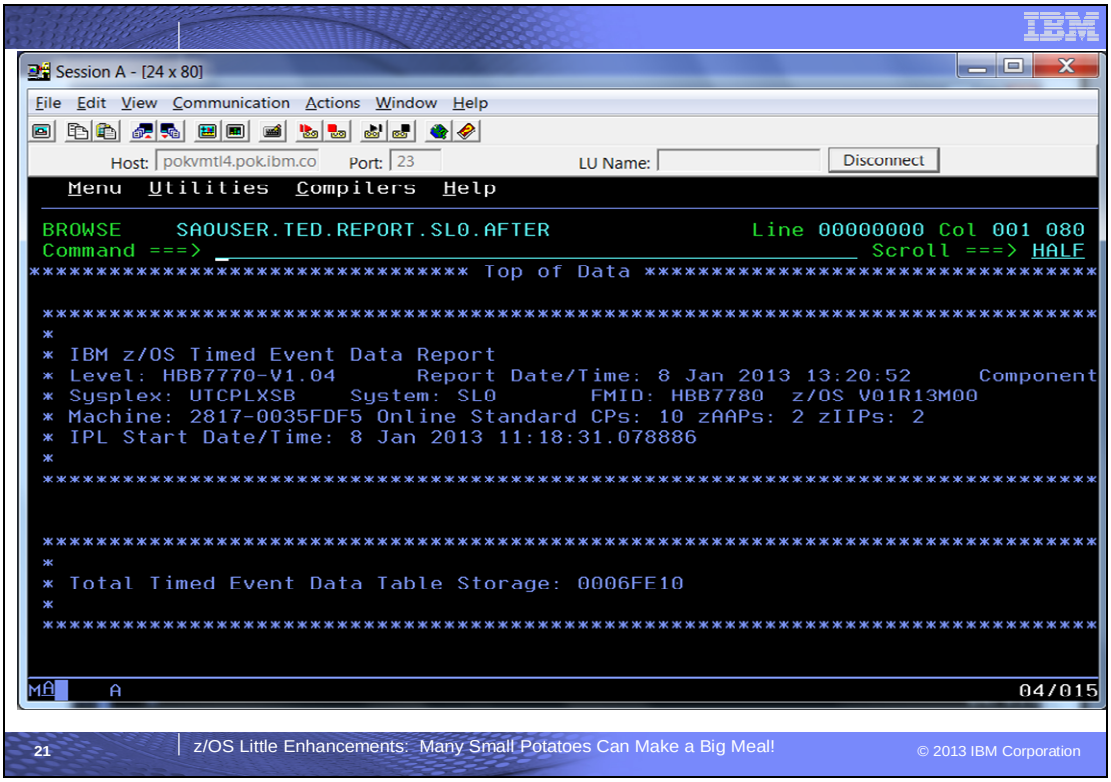
BCP: Timed Event Data Report

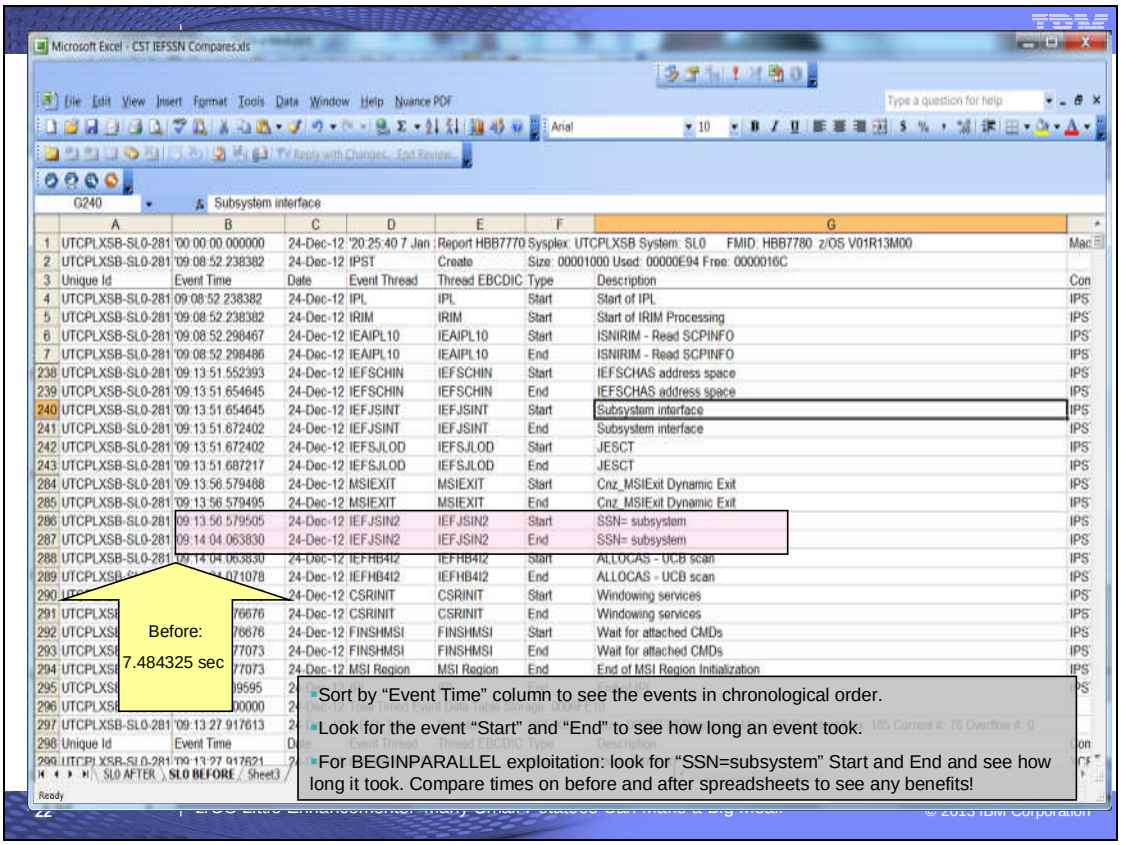
- **Considerations:** The IEAVFTED-produced report has two sections with headers: formatted entries, and spreadsheet data.
  - – You can use the spreadsheet portion to import in a spreadsheet program. That makes it easy to sort on data – such as quickly finding out which events took the longest time.
  - – Once you have the report downloaded:
    a) Delete everything in the top half (the formatted entries)
    b) Import to the spreadsheet program, indicating that the delimiter was a semicolon (or other delimiter that you indicated you wanted used when you created the report).
- **Usage Example:**
  1. Before using IEFSSNxx, from TSO: `===>` **`IEAVFTED DA('SAOUSER.TED.REPORT.SL0.BEFORE')`**

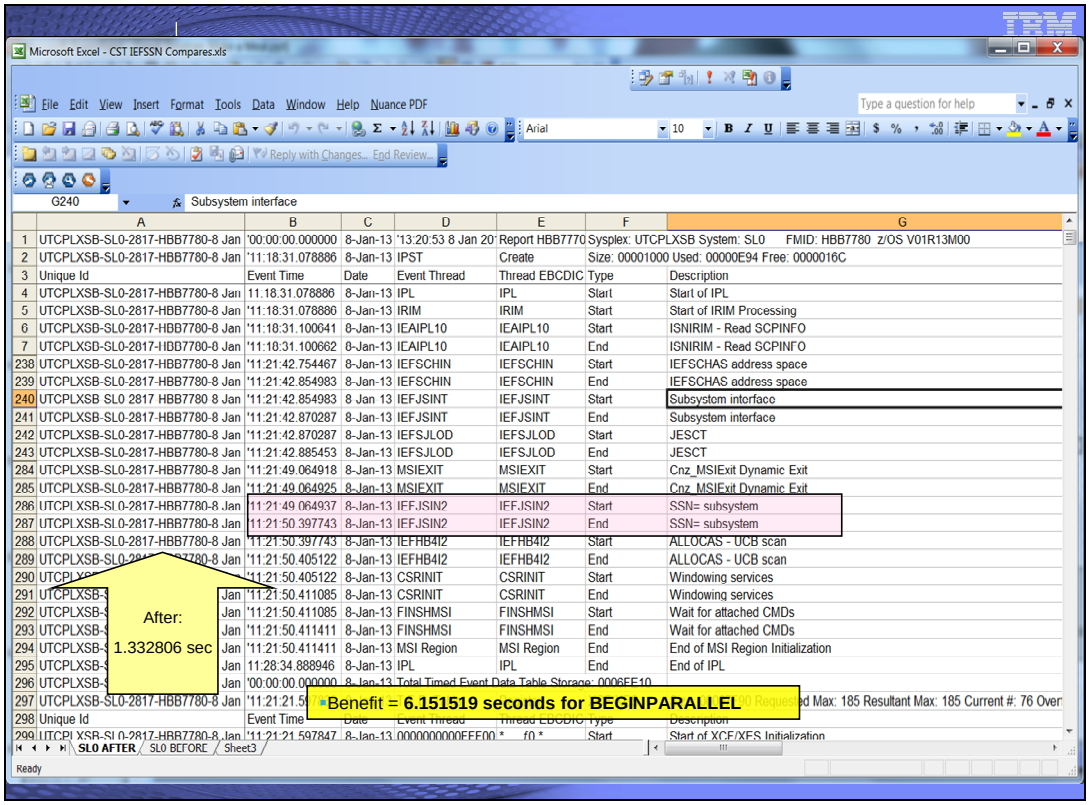
```
IEAVFTED Processing Started
IEAVFTED Processing Complete
***
```

  2. Incorporate BEGINPARALLEL in IEFSSNxx, and IPL at the next scheduled window
  3. After using IEFSSNxx, from TSO: `===>` **`IEAVFTED DA('SAOUSER.TED.REPORT.SL0.AFTER')`**

```
IEAVFTED Processing Started
IEAVFTED Processing Complete
***
```



Session A - [24 x 80]
File Edit View Communication Actions Window Help
Host: pokvmtl4.pok.ibm.co Port: 23 LU Name: Disconnect
Menu Utilities Compilers Help
BROWSE SAOUSER.TED.REPORT.SL0.AFTER Line 00000000 Col 001 080
Command ===> Scroll ===> HALF
Top of Data
* IBM z/OS Timed Event Data Report
* Level: HBB7770-V1.04 Report Date/Time: 8 Jan 2013 13:20:52 Component
* Sysplex: UTCPLXSB System: SL0 FMID: HBB7780 z/OS V01R13M00
* Machine: 2817-0035FDF5 Online Standard CPs: 10 zAAPs: 2 zIIPs: 2
* IPL Start Date/Time: 8 Jan 2013 11:18:31.078886
* Total Timed Event Data Table Storage: 0006FE10
MA A 04/015
21
z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal!
© 2013 IBM Corporation


Microsoft Excel - CST IEFSSN Compares.xls
G240 Subsystem interface
Before:
7.484325 sec
Sort by "Event Time" column to see the events in chronological order.
Look for the event "Start" and "End" to see how long an event took.
For BEGINPARALLEL exploitation: look for "SSN=subsystem" Start and End and see how long it took. Compare times on before and after spreadsheets to see any benefits!



BCP: Timed Event Data Report

Timed Event Data Service allows users to record events to a Timed Event Data Table to help determine program flow and performance. Each event is time-stamped and stored with data collected by the service as well as optional data provided by the user. A new macro, IEATEDS, is used to invoke the REGISTER service to establish a Timed Events Data Table for the user, and a RECORD service to add events to the table. A new REXX Exec, IEAVFTED, is provided to format the data in the table as a Timed Event Data Report and write it to a data set or z/OS UNIX file. The Timed Event Data Report includes a human-readable section and a spread sheet section that can be imported into a spread sheet program for analysis. Using the IEATEDS service, users will be able to RECORD events at various places in their code to determine elapsed time performance, and the optional data can be used to provide context, such as loop counters and return codes. By using the Timed Event Data service, users can determine how factors such as code changes, tuning changes, and configuration changes affect elapsed time performance. The IEATEDS macro contains a complete description of the service along with several examples.

Note: The IEAVFTED REXX exec is a compiled REXX program which requires the full REXX compiler run-time libraries (the REXX Library Product) installed before attempting to use IEAVFTED. Note that IEAVFTED will not work with the REXX Alternate Runtime Library z/OS Base (which is FMID HWJ9143).

The IEAVFTED report is especially helpful if you want to see the savings you got by exploiting IEFSSNxx BEGINPARALLEL. Use the procedure in the slide above to determine your benefit.

There are other commands and tools that you may be familiar with using for determine execution time in certain areas. The program IPLSTATS is mentioned heavily in the Redbook *System z Mean Time to Recovery Best Practices.*

Reference information: *z/OS MVS Programming: Authorized Assembler Services Reference, Volume 2 (EDT-IXG)*

z/OS R12 Enhancements

BCP: Some PROGxx Goodies

- **What:** Prior to z/OS R12, when you did an LPA ADD you had to also remember to include all the aliases for the module. As of z/OS R12, you can use ADDLIAS (not the default) to automatically include the aliases.
- **Usage Example:**
  - **SETPROG LPA,ADD,MODNAME=(CNMCNETV),DSNAME=MWALLE.MY.LPAMODS,ADDALIAS**
  - **Output:**

```
CSV551I 10.42.50 LPA ADD 925
SUCCESSFUL: 2  UNSUCCESSFUL: 0  NOT PROCESSED: 0
MODULE     RESULT
CNMCNETV   SUCCESSFUL
CNMNETV    SUCCESSFUL
```

- **What:** Prior to z/OS R12, when you wanted to replace a Dynamic Exit routine, you had to do a DELETE and then an ADD. This meant for some period of time that exit routine was not available on the system. As of z/OS R12, you can use REPLACE so that the exit routine is not unavailable for that period of time.
- **Considerations:** You can't REPLACE an exit routine that hadn't been ADD'd. In other words, make sure the exit routine has been ADD'd before trying to do a REPLACE.
- **Usage Example:**
  - **SETPROG EXIT,REPLACE,EXITNAME=IRREVX01,MODNAME=IRREVX1A,DSNAME=COMMON.LOOKFEEL.LINKLIB**
  - **Output:**

```
CSV420I MODUL IRREVX1A HAS BEEN REPLACED FOR EXIT IRREVX01
```


z/OS R12 Enhancements

BCP: Some PROGxx Goodies

- **What:** Prior to z/OS R12, it was easy to make common mistakes on Dynamic LPA and LNKLST in PROGxx and SETPROG, and the DISPLAY PROG,EXIT command. As of z/OS R12, you can use the DEFAULTS statement in PROGxx to make it less error-prone.
- **Considerations:** **`D PROG,DEFAULTS`** shows what you currently have
- **Usage Example:**
  - **To your PROGxx member add your preferences:**

```
DEFAULTS LPA     ADDALIAS
DEFAULTS LNKLST  REQCOPYFROM COPYFROMCUR
DEFAULTS EXIT    EXITTYPE(INSTALLATION)                    (or ALL, NOTPROGRAM)
```

  - **SETPROG LPA,ADD,MODNAME=(CNMCNETV),DSNAME=MWALLE.MY.LPAMODS**
    - **Output:**

```
CSV551I 10.42.50 LPA ADD 925
SUCCESSFUL: 2  UNSUCCESSFUL: 0  NOT PROCESSED: 0
MODULE     RESULT
CNMCNETV   SUCCESSFUL
CNMNETV    SUCCESSFUL
```

  - **SETPROG LNKLST DEFINE NAME(MARNALL)**
    - **Output: the MARNALL LNKLST set was copied from the current LNKLST.**
  - **D PROG,EXIT**
    - **Output: same as the D PROG,EXIT,INSTALLATION output.**

BCP: Some PROGxx Goodies

Note: There were several PROGxx-related enhancements in z/OS R12. Only a few are included below.

Dynamic LPA Enhancements

Automatic alias processing option in PROGxx and on CSVDYLPA

By default, LPA module alias names are not automatically handled. By default, if a module has aliases, the module name and all associated aliases must be specified within the same request. Otherwise, one of the following outcomes could occur, depending upon the initial state of the system:

- The module name or alias will not be found
- A duplicate copy of the same module will be loaded
- A previous copy of the module will be used.

You can use the ADDALIAS option to indicate that the system is to process aliases of the specified modules.

For support in PROGxx, the added syntax on LPA ADD is:
ADDALIAS | NOADDALIAS : ADDALIAS, or ALIAS, indicates to process provided names and aliases of the provided names. NOADDALIAS, or NOALIAS, indicates to process only the names provided. You can use NOADDALIAS to override the default values set by DEFAULTS LPA ADDALIAS.
Default Value: NOADDALIAS, or the value set by DEFAULTS LPA ADDALIAS | NOADDALIAS.

For support on CSVDYLPA REQUEST=ADD:
ADDALIAS=NO : Indicates not to add aliases, but only process the input names provided. The default is ADDALIAS=NO.
ADDALIAS=YES : When BYADDR=NO and MODINFOTYPE=MEMBERLIST are specified, ADDALIAS is an optional parameter that indicates whether to add aliases of the input names. Note: When adding is being done from a z/OS UNIX file, which is indicated by the PATHNAME keyword, ADDALIAS=YES is supported but cannot find aliases because this construct does not exist for z/OS UNIX files.

For support on SETPROG LPA,ADD command:
ADDALIAS | NOADDALIAS : ADDALIAS, or ALIAS, indicates to process provided names and aliases of the provided names. NOADDALIAS, or NOALIAS, indicates to process only the names provided. You can use NOADDALIAS to override the default values set by DEFAULTS LPA ADDALIAS.
Default Value: NOADDALIAS, or the value set by DEFAULTS LPA ADDALIAS | NOADDALIAS.

Exploitation Example:
Here is an example BEFORE using this support (the LPA module CNMCNETV has an alias of CNMNETV for this test):

```
SETPROG LPA,ADD,MODNAME=(CNMCNETV),DSNAME=MWALLE.MY.LPAMODS
IEF196I IEF237I 0980 ALLOCATED TO SYS00003
IEF196I IEF285I   MWALLE.MY.LPAMODS                          KEPT
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV551I 10.33.34 LPA ADD 902
SUCCESSFUL: 1  UNSUCCESSFUL: 0  NOT PROCESSED: 0
MODULE    RESULT
CNMCNETV  SUCCESSFUL
D PROG,LPA,MOD=CNMCNETV
CSV550I 10.34.06 LPA DISPLAY 904
FLAGS  MODULE    ENTRY PT  LOAD PT   LENGTH    DIAG
 D P   CNMCNETV  87DF3000  07DF3000  00000250  0243CD18
D PROG,LPA,MOD=CNMNETV
CSV550I 10.34.54 LPA DISPLAY 906
  CNMNETV WAS NOT FOUND IN THE LPA
```

Here is the same example AFTER using the support for ADDALIAS:

```
SETPROG LPA,ADD,MODNAME=(CNMCNETV),DSNAME=MWALLE.MY.LPAMODS,ADDALIAS
```

```
IEF196I IEF237I 0980 ALLOCATED TO SYS00005
IEF196I IEF285I   MWALLE.MY.LPAMODS                          KEPT
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV551I 10.42.50 LPA ADD 925
SUCCESSFUL: 2  UNSUCCESSFUL: 0  NOT PROCESSED: 0
MODULE    RESULT
CNMCNETV  SUCCESSFUL
CNMNETV   SUCCESSFUL
IEF196I IEF237I 0980 ALLOCATED TO SYS00052
IEF196I IEF285I   SYS1.LINKLIB                               KEPT
IEF196I IEF285I   VOL SER NOS= ZD112 .
D PROG,LPA,MOD=CNMCNETV
CSV550I 10.42.59 LPA DISPLAY 930
FLAGS  MODULE    ENTRY PT  LOAD PT   LENGTH    DIAG
 D P   CNMCNETV  87DF3000  07DF3000  00000250  0243CD18
D PROG,LPA,MOD=CNMNETV
CSV550I 10.43.05 LPA DISPLAY 932
FLAGS  MODULE    ENTRY PT  LOAD PT   LENGTH    DIAG
 D P   CNMNETV   87DF3000  07DF3000  00000250  020CF298
```

Dynamic Exit support for REPLACE

You can now replace a dynamic exit that has been added, instead of doing a DELETE and ADD.

For support in PROGxx, the added syntax is:

```
EXIT REPLACE
    EXITNAME(ex)
    MODNAME(mmmm)
   [STATE({ACTIVE|INACTIVE})]
   [DSNAME(dsn)]
```

For support on SETPROG EXIT command:

```
SETPROG EXIT REPLACE,EXITNAME=exitname,MODNAME=modname,
 [,STATE={ACTIVE|INACTIVE}]
 [,DSNAME=dsname]
```

Exploitation Example:

Here is an example of using this support. I decide to add module IRREVX1A for RACF dynamic exit IRREVX01 from one data set:

```
SETPROG EXIT,ADD,EXITNAME=IRREVX01,MODNAME=IRREVX1A,DSNAME=MWALLE.MY.DYNEXITS
IEF196I IEF237I 0980 ALLOCATED TO SYS00059
IEF196I IEF285I   MWALLE.MY.DYNEXITS                         KEPT
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV420I MODULE IRREVX1A HAS BEEN ADDED TO EXIT IRREVX01
D PROG,EXIT,EXITNAME=IRREVX01,DIAG
CSV464I 16.22.00 PROG,EXIT DISPLAY 878
EXIT IRREVX01
MODULE    STATE EPADDR    LOADPT    LENGTH    JOBNAME   PARAM
IRREVX1A    A   8876F010  0876F010  00000040  *
```

Now, I want to replace the module for this RACF dynamic exit to module IRREVX1A found in another data set:

```
SETPROG EXIT,REPLACE,EXITNAME=IRREVX01,MODNAME=IRREVX1A,
DSNAME=COMMON.LOOKFEEL.LINKLIB
IEF196I IEF237I 0980 ALLOCATED TO SYS00064
IEF196I IEF285I   COMMON.LOOKFEEL.LINKLIB                    KEPT
```

```
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV420I MODULE IRREVX1A HAS BEEN REPLACED FOR EXIT IRREVX01
D PROG,EXIT,EXITNAME=IRREVX01,DIAG
CSV464I 16.49.38 PROG,EXIT DISPLAY 280
EXIT IRREVX01
MODULE    STATE EPADDR    LOADPT    LENGTH    JOBNAME   PARAM
IRREVX1A    A   8861E5C8  0861E5C8  000001E0  *
```

New PROGxx DEFAULTS support

You can specify several options in PROGxx to control defaults the system is to use when processing LPA and LNKLST statements in PROGxx and the SETPROG commands, and what kinds of exits the DISPLY PROG,EXIT command will show.

For support in PROGxx, the added syntax is:

- `DEFAULTS LNKLST [REQCOPYFROM | NOREQCOPYFROM] [COPYFROMCUR | NOCOPYFROMCUR]`
  Indicates that this default applies to LNKLST statements in PROGxx and SETPROG LNKLST commands.
- `DEFAULTS LPA [ADDALIAS | NOADDALIAS]`
  Indicates that this default applies to LPA statements in PROGxx and SETPROG LPA commands.
- `DEFAULTS EXIT [ EXITTYPE({ALL | INSTALLATION | NOTPROGRAM}) ]` Indicates that this default applies to the DISPLAY PROG,EXIT command.

Exploitation Example:

I created a PROGMW in my parmlib concatenation:

```
EDIT       SYS1.PARMLIB.POK(PROGMW)
 Command ===>
 ****** ***************************** Top of Data *******
 000001 DEFAULTS LNKLST REQCOPYFROM COPYFROMCUR
 000002 DEFAULTS LPA    ADDALIAS
 000003 DEFAULTS EXIT   EXITTYPE(INSTALLATION)
 ****** **************************** Bottom of Data *****
```

I activate this PROGMW parmlib member where PROG00 and PROGAA were already in use on the system:

```
SET PROG=(MW,00,AA)
IEE252I MEMBER PROGMW   FOUND IN SYS1.PARMLIB.POK
CSV562I LNKLST DEFAULTS ARE SET TO 891
COPYFROMCUR, REQCOPYFROM
CSV563I LPA DEFAULTS ARE SET TO 892
ADDALIAS
CSV566I EXIT DEFAULTS ARE SET TO 893
DISPLAY EXITTYPE=INSTALLATION
IEE252I MEMBER PROG00   FOUND IN SYS1.PARMLIB.POK
…
```

An LPA example: Now, I "forget" to say ADDALIAS, on the same example above. Notice how the alias is automatically added:

```
SETPROG LPA,ADD,MODNAME=(CNMCNETV),DSNAME=MWALLE.MY.LPAMODS
IEF196I IEF237I 0980 ALLOCATED TO SYS00052
IEF196I IEF285I   MWALLE.MY.LPAMODS
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV551I 13.41.52 LPA ADD 125
SUCCESSFUL: 2  UNSUCCESSFUL: 0  NOT PROCESSED: 0
MODULE    RESULT
CNMCNETV  SUCCESSFUL
CNMNETV   SUCCESSFUL
```

A LNKLST example: Now, I "forget" to say COPYFROM(CURRENT)

```
D PROG,LNKLST
CSV470I 13.53.06 LNKLST DISPLAY 127
LNKLST SET LNKLST00   LNKAUTH=LNKLST
ENTRY  APF  VOLUME  DSNAME
   1    A   ZD112   SYS1.LINKLIB
   2        ZD112   SYS1.MIGLIB
   3        ZD112   SYS1.CSSLIB
   4    A   ZD112   SYS1.SIEALNKE
   5        ZD112   SYS1.SIEAMIGE
   …
   47       ZD112   COMMON.LOOKFEEL.LINKLIB
   48    A  ZD112   REXX.SEAGALT
SETPROG LNKLST DEFINE NAME(MARNALL)
CSV500I LNKLST SET MARNALL HAS BEEN DEFINED
SETPROG LNKLST ADD NAME(MARNALL) DSNAME(MWALLE.MY.DYNEXITS) ATBOTTOM
IEF196I IEF237I 0980 ALLOCATED TO SYS00053
IEF196I IEF285I   MWALLE.MY.DYNEXITS                          KEPT
IEF196I IEF285I   VOL SER NOS= ZD112 .
CSV501I DATA SET MWALLE.MY.DYNEXITS 134
HAS BEEN ADDED TO LNKLST SET MARNALL
SETPROG LNKLST ACTIVATE NAME(MARNALL)
IEF196I IEF237I 0980 ALLOCATED TO SYS00054
IEF196I IEF237I 0980 ALLOCATED TO SYS00055
IEF196I IEF237I 0980 ALLOCATED TO SYS00056
IEF196I IEF237I 0980 ALLOCATED TO SYS00057
…
CSV500I LNKLST SET MARNALL HAS BEEN ACTIVATED
```

And it really did COPYFROM(CURRENT):

```
D PROG,LNKLST
CSV470I 14.01.31 LNKLST DISPLAY 335
LNKLST SET MARNALL   LNKAUTH=LNKLST
ENTRY  APF  VOLUME  DSNAME
   1    A   ZD112   SYS1.LINKLIB
   2        ZD112   SYS1.MIGLIB
   3        ZD112   SYS1.CSSLIB
   4    A   ZD112   SYS1.SIEALNKE
   5        ZD112   SYS1.SIEAMIGE
   …
   47       ZD112   COMMON.LOOKFEEL.LINKLIB
   48    A  ZD112   REXX.SEAGALT
   49       ZD112   MWALLE.MY.DYNEXITS
```

An EXIT example: Prior to my PROGMW, the DISPLAY PROG EXITS would show me ALL exits. And if I wanted to just see INSTALLATION exits, I had to use DISPLAY PROG,EXITS,INSTALLATION:

```
D PROG,EXIT
CSV460I 12.56.33 PROG,EXIT DISPLAY 789
EXIT              DEF EXIT              DEF EXIT              DEF
CSVDYLPA           E  CSVDYNEX           E  HZSADDCHECK       I
IEASDUMP.QUERY     E  IEASDUMP.GLOBAL    E  IEASDUMP.LOCAL     E
IEASDUMP.SERVER    E  IEASDUMP.POSTDMP   E  IXC_ELEM_RESTART   E
IXC_WORK_RESTART   E  ISGNQXIT           E  ISGNQXITFAST       E
ISGCNFXITSYSTEM    E  ISGCNFXITSYSPLEX   E  ISGNQXITBATCH      E
ISGNQXITQUEUED1    E  ISGNQXITQUEUED2    E  ISGENDOFLQCB       E
ISGNQXITPREBATCH   E  ISGNQXITBATCHCND   E  ISGDGRSRES        E
```

```
CNZ_MSGTOSYSLOG   E  IEHINITT_EXIT     E  REKEY_EXIT        E
IEF_ALLC_OFFLN    E  IEF_SPEC_WAIT     E  IEF_VOLUME_ENQ    E
IEF_VOLUME_MNT    E  IEFDB401          E  IEF_ALLC_MOD      E
IEF_ALLC_EVENT    E  CEE_ABEND_EXIT    E  CNZ_WTOMDBEXIT    E
IEFJFRQ           E  SYSSTC.IEFU29     E  SYS.IEFU84        E
SYS.IEFUJV        E  SYS.IEFU29        E  SYS.IEFU83        E
SYS.IEFUJI        E  SYS.IEFACTRT      E  IRREVX01          E
IRRVAF01          E  IGDACSDX          E  BPX_PREPROC_INIT  E
BPX_POSPROC_INIT  E  BPX_IMAGE_INIT    E  BPX_PREPROC_TERM  E
CSVLLIX1          E  CSVLLIX2          E  HASP.$EXIT0       E
D PROG,EXIT,INSTALLATION
CSV460I 12.57.50 PROG,EXIT DISPLAY 793
EXIT             DEF EXIT             DEF EXIT             DEF
IXC_ELEM_RESTART  E  IXC_WORK_RESTART  E  SYSSTC.IEFU29     E
SYS.IEFU84        E  SYS.IEFUJV        E  SYS.IEFU29        E
SYS.IEFU83        E  SYS.IEFUJI        E  SYS.IEFACTRT      E
CSVLLIX1          E  CSVLLIX2          E  HASP.$EXIT0       E
```

After my PROGMW was activated, the DISPLAY PROG EXITS will show me INSTALLATION exits. Note that I can still see ALL exits by just doing a DISPLAY PROG,EXITS,ALL:

```
D PROG,EXIT
CSV460I 14.09.52 PROG,EXIT DISPLAY 347
EXIT             DEF EXIT             DEF EXIT             DEF
IXC_ELEM_RESTART  E  IXC_WORK_RESTART  E  SYSSTC.IEFU29     E
SYS.IEFU84        E  SYS.IEFUJV        E  SYS.IEFU29        E
SYS.IEFU83        E  SYS.IEFUJI        E  SYS.IEFACTRT      E
CSVLLIX1          E  CSVLLIX2          E  HASP.$EXIT0       E
```

z/OS R12 Enhancements

DFSMS: IDCAMS DELETE All Members

- **What:** Prior to z/OS R12, IDCAMS DELETE could only delete only one member at a time. You'd have to invoke the DELETE command for each member you wanted to delete. Now, you can delete all members at once!
- **Considerations:** Remember, IDCAMS DELETE MASK command (as of R11) allows you to delete more than one data set at a time.
- **Usage Example:**

```
//DELMEM   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
         DELETE MWALLE.TESTDEL.MEMS(*)
/*
```

- Output:

```
DELETE MWALLE.TESTDEL.MEMS(*)
IDC0553I ALL MEMBERS IN DATA SET MWALLE.TESTDEL.MEMS  DELETED
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
```

- **Note that on z/OS R12 and R13 there is no member "partial wildcard delete" support. For instance:**

```
//DELMEM   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
         DELETE MWALLE.TESTDEL.MEMS2(SMP*)
/*
```

- **Results in:**

```
DELETE MWALLE.TESTDEL.MEMS2(SMP*)
IDC3203I ITEM 'MWALLE.TESTDEL.MEMS2' DOES NOT ADHERE TO RESTRICTIONS
IDC3202I ABOVE TEXT BYPASSED UNTIL NEXT COMMAND. CONDITION CODE IS 12
```

26 | z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal! |

z/OS R11 Enhancements

Small Enhancements of System Programmer Interest

- **TSO/E: LOGONHERE reconnect support**
- **BCP: D ALLOC and SETALLOC commands**
- **z/OS UNIX: ALTROOT BPXPRMxx support**

27 | z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal! |

z/OS R11 Install-Related Enhancements

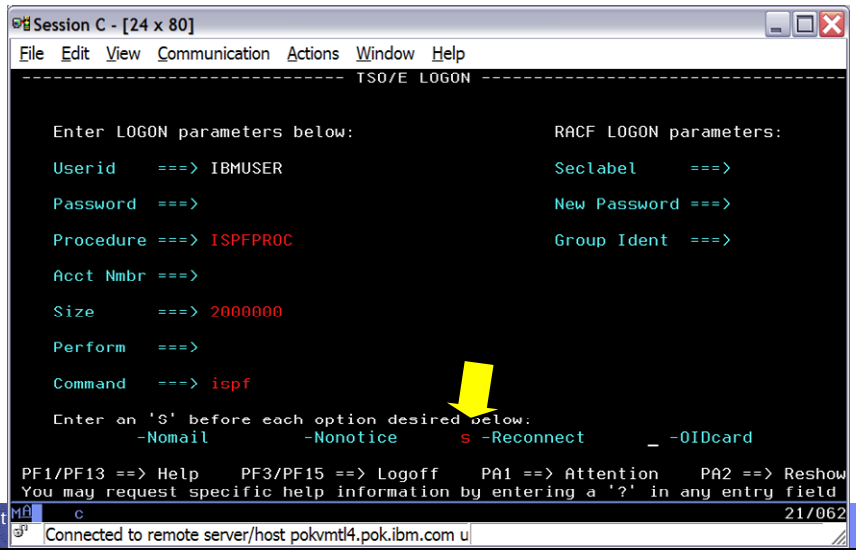
TSO/E: LOGONHERE reconnect support

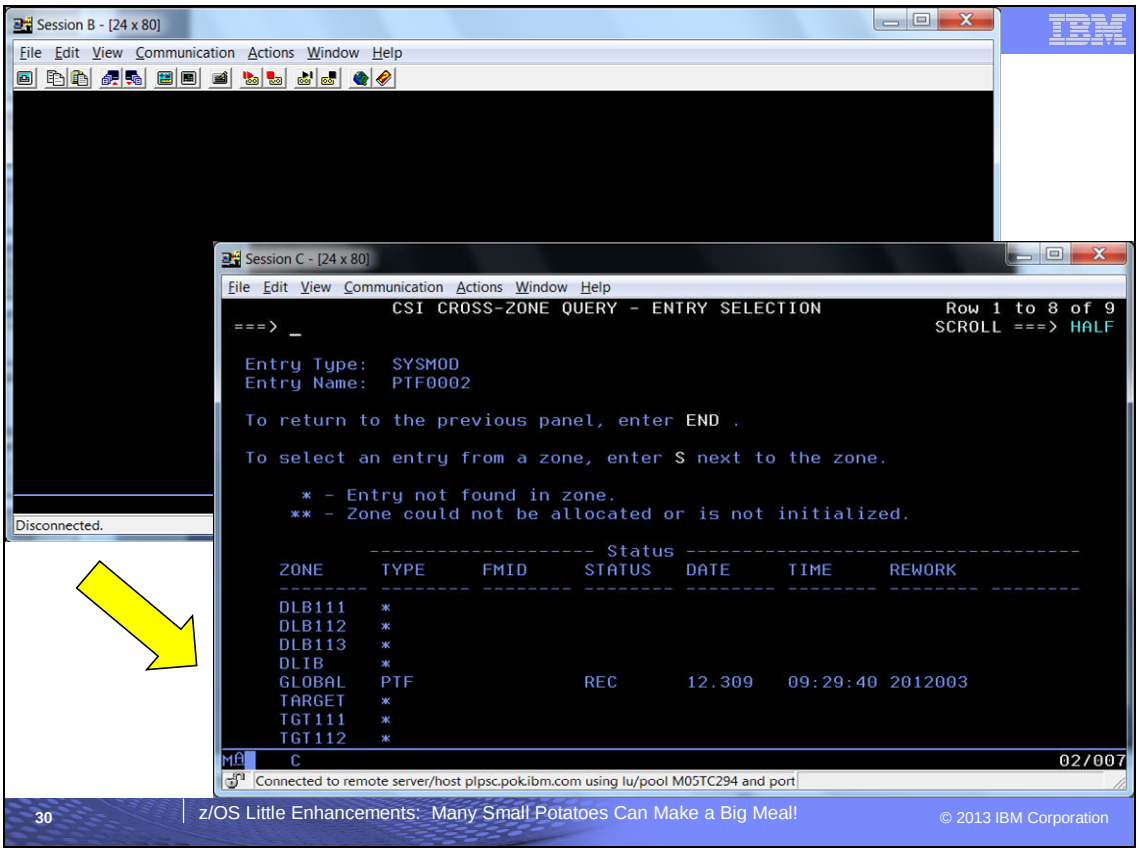
- **What:** Support for VTAM unconditional reconnect.
  - –Allows you to reconnect to your session even if no disconnection has been detected. You "resume" right were you were before.
  - –So easy to switch from one computer to another now! Hopefully should reduce the number of times that operators have to cancel TSO/E user IDs.
- **How to use:**
  - –Support is turned on by default, but you can turn it off:
    - –IKTSOxx LOGON statement `LOGONHERE(OFF)` then SET `IKJTSO=xx`, or
    - –TSO/E `PARMLIB UPDATE(xx)` command
  - –`D IKJTSO,LOGON` can tell you what you currently have
- **Considerations:** Verify your TSOKEYxx RECONLIM= setting to make sure it is non-zero. (Zero means a zero wait reconnection time, or reconnection is not possible.) RECONLIM default is 3 minutes.

z/OS R11 Install-Related Enhancements

TSO/E: LOGONHERE reconnect support

- **Usage Example:**
  - –Verify that TSOKEYxx RECONLIM isn't zero, then logon selecting "-Reconnect".





TSO/E LOGONHERE support for VTAM unconditional reconnect

TSO/E LOGONHERE support for VTAM unconditional reconnect now allows you to reconnect to your session even if no disconnection has been detected. By default as of z/OS V1R11, LOGONHERE support is turned on. By specifying the reconnect option, you can easily switch from one computer to another or reestablish a session after a loss of connectivity (even with a new IP address).

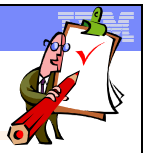
To control this, a new PARMLIB option LOGONHERE(ON/OFF) has been added under the LOGON statement in IKJTSOxx. The default is ON, which should reduce the number of times that operators have to cancel TSO/E user IDs. However, the old behavior can be restored by setting the value to OFF. Before z/OS V1R11, TSO/E LOGON RECONNECT would not always work. If the system could not detect that a TSO/E user ID was disconnected, it would tell users their ID was already in use.

> **IKJTSOxx LOGON statement:** Specifies the system settings for the TSO/E LOGON command:
> **LOGONHERE(ON|OFF)** Specifies whether the RECONNECT option on the TSO/E LOGON panel will be honored even when the system does not detect a disconnected state and the user appears to be logged on. This allows users to reconnect their session from a new terminal without canceling their previous session first, similar to how the LOGONHERE option works under z/VM.
> **Default:** ON

Hint: Verify your TSOKEYxx RECONLIM= setting to make sure it is non-zero. **RECONLIM** Specifies the time limit in minutes within which a user may reconnect after his TP line has been disconnected. Note that the default setting of RECONLIM=0 means that there is a zero wait for reconnection, which means that a reconnect is not possible. **Value Range:** 0-32767 **Default:** 3.

z/OS R11 Install-Related Enhancements

BCP: D ALLOC and SETALLOC commands



- **What:** You can now see your current ALLOCxx settings with a command, and change most of them dynamically. This is great for availability, since you don't need an IPL to change the settings.
  - Exception: policy for 2 digits years (2DGT_EXPDT)
- **How to use:** `SETALLOC` *`options`* and `D ALLOC,OPTIONS`
  - The *`options`* statement on the command is different than the parmlib syntax. For example command syntax: `SYSTEM,IEFBR14_DELMIGDS=NORECALL` vs. parmlib syntax: `SYSTEM IEFBR14_DELMIGDS(NORECALL)`
- **Considerations:** Note this is for changing specific settings, not the whole ALLOCxx parmlib member. It is not SET ALLOCxx!
  - The more dynamics we use, the more we need to ensure that we harden the newly desired values!

z/OS R11 Install-Related Enhancements

BCP: D ALLOC and SETALLOC commands

- **Usage Example:**
  - **`SETALLOC SYSTEM,IEFBR14_DELMIGDS=NORECALL,TAPELIB_PREF=BYDEVICES`**

```
IEFA010I SETALLOC COMMAND SUCCESSFUL 829
IEFBR14_DELMIGDS SET TO NORECALL.
TAPELIB_PREF SET TO BYDEVICES.
```

  - **`D ALLOC,OPTIONS`**

```
IEFA003I 16.16.38 ALLOC OPTIONS 832
…
SYSTEM          IEFBR14_DELMIGDS: NORECALL
                TAPELIB_PREF:     BYDEVICES
                REMIND_INTV:      90
                VERIFY_UNCAT:     FAIL
```

DISPLAY ALLOC and SETALLOC commands
Two new commands are introduced in V1R11: DISPLAY ALLOC and SETALLOC. The DISPLAY ALLOC command enables operators to determine what ALLOCxx settings are currently active; the SETALLOC command is used to dynamically modify Device Allocation parameters and settings without re-ILPing.

Displaying MVS Device Allocation Settings Information
Use the DISPLAY ALLOC,OPTIONS command to display either of the following:

- The current MVS Device Allocation settings that are in use, as set by the ALLOCxx parmlib member at IPL, or modified by the SETALLOC operator command.
- The system defaults, if no ALLOCxx member has been specified or no SETALLOC command has been processed.

`D ALLOC,OPTIONS [,L={a|name|name-a}]`
`OPTIONS` Indicates the categories and Allocation settings that are currently being used. Certain options are displayed only when they are applicable to the settings that the system is using.
`L=`*`a`*, *name, or name-a* Specifies the display area (*a*), console name (*name*), or both (*name-a*) where the display is to appear. If you omit this operand, the display is presented in the first available display area or the message area of the console through which you enter the command.

SETALLOC command
Use the SETALLOC command to dynamically modify Device Allocation settings.

```
SETALLOC {SPACE[,PRIMARY=n]
              [,SECONDARY=n]
          [,DIRECTORY=n]
         [,MEASURE={TRK|CYL|AVEBLK}]
         [,BLKLNGTH=n]
         [,ROUND={ROUND|NOROUND}]
         [,PRIM_ORG={CONTIG|MXIG|ALX}]
         [,RLSE={RLSE|NORLSE}] }
        {UNIT[,NAME=group]
        [,UNITAFF=unit]
        [,REDIRECTED_TAPE={TAPE|DASD}] }
        {TIOT,SIZE=n}
        {SDSN_WAIT,WAITALLOC={YES|NO}}
        {VOLUME_ENQ,POLICY={WTOR|CANCEL|WAIT}}
        {VOLUME_MNT,POLICY={WTOR|CANCEL}}
        {SPEC_WAIT[,POLICY={WTOR|CANCEL|WAITHOLD|WAITNOH}]
             [,MAXNWAIT=n]
             [,POLICYNW={WTOR|CANCEL}] }
        {ALLC_OFFLN[,POLICY={WTOR|CANCEL|WAITHOLD|WAITNOH}]
             [,MAXNWAIT=n]
             [,POLICYNW={WTOR|CANCEL}] }
        {CATLG_ERR[,FAILJOB={YES|NO}]
             [,ERRORMSG={YES|NO}] }
        {VERIFY_VOL,POLICY={YES|NO}}
        {SYSTEM[,IEFBR14_DELMIGDS={LEGACY|NORECALL}]
               [,TAPELIB_PREF={EQUAL|BYDEVICES}]
               [,REMIND_INTV=intv] }
```

z/OS R11 Install-Related Enhancements

z/OS UNIX: ALTROOT BPXPRMxx support

- **What:** In a shared file configuration, the *alternate sysplex root file system* is a hot standby for the sysplex root file system when it becomes unowned.
  - Complements z/OS R10 function to dynamically replace the sysplex root file system with a command (`F OMVS,NEWROOT=`*`new.root.file.system.name`*`,COND=<Yes|No>`)
- **How to use:**
  - Set up your alternate sysplex root (alloc, mount, copy, unmount)
  - Prepare your BPXPRMxx (and use `setomvs syntaxcheck`):
    - `ALTROOT FILESYSTEM ('`*`file.name`*`') MOUNTPOINT('`*`/alt_mountpt`*`') [PARM(' ')]`
  - Process the alternate sysplex root: `SET OMVS=(xx).`
    - Remove this support with BPXPRMxx: `ALTROOT NONE`
  - Use `D OMVS,OPTIONS` to see the alternate root you set up
- **Considerations:**
  - Several requirements and a restriction:
    - All systems must be at R11 or higher
    - Must be a shared file system configuration (single system sysplex is fine)
    - Alternate sysplex root must have same mountpoints (validated) and symbolic links (not-validated) as the current sysplex root.
    - …and some other requirements
    - ALTROOT statement is ignored during processing of the F BPXOINIT,FILESYS=REINIT
    - Restriction: any changes to sysplex root must also be done to alternate sysplex root.

z/OS R11 Install-Related Enhancements

z/OS UNIX: ALTROOT BPXPRMxx support

- **Usage Example:**
  - Assuming that OMVS.SYSPLEX.ALTROOT.ZFS was set up as the alternate root. BPXPRMWW contains:

```
ALTROOT FILESYSTEM('OMVS.SYSPLEX.ALTROOT.ZFS')
   MOUNTPOINT('/altroot')
```

  - **`D OMVS,OPTIONS`**

```
BPXO043I 13.56.01 DISPLAY OMVS 685
OMVS     0010 ACTIVE              OMVS=(WW)
CURRENT UNIX CONFIGURATION SETTINGS:
MAXPROCSYS      =       1000    MAXPROCUSER     =        500
…
ALTROOT         = OMVS.SYSPLEX.ALTROOT.ZFS
MAXUSERMOUNTSYS =          0    MAXUSERMOUNTUSER=          0
```

  - **`D OMVS,F`**

```
ZFS           59 ACTIVE                    READ  12/28/2012
  NAME=OMVS.SYSPLEX.ALTROOT.ZFS                     10.21.07
  PATH=/altroot
  OWNER=SYSA     AUTOMOVE=Y CLIENT=N
```

Automatically replacing the sysplex root file system
In a sysplex shared file configuration, the *alternate sysplex root file system* is a hot standby for the sysplex root file system that is used to replace the current sysplex root file system when the sysplex root file system becomes unowned. The alternate sysplex root file system is established by using the ALTROOT statement in the BPXPRMxx parmlib member during OMVS initialization or by using the SET OMVS command.
Requirements:

- A shared file system configuration is required. However, the sysplex can be a single system.
- All systems in the shared file system environment must be at z/OS V1R11 at the minimum.
- The alternate sysplex root must have the same mount points and symbolic links as the current sysplex root. The mount points are validated during processing, but the symbolic links are not. If mount points are missing, the current sysplex root is not replaced by the alternate sysplex root.
- The file system type for the alternate sysplex root and the current sysplex root must be either HFS or ZFS. They do not have to be identical.
- The alternate sysplex root PFS must be active on all systems in the shared file system configuration.
- If the SECLABEL class is active and the MLFSOBJ option is active, then the multilevel security label for the alternate sysplex root must be identical to the assumed multilevel security label of the current sysplex root.
- The sysplex root or any directories in the sysplex root file system must not be exported by the DFS or SMB server.
- The real path name for the mount points in the current sysplex root must not exceed 64 characters in length.

Restriction: The ALTROOT statement is ignored during processing of the F BPXOINIT,FILESYS=REINIT system command. You will have to manually issue SET OMVS=(*xx*) where BXPRMxx is the parmlib member containing the original ALTROOT statement.

Steps for setting up the alternate sysplex root for the dynamic replacement of the current sysplex root
Before you begin: You need to ensure that the alternate sysplex root does not reside in the same volume, device, and control unit as the current sysplex root.
Guideline: To minimize the single point of failure, the alternate sysplex root file system should be a different PFS type than that of the current sysplex root file system. Perform the following steps to establish an alternate sysplex root in a shared file system environment.

1. Allocate a new file system to be used as the alternate sysplex root file system.
    a. The UID, GID and the permission bits of the root directory in the alternate sysplex root file system must match the root directory in the current sysplex root file system
    b. If the SECLABEL class is active and the MLFSOBJ option is active, then the multilevel security label for the alternate sysplex root must be identical to the assumed multilevel security label of the current sysplex root.
2. On the alternate sysplex root, set up the mount points and the symbolic links. The mount points and the symbolic links must be same as the ones on the current sysplex root.
    a. Mount the alternate sysplex root file system at a temporary mount point (for example, **/altroot**).
    b. Select one of the following recommended ways to set up mount points and symbolic links:
        - Use the **pax** shell command to populate the alternate sysplex root file, using the current sysplex root as a source. For example:
          cd /
          pax -wr -pe -XCM ./ /altroot
        - Use **copytree** to populate the alternate sysplex root, using the current sysplex root as a source.
          For example: copytree -as / /altroot
        - Manually issue **mkdir** and **ln -s** shell commands to create the mount point directories and symbolic links similar to the current sysplex root.
    c. Unmount the alternate sysplex root.
3. Specify ALTROOT in the BPXPRMxx parmlib member with the mount point in the root directory of the current sysplex root file system. **Restriction:** The ALTROOT mount point must not exceed 64 characters in length.
   Example:
   ALTROOT FILESYSTEM ('OMVS.ALTROOT.ZFS')
   MOUNTPOINT('/sysalt') PARM ('FSFULL')
   You can use the SETOMVS SYNTAXCHECK operator command to validate the ALTROOT syntax.
   For information about the PARMs available, see *z/OS Initialization and Tuning Reference.*

4. Make sure that all systems in the shared file system environment have direct access to the new file system and can locally mount it.
5. Process the ALTROOT statement by using the SET OMVS command or by initializing the OMVS with the updated BPXPRMxx parmlib member. **Example:** SET OMVS=(xx)

When you are done, you have established an alternate sysplex root in the shared file system configuration. The alternate sysplex root is mounted in read-only mode at the specified mount point and designated as AUTOMOVE. When the alternate sysplex root becomes the current sysplex root, it is mounted in read-only mode and designated as AUTOMOVE regardless of the current sysplex root settings.

Requirement: If you make changes to the current sysplex root after alternate sysplex root has been successfully established, you must make the same changes to the alternate sysplex root as well.

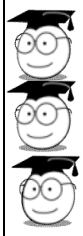
Steps for removing the alternate sysplex root support
Perform the following steps to remove the alternate sysplex root support.

1. In the BPXPRMxx parmlib member, replace the ALTROOT FILESYSTEM statement with the following statement: ALTROOT NONE
   Because the ALTROOT NONE and ALTROOT FILESYSTEM statements are mutually exclusive, only one can be specified in the BPXPRMxx parmlib member.
   If concatenating parmlib members result in multiple ALTROOT statements, then the first parmlib member specified on the OMVS= operator command that contains the ALTROOT statement will take effect.
2. Issue a SET OMVS operator command to process the ALTROOT NONE statement. **Example:**
   SET OMVS=(XX)

When you are done, you have removed the alternate sysplex root support and deleted any outstanding BPXF253E messages. The alternate sysplex root file system can be left mounted as a regular file system on all systems in the sysplex. If you need to reestablish the alternate sysplex root support with the same file system name, the file system will have to be unmounted globally before it can be used in the ALTROOT FILESYSTEM statement. Use your preferred unmount method to unmount the alternate sysplex root.

Older than dirt on potatoes…

Small Enhancements of System Programmer Interest



- **DFSMS: STGADMIN.DPDSRN.*oldname***
- **z/OS UNIX: Dynamic Service Activation**
- **BCP: IEARELEC to remove inactive EMCS consoles**

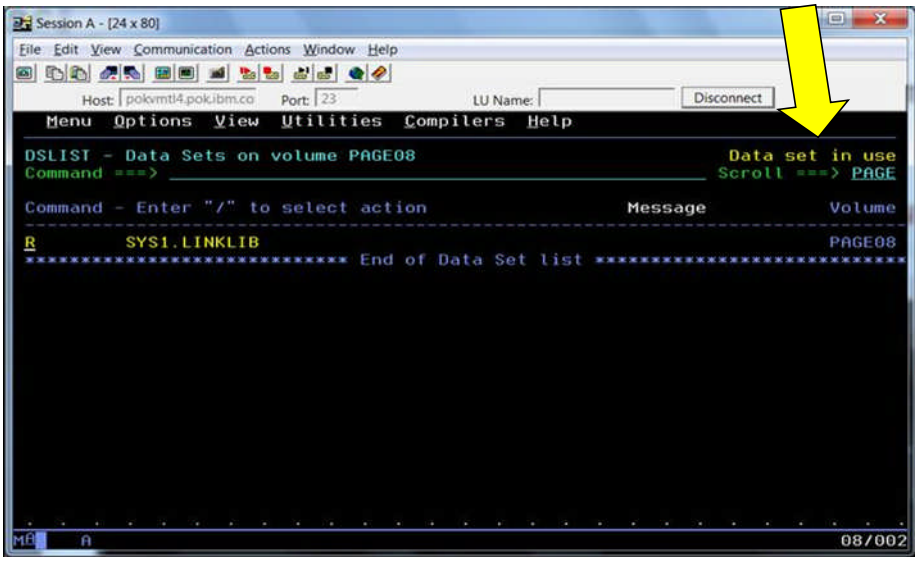
Older than dirt on potatoes:
DFSMS: STGADMIN.DPDSRN.*oldname*

- **What:** Provided by the operating system, a way to rename a non-SMS data set whose name is in use by another address space. (That is, to rename a duplicately named data set.) *This support carries inherent risks that must be understood and used wisely.*
  - Data sets that are currently in use on the driving system, while you are maintaining a target system, are good examples of those data sets that are duplicately named that you may want to rename.
  - System-Specific Aliases (SSAs) is the way that ServerPac gets around this problem during system replace installs.
  - There are probably many “home grown” tools to do this already, as the need has been around forever.
- **How to use:**
  1. Create `FACILITY` class profile `STGADMIN.DPDSNRN.`*`oldname`* for the data set you want to rename. Obtain READ access to this profile.
  2. Rename the data using ISPF PDF, after understanding consequences.
  - You could write your own program as well, using the correct CAMLIST macro expansion and authorization check.
  - SMF type 18 record written for successful duplicately-named renames: "So let it be written, so let it be done"
- **Considerations:** `STGADMIN.DPDSNRN.*` wildcarding of the profile name is strongly not recommended, as it could cover more data sets than intended. IDCAMS and IEHPROGM do not exploit `STGADMIN.DPDSNRN`.

Older than dirt on potatoes:
DFSMS: STGADMIN.DPDSRN.*oldname*

- **User Example:**

Before:

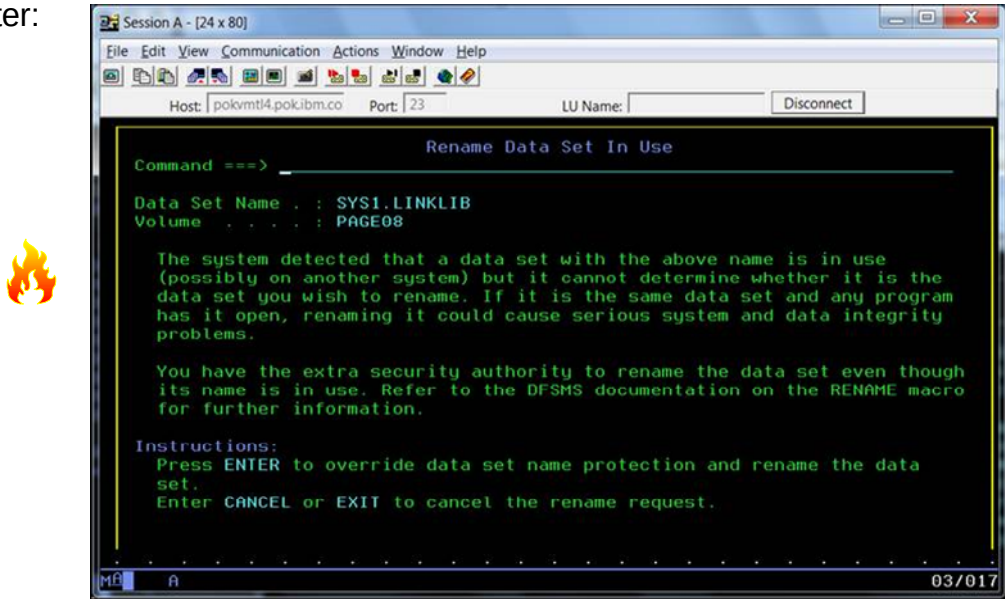


Older than dirt on potatoes:

DFSMS: STGADMIN.DPDSRN.*oldname*

- **User Example:**

After:



38 z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal! © 2013 IBM Corporation

STGADMIN.DPDSRN.*oldname*

In general, you cannot rename a data set whose name is the same as any data set that is allocated to another address space in the same system or in the scope of the SYSDSN enqueue. For this support, the system bypasses this restriction if all of the following are true:

- Your program sets on a certain bit in the CAMLST macro expansion. You can code this instruction: OI listname+2,X'10'.
- You have at least read authority to the RACF facility class named `STGADMIN.DPDSRN.`*`olddsname`*, where *olddsname* is up to 23 characters of the existing data set name. You can use a generic class name such as `STGADMIN.DPDSRN.SYS2*`. IBM recommends that no one have authority to `STGADMIN.DPDSRN.*` because it is too broad.
- The data set is not SMS-managed.

You can use the data set rename option of ISPF PDF. If you attempt to rename a non-SMS-managed, non-VSAM data set, the data set name is in use and you have the appropriate RACF facility class authority, then ISPF PDF asks whether you wish to proceed because you know that the data set is not actually open. Let the rename proceed only if you know the data set being renamed is not open on any system.

Attention: This option should be used with extreme caution. Very few people should have RACF authority to `STGADMIN.DPDSRN.`*`olddsname`*. Do not use this option unless you know the data set is not open on any system. After the data set is renamed, someone could delete it in a different address space. If someone has it open by the old name, new data sets will appear at those places on the disk. This would be a security violation that the system does not detect.

The data set rename function writes a type 18 SMF record to provide information to storage administrators, system programmers, and auditors. The record contains an indicator of whether it was successful due to the use of this duplicate name override function. If you request the option in the CAMLST macro expansion but the data set name is not in use, then the SMF indicator will not be on.

Older than dirt on potatoes:

z/OS UNIX: Dynamic Service Activation

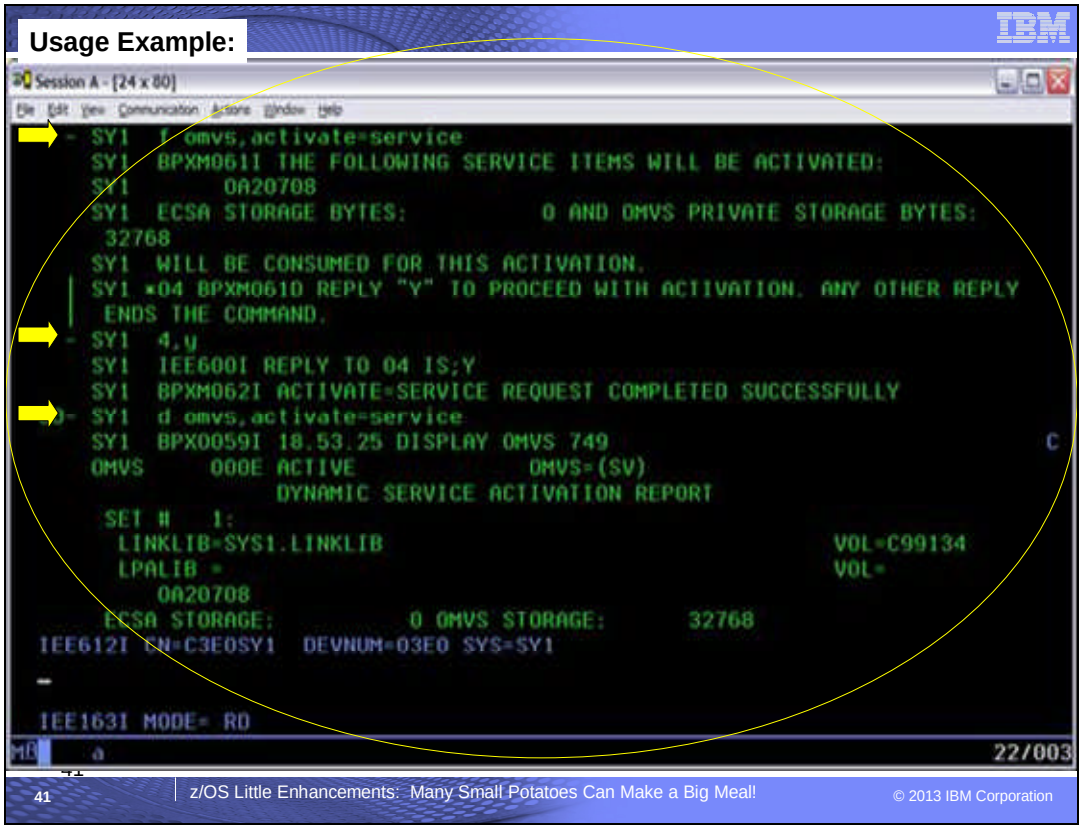
- **What:** Install corrective z/OS UNIX service without an IPL in some cases. Previously, an IPL was always required to refresh the kernel after installing maintenance.
  - Only those PTFs with `++HOLD REASON(DYNACT)` data will be capable of dynamic activation.
  - You must follow whatever explicit instructions are given within the ++HOLD documentation to perform the activation and deactivation.
- **How to use:** Set up `SERV_LPALIB` and `SERV_LINKLIB` statements in BPXPRMxx (or with `SET OMVS=xx` or `SETOMVS` commands). This will identify the load libraries for z/OS UNIX LPA and LINKLIB resident modules where you've installed SMP/E maintenance to be activated dynamically. Then issue commands to activate or deactivate. Example:
  `SERV_LPALIB('SYS1.LPALIB','C99134')` and
  `SERV_LINKLIB('SYS1.LINKLIB','C99134')` in BPXPRMxx, then
  - `F OMVS,ACTIVATE=SERVICE` activates the service.
  - `F OMVS,DEACTIVATE=SERVICE` backs off the service.
  - `D OMVS,ACTIVATE` displays the current set of services that were dynamically activated
- **Considerations:**
  - The `SERV_LPALIB` and `SERV_LINKLIB` data sets must be APF authorized on the system where the dynamic activation is to occur.
  - You need to stay fairly current on service to best be able to exploit this feature.
  - Not intended to be used to as a complete replacement for regular preventative maintenance application.

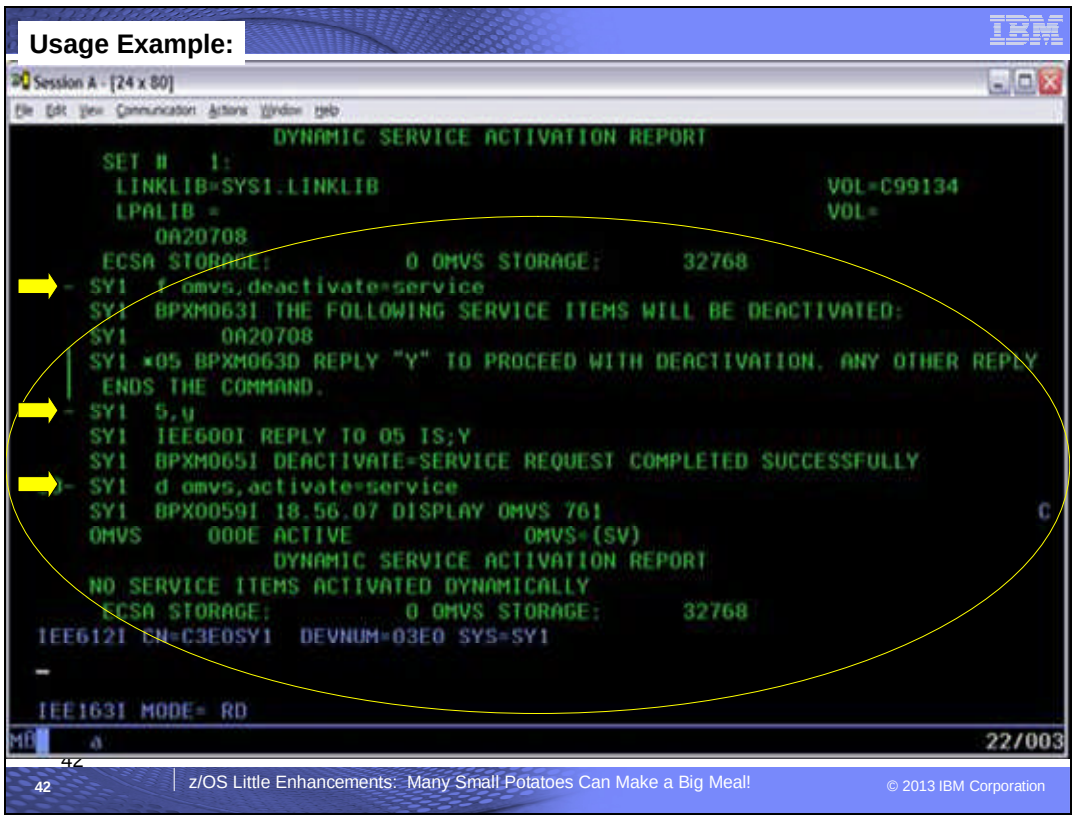
Usage Example:

```
Session A - [24 x 80]
File  Edit  View  Communication  Actions  Window  Help
       USED LOADMS IN SYS1.PARMLIB ON 107F
       ARCHLVL = 2    MTLSHARE = N
       IEASYM LIST = (MS,L)
       IEASYS LIST = (MS) (OP)
       IODF DEVICE 02FF
       IPL DEVICE 107F VOLUME C99135
    - SY1  d omvs,activate=service
       SY1  BPXO059I 18.42.55 DISPLAY OMVS 697                          C
       OMVS     000E ACTIVE                  OMVS=(MS)
                     DYNAMIC SERVICE ACTIVATION REPORT
       NO SERVICE ITEMS ACTIVATED DYNAMICALLY

    - SY1  set omvs=sv
       SY1  BPXO032I THE SET OMVS COMMAND WAS SUCCESSFUL.
    - SY1  d omvs,activate=service
       SY1  BPXO059I 18.44.47 DISPLAY OMVS 710                          C
       OMVS     000E ACTIVE                  OMVS=(SV)
                     DYNAMIC SERVICE ACTIVATION REPORT
       NO SERVICE ITEMS ACTIVATED DYNAMICALLY

  IEE612I CN=C3E0SY1  DEVNUM=03E0 SYS=SY1
  -
  IEE163I MODE= RD
MA    a                                                        22/003
```





z/OS UNIX: Dynamic Service Activation

As of z/OS R7, you can dynamically activate and deactivate service items (++PTFs, ++APARs, ++USERMODs) that affect the z/OS UNIX System Service component modules without having to re-IPL. This capability is primarily intended to allow an installation to activate corrective service to avoid unplanned re-IPLs of your systems. Additionally, this capability can be used to activate a temporary patch that can be used in gathering additional documentation for a recurring system problem. (Although this capability could conceivably be used to activate preventive service on an ongoing basis, it is not intended for this purpose as a replacement for the regular application of service that does require a re-IPL.)

Those PTFs that can be dynamically activated will have ++HOLD REASON(DYNACT) information within the PTF indicating whether the PTF can be activated as such. Additionally, any ++USERMOD or ++APAR provided from IBM will have explicit instructions provided by the IBM Service indicating whether the ++USERMOD or ++APAR can be dynamically activated, as well. Although a service item may be identified as being capable of dynamic activation, the level of a given system may not be current enough to allow the activation of the service item.

- `F OMVS,ACTIVATE=SERVICE` activates the service.
- `F OMVS,DEACTIVATE=SERVICE` backs off the service.
- `D OMVS,ACTIVATE` displays the current set of services that were dynamically activated.

Guideline: In order to be prepared to exploit dynamic service activation, you must stay current on z/OS UNIX component maintenance. Staying current makes it more likely that any given service item can be activated dynamically, because the running system will be at a high enough level to accept the service item. On a periodic or as-needed basis, you will have to determine the selected PTFs that you would be interested in activating dynamically for corrective purposes. These would likely be the PTFs that are of highest severity and highest impact related to your workloads. Although the dynamic service activation feature can be used to activate most UNIX System Services component PTFs, it is not intended to be used as a way to activate a large set of maintenance for preventive purposes.

Service items are activated from service activation libraries that have been identified via the `SERV_LPALIB` and `SERV_LINKLIB` parameters in the BPXPRMxx parmlib member. The service activation libraries contain the service items that have already been installed and that you want to activate on the next `F OMVS,ACTIVATE=SERVICE` command. These libraries must be APF authorized on the system that you are executing the activation.

In BPXPRMxx, details on the statements are:

- **`SERV_LPALIB('dsname','volser')`** Specifies the target service library where the z/OS UNIX System Services modules that are normally built into LPA are located.
  - **Value Range:** *dsname* is a 1-to-44 character value representing a valid MVS load library data set name. The alphabetic characters in the load library name must be uppercase. *volser* is a 1-to-6 character value representing a valid volume serial number for the volume that contains the specified MVS load library. The alphabetic characters in the volume serial number must be uppercase.
  - You can change the value of `SERV_LPALIB` dynamically using the `SETOMVS` or `SET OMVS` command. To make a permanent change, edit the BPXPRMxx member that will be used for future IPLs.
- **`SERV_LINKLIB('dsname','volser')`** Specifies the target service library where the z/OS UNIX System Services modules that are normally loaded from SYS1.LINKLIB into the private area of the OMVS address space are located.
  - **Value Range:** *dsname* is a 1-to-44 character value representing a valid MVS load library data set name. The alphabetic characters in the load library name must be uppercase. *volser* is a 1-to-6 character value representing a valid volume serial number for the volume that contains the specified MVS load library. The alphabetic characters in the volume serial number must be uppercase.
  - You can change the value of `SERV_LINKLIB` dynamically using the `SETOMVS` or `SET OMVS` command. To make a permanent change, edit the BPXPRMxx member that will be used for future IPLs.

For more information about using this enhancement, see *z/OS UNIX System Services: Planning.*

Older than dirt on potatoes:

BCP: IEARELEC to remove inactive EMCS consoles

- **What:** You may have a longer IPL, because of console data refresh times in a sysplex. Information on all EMCS consoles is sent across systems, including inactive EMCS consoles that are no longer needed.
- You have the ability to remove inactive EMCS consoles that are no longer needed. This will allow reduced system refresh and IPL times into a sysplex.
  - Consoles Distributed mode may also reduce IPL times into a sysplex.
- **How to use:**
  - Assemble a sample program (IEARELEC), and APF authorize it.
  - Invoke the program with a batch job to remove all inactive EMCS consoles with PARM='CONSNAME(*filter*)'. Wildcarding is supported.
- **Considerations:** Only inactive EMCS consoles will be removed.
  - When specifying PARM='CONSNAME(*)' EMCS consoles that are inactive and are not IBM internal consoles are removed.

Older than dirt on potatoes:

BCP: IEARELEC to remove inactive EMCS consoles

- **How to use:** Once IEARELEC is assembled, link edited, and APF authorized.

```
//JOBALL JOB …
//S1 EXEC PGM=IEARELEC,PARM='CONSNAME(*)'
//STEPLIB DD DSN=MWALLE.APF.LINKLIB,DISP=SHR
   CNZ4002I EMCS CONSOLE REMOVAL FOR WILDCARD PATTERN *
   FOUND: 8  REMOVED: 0  NOT REMOVED: 8
   MRC105I ONE OR MORE EMCS CONSOLES MATCHING THE WILDCARD
    PATTERN OF *         WERE NOT REMOVED
```

- `//S2 EXEC PGM=IEARELEC,PARM='CONSNAME(CONSOL*)'`
  - Will attempt to remove any inactive EMCS console with a name that begins with 'CONSOL' (for example, CONSOL01,CONSOL02, etc.)
- `//S3 EXEC PGM=IEARELEC,PARM='CONSNAME(SY?CON*)'`
  - Will attempt to remove any inactive EMCS console whose name has as its first two characters, 'SY', and its fourth thru sixth characters, 'CON' (for example, SY1CON1, SY1CON2, SY2CON1, SY2CON2, etc.)

BCP: IEARELEC Sample for Removing EMCS Consoles

z/OS R7 delivered the support for deleting unused EMCS consoles. You can delete the definition of any inactive extended MCS console, thus freeing the ID that had been assigned to the extended MCS console. The system then can reuse that ID for a newly-defined extended MCS console. To remove a console definition, use the sample JCL for program IEARELEC in SYS1.SAMPLIB. The following restrictions for removing an extended MCS console apply:

- The extended MCS console must be inactive.
- Extended MCS consoles can only be removed on a z/OS V1R7 or higher system. The removal will be communicated to systems at a lower level.
- The console ID of a removed extended MCS console can be reused once it has been deactivated and removed. It is safe to use the console ID to process a command response, but you should avoid saving the console ID for later processing. Therefore, you should use the console name to direct messages to specific consoles. If the console ID is used, messages may end up going to unintended consoles.
- The console ID of a removed extended MCS console can only be reused by activating another extended MCS console on a z/OS V1R7 or higher system.

Sample invocation of IEARELEC:

```
//JOBA JOB ...
//RM1 EXEC PGM=IEARELEC,PARM='CONSNAME(CONSOL01)'
```

JOBA will attempt to remove an inactive EMCS console named ‘CONSOL01’. If this inactive console is not found, you will see:

```
CNZ4001I CONSOLE CONSOL01 WAS NOT REMOVED. EMCS CONSOLE IS NOT DEFINED
MRC103I EMCS CONSOLE CONSOL01 WAS NOT REMOVED. RETURN CODE 0008, REASON CODE 0818
```

Another sample with wildcarding, with the sample output:

```
//JOBB JOB ...
//RM2 EXEC PGM=IEARELEC,PARM='CONSNAME(SY?CON*)'
CNZ4002I EMCS CONSOLE REMOVAL FOR WILDCARD PATTERN SY?CON*
FOUND: 4 REMOVED: 4 NOT REMOVED: 0
THE FOLLOWING EMCS CONSOLES WERE REMOVED:
SY1CON1 SY1CON2 SY2CON1 SY2CON2
```

When you specify CONSNAME(*) for IEARELEC, all EMCS consoles that are inactive and are not IBM internal consoles are removed.

Summary of What We Might Want to SHARE with Our User Community:

System Programmer & User Items:

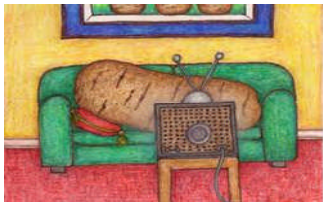
- **SDSF:** Cursor-sensitive Sort
- **DFSMS:** IDCAMS DELETE All Members
- **TSO/E:** LOGONHERE reconnect support

System Programmers Items:

- **z/OS UNIX:** Prevent Content Overlay during MOUNT
- **DFSMS:** IEBPDSE Batch Program
- **BCP:** IEFSSNxx BEGINPARALLEL
- **BCP:** Timed Event Data Report
- **BCP:** Some PROGxx Goodies
- **BCP:** D ALLOC and SETALLOC commands
- **z/OS UNIX:** ALTROOT BPXPRMxx support
- **DFSMS:** STGADMIN.DPDSRN.*oldname*
- **z/OS UNIX:** Dynamic Service Activation
- **BCP:** IEARELEC to remove inactive EMCS consoles

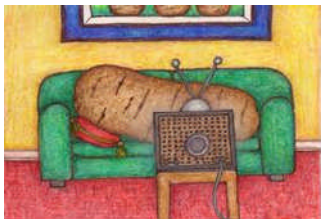
z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal ! Summary

- **z/OS V1.13:**
  - **z/OS UNIX: Prevent Content Overlay during MOUNT**
    - Good to protect from overmounts.
  - **DFSMS: IEBPDSE Batch Program**
    - Helpful to see if your PDSEs are structurally sound.
  - **SDSF: Cursor-sensitive Sort**
    - Easy to use, and can be helpful when looking for something.
- **z/OS V1.12:**
  - **BCP: IEFSSNxx BEGINPARALLEL**
    - A time saver that is easy to implement.
  - **BCP: Timed Event Data Report**
    - Data proof of where time is spent.
  - **BCP: Some PROGxx Goodies**
    - Nice defaults to set up.
  - **DFSMS: IDCAMS DELETE All Members**
    - Something long desired!



z/OS Little Enhancements: Many Small Potatoes Can Make a Big Meal ! Summary

- **z/OS V1.11:**
  - **TSO/E: LOGONHERE reconnect support**
    - It really works.
  - **BCP: D ALLOC and SETALLOC commands**
    - An availability aid.
  - **z/OS UNIX: ALTROOT BPXPRMxx support**
    - Hot standby makes sysplex root no longer a single point of failure.



- **Older than dirt on potatoes:**
  - **DFSMS: STGADMIN.DPDSRN.*oldname***
    - Nice to have this option, but beware of using it.
  - **z/OS UNIX: Dynamic Service Activation**
    - Corrective service availability aid.
  - **BCP: IEARELEC to remove inactive EMCS consoles**
    - Can assist with faster IPLs.

