z/OS Unix System Services V2R1 Latest Status and New Features

Vivian W Morabito

March 11, 2014 1:30PM - 2:30PM

Platinum Ballroom Salon 5

morabito@us.ibm.com

Table of Contents

Automount Enhancements

- Automount -f query to display last use information.
- Mapping File Updates:
  - new key “pathperm( )” in allocany and allocuser to allow the user to control the permissions.
  - new key “euid” in allocany and allocuser - Use effective UID/GID
  - new “charcase” keyword – enhances automount to have “name” in the generic entry to be case sensitive.
- Master File Update:
  - Automount policy creation supports symbolics
- Change automount with -a (append) to be a serialized operation.

Automount Enhancements, automount -f

automount –f FileSysName

Displays the information of the job that last accessed the specified file system.

The file system name must be specified and it is treated as case-insensitive.

The information includes:

file system name, mount point, state, timer, UID, PID, and job name.

The state has two values: duration and delay.

The timer is the minutes left for the specified file system to be in this state

automount -f example

```
#  automount -f OMVSSPA.T015002.ZFSTPNS
OMVSSPA.T015002.ZFSTPNS
/u2/T015002
STATE: Duration    TIMER: 22
EUID:15002   PID:84082835   JOBNAME:T015002

#  touch /u2/T015002/fil

#  automount -f OMVSSPA.T015002.ZFSTPNS
OMVSSPA.T015002.ZFSTPNS
/u2/T015002
STATE: Duration    TIMER: 21
EUID:0   PID:84082860   JOBNAME:TOTTEN2
```

Note, see change in last user update, duration does not reset.

automount -f example

```
#  /usr/sbin/automount -f OMVSSPA.T015002.ZFSTPNS
OMVSSPA.T015002.ZFSTPNS
/u2/T015002
STATE: Delay    TIMER: 22
EUID:15002   PID:393683   JOBNAME:T015002


#  touch /u2/T015002/filea


#  /usr/sbin/automount -f OMVSSPA.T015002.ZFSTPNS
OMVSSPA.T015002.ZFSTPNS
/u2/T015002
STATE: Delay    TIMER: 30
EUID:0   PID:67306686   JOBNAME:TOTTEN
```

Note, see change in last user update and delay.

new key “pathperm( )” in allocany & allocuser

- When a new file system is allocated, the permission is set to the value of pathperm (the default is 750). If permission is not specified, or if the value is 000, the default is used.

- To display the pathperm value, whether or not it is specified for allocany and allocuser, use the automount -q option.

Note: File system type zFS is only supported type for this new support.

Use effective UID/GID when euid keyword is specified

- **When a zFS is created and allocated to a new user, the owner UID and GID are based on that user.**

- **By default, automount uses the UID and GID of the user ID that owns the process.**

- **If the euid keyword is specified for allocany or allocuser, the thread-level UID and GID are used instead.**

Pathperm & euid example-the policy file

From automount –q:

```
/home
name                    *
filesystem              OMVSSPA.SVT.USER.<uc_name>.ZFS
type                    ZFS
allocany                space(200,50) storclas(USSFS) pathperm(700) euid
mode                    rdwr
duration                nolimit
delay                   10
```

Pathperm & euid example

```
# /u/totten/seuid

current uid= 0
after seteuid, current uid= 0
after seteuid, current euid= 25
about to issue creat() in wellie's home..

returned success, exiting


MEGA@J78:/>df /home/wellie
Mounted on        Filesystem                          Avail/Total    Files      Status
/home/wellie      (OMVSSPA.SVT.USER.WELLIE.ZFS) 284926/288000 4294967291
Available


# ls -nd /home/wellie
drwx------    2 25       512          8192 Jul 16 10:57 /home/wellie
```

support symbolics in Master File

MVS system static symbols can be used in the master files, such as &SYSNAME.

- Use only static system symbols to avoid unexpected results.
- Automount cannot dynamically change the symbols. Symbols are resolved at the time when loading automount policy. If the symbol is dynamically changed after the policy is loaded, the policy must be reloaded to have the symbol resolved again.
- Query option (-q) of automount facility displays the substitutions of the symbols.

Advantage: Can use one Master file in different systems in your shared environment.

System symbolics in the master file

Given: D SYMBOLS shows: &SYSNAME. = "J40"

mkdir /J40/temphome

Update /etc/auto.master with:

/&SYSNAME./temphome /etc/autohome

Restart automount,

see output of D OMVS,F:

AUTOMNT 374 ACTIVE RDWR 07/17/2013 L=78
NAME=*AMD/J40/temphome 10.02.07 Q=78
PATH=/J40/temphome

OWNER=J40 AUTOMOVE=U CLIENT=N

new charcase keyword

Enhance automount to provide the capability to have “name” in the generic entry to be case sensitive.

`charcase [lower|upper|asis]`

Indicates the case for names that can match the * specification. This keyword is valid on any specification but is only meaningful on the generic entry.

`lower` Only names that are composed of lowercase characters can match the * specification. Numbers and special characters can also be used. When this keyword is specified, uppercase characters are not allowed. This is equivalent to lowercase yes.

`upper` Only names that are composed of uppercase characters can match the * specification. Numbers and special characters can also be used. When this keyword is specified, lowercase characters are not allowed.

`asis` Any name can match the * specification. This is the default and is equivalent to lowercase no.

Note: Existing LOWERCASE and CHARCASE are mutually exclusive

new charcase keyword example

With /J40/temphome automount managed as follows:

```
name          *
filesystem    USER.<uc_name>.TFS
type          TFS
mode          rdwr
charcase      upper
duration      nolimit
delay         10
```

 # touch /J40/temphome/TMPDIRone/file4 touch: file "/J40/temphome/TMPDIRone/file4": EDC5129I No such file or directory.(errno2=0x0594003D)

 # touch /J40/temphome/TMPDIRONE/file4

 # df -Pk /J40/temphome/TMPDIRONE/

Filesystem	1024-blocks	Used	Available	Capacity	Mounted on
USER.TMPDIRONE.TFS	1024	60	964	6%	/J40/temphome/TMPDIRONE

seralize append across the sharing group

- automount with -a (append) was enhanced to be a serialized operation.
- This is beneficial in the case where append is used on initial load of the policy from various systems that try to only load the policy that relates to that system.
- Now initialization of multiple systems is serialized.

TFS Enhancements

- FSFULL Monitoring:
  - With eventual action message when it is 100% full regardless of the FSFULL setting. This is consistent with HFS & zFS messages as space becomes critical.
  - Modify command to change the default FSFULL value for subsequent TFS mounts.
- Increase the size of a TFS file system
  - Manually
  - Automatically
- Produce unique reason codes with bpxmtext support

FSFULL

-fsfull(threshold,increment)

Sets FSFULL monitoring to the specified values. When specified, this option overrides the default setting.

threshold is a number in the range 1-99. When the temporary file system is *threshold* percent full, a message is issued.

increment is a number in the range 1-99. Once the temporary file system is threshold full, TFS uses *increment* to update or delete a threshold message that was issued when the file system fills or empties.

Example:

-FSFULL(85,5)

You would see message BPXTF009E when the file system is 85 percent full. Then it would issue another message when the file system is 90, 95 and 99 percent full.

-ea and -em

-ea count

Allows the TFS file system to automatically grow *count* times.

-em count

Allows the TFS file system to manually grow *count* times.

TFS will grow 1K blocks each time it grows. For the default 4K block size, this is 4MB growth in size (1K * 4K = 4MB).

The extension occurs when the file system fills up, prior to the file system full message.

Restriction: The sum of auto-extend and manual extend cannot exceed 500.

FSFULL support in BPXPRMxx member

FILESYSTYPE TYPE(TFS2) ENTRYPOINT(BPXTFS) ASNAME(TFSPROC2,'SUB=MSTR')
PARM('-fsfull(85,5) -ea 50 -em 10')
FILESYSTYPE TYPE(TFS1) ENTRYPOINT(BPXTFS) ASNAME(TFSPROC1,'SUB=MSTR')
PARM('-nofsfull')

```
D OMVS,P      (last 2 sections of output):
 PFS TYPE      FILESYSTYPE PARAMETER INFORMATION
  TFS2          -fsfull(85,5) -ea 50 -em 10
  TFS1          -nofsfull
 PFS TYPE   STATUS INFORMATION
  TFS2      GLOBAL SETTINGS: fsfull(85,5) ea 50 em 10
  TFS1      GLOBAL SETTINGS: fsfull(99,5) ea 0 em 0
```

Tip: BE SURE TO USE THE DASH ON THE PARM

FSFULL, ea & em on mount statement

You may choose to override the new parameters setting for specific file systems:

```
J40 /u/totten# mkdir tfstmp2
J40 /u/totten# mount -f TFS_2 -t tfs2 -o'-fsfull(10,5) -ea 10 -em 0'
  tfstmp2
J40 /u/totten# df -Pkv tfstmp2
Filesystem          1024-blocks        Used  Available  Capacity Mounted on
TFS_2                     1024          88        936     9% /u/totten/tfstmp2
TFS2, Read/Write, Device:392, ACLS=Y
-fsfull(10,5) -ea 10 -em 0
File System Owner : J40          Automove=Y      Client=N
Filetag : T=off   codeset=0
```

Tip: BE SURE TO USE THE DASH ON THE PARM

TFS query

Use MODIFY to query the file system:

```
F TFSPROC2,Q
BPXTF012I GLOBAL SETTINGS: fsfull(85,5) ea 50 em 10
BPXTF016I TFS_2
BPXTF016I    31 or 64 bit:      64  Free blocks:        234
BPXTF016I    Block size:      4096  Total blocks:       256
BPXTF016I    Cache hit:      30731  Cache miss:       15380
BPXTF016I    Cast out:       15107  Copy out:            17
BPXTF016I    fsfull threshold:  10  fsfull increment:     5
BPXTF016I    auto-extend:       10  manual extend:       5
```

FSFULL monitoring

As the file system hits the specified utilization percentage and increments beyond, will see the following messages:

```
BPXTF009E FILESYSTEM EXCEEDS 10% FULL: TFS_2
BPXTF009E FILESYSTEM EXCEEDS 15% FULL: TFS_2
BPXTF009E FILESYSTEM EXCEEDS 19% FULL: TFS_2
BPXTF009E FILESYSTEM EXCEEDS 24% FULL: TFS_2
BPXTF009E FILESYSTEM EXCEEDS 29% FULL: TFS_2
```

Note that there is some rounding on the reporting.

Messages are DOMed as appropriate (extension, file removal, etc).

TFS grow

Use the MODIFY to manually grow a file system.

```
F tfs,grow filesysname
```

where filesysname is the name of the file system to be extended. Each extension is 1K blocks.

```
F TFSPROC2,GROW TFS_2

BPXTF014I FILE SYSTEM EXTENDED ea 10 em 4 TFS_2

F TFSPROC2,Q

BPXTF012I GLOBAL SETTINGS: fsfull(85,5) ea 50 em 10

BPXTF016I TFS_2

BPXTF016I    31 or 64 bit:      64  Free blocks:       1259

BPXTF016I    Block size:      4096  Total blocks:      1280

BPXTF016I    Cache hit:      30735  Cache miss:       15381

BPXTF016I    Cast out:       15108  Copy out:            17

BPXTF016I    fsfull threshold:  10  fsfull increment:     5

BPXTF016I    auto-extend:       10  manual extend:       4
```

Changing defaults for subsequent mounts

Use the MODIFY to change global defaults.

Syntax:

```
F tfs,FSFULL(threshold,increment)
```

The newly specified threshold and increment will be used on subsequent mounts.

Syntax:

```
F tfs,EA number
F tfs,EM number
```

where number is the number of automatic or manual extends allowed for a file system that is subsequently mounted without using the -ea or -em parameters.

Changing defaults example

```
D OMVS,P
PFS TYPE   STATUS INFORMATION
  .....
 TFS2      GLOBAL SETTINGS: fsfull(85,5) ea 50 em 10

F TFSPROC2,EM 300
BPXTF012I GLOBAL SETTINGS: fsfull(85,5) ea 50 em 300

D OMVS,P
PFS TYPE   STATUS INFORMATION
  .....
 TFS2      GLOBAL SETTINGS: fsfull(85,5) ea 50 em 300
```

PIPE Enhancements

- MAXPIPEUSER
- PIPE UID DISPLAY command and zlsof
- PIPE/FIFO system/process limit monitoring and limit expansion

MAXPIPEUSER

New MAXPIPEUSER BPXPRMxx statement was implemented that provides a configurable "per UID" limit on the number of named and unnamed pipes that an application can open and use concurrently.

The MAXPIPEUSER limit can be dynamically altered using the SETOMVS MAXPIPEUSER= or SET OMVS RESET= system commands.

The D OMVS,OPTIONS system command was changed to list the current MAXPIPEUSER value.

The D OMVS,PIPES command was changed to include listing the current MAXPIPEUSER value.

MAXPIPEUSER

```
SETOMVS MAXPIPEUSER=50
BPXO006I ERROR IN SETOMVS COMMAND. THE MAXPIPEUSER
PARAMETER VALUE IS OUT OF THE ALLOWED RANGE OF 256 TO 8730.
BPXO012I ERRORS OCCURRED IN THE PROCESSING OF THE
SETOMVS COMMAND; NO VALUES WERE SET.
```

```
SETOMVS MAXPIPEUSER=1024
BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
```

Note: The limit of 8730 is always applied to UID=0 user.

MAXPIPEUSER, D OMVS,PIPES displays

```
D OMVS,OPTIONS
```
(last line shows current MAXPIPEUSER value)

```
MAXPIPEUSER      =        1024     PWT                = ENV
```

New Display Pipes command shows the top two high-use UIDs:

```
D OMVS,PIPES

BPXO073I 13.00.41 DISPLAY OMVS 975

OMVS      0011 ACTIVE                OMVS=(ST,R1,MM,ZM)

PIPE OWNER SUMMARY          MAXPIPEUSER=1024

                           CURRENT    HIGHWATER

USERID              UID     USAGE       USAGE

*MULTNAM              0        36          806

HOMER               272         8           61

HIGHWATER USER:

  USERID=T016563 UID=16563 HIGHWATER USAGE=1000
```

Note: Use the D OMVS,PIPES,RESET to reset the user HIGHWATER USAGE and the HIGHWATER USER information and display the two UIDs with the highest pipe create count.

D OMVS,PIPES,ALL - Displays all the open pipe creators

```
D OMVS,PIPES,ALL

BPXO073I 13.01.41 DISPLAY OMVS 926

OMVS      0011 ACTIVE                OMVS=(ST,R1,MM,ZM)

PIPE OWNER SUMMARY          MAXPIPEUSER=1024

                           CURRENT    HIGHWATER

USERID                UID     USAGE       USAGE

*MULTNAM                0        36          806

APACHEWK             2575         4            5

ZFSUBS               2573         2            6

NFSUBS               2572         3            6

HOMER                 272         4           61

ISTWAS8              2676         1            2

ISTWAS7              2675         1            3

OMFCONF              1218         2            6

HIGHWATER USER:

  USERID=T016563 UID=16563 HIGHWATER USAGE=1000
```

D OMVS,PIPES display by UID

Display the processes that created active pipes for specific UID

```
D OMVS,PIPES,UID=272

 BPXO074I 13.15.06 DISPLAY OMVS 402

 OMVS      0011 ACTIVE                OMVS=(ST,R1,MM,ZM)

 TOTAL CURRENT USAGE=8

                    CURRENT     SYSTEM

       PID          USAGE       NAME

         69541            4     C00

     50401397             3     C00

     16842835             1     C00
```

Note: Only the SYSTEM NAME is provided to handle the situation where a named-pipe (FIFO) is opened from a client system. Named pipes are always located at the system that owns the file system containing the FIFO.

zlsof – fully supported TSO, shell and REXX command

```
zlsof [-p[pids]|[-a[asids]|[-j[jobs]] [-u[users]] [-c]
  [-d] [-t] [-i] [-l] [-n] [-su] [-v] [-m maxtime]
[-rw[seconds]] [pathname|pipe|socket]
```

Description

The zlsof utility displays information about open files, sockets, and pipes (including named pipes, which are also known as FIFO special files). The display includes the file name or inode number, associated PID, user, file system, and whether the file was locked by the byte range lock manager (BRLM).

zlsof – example, filter output by PID

```
J50 /u/totten# zlsof -p 50668939

zlsof version=130111
  Searching for all file usage by PID 50668939
  Command       PID User   File System              Mountpoint Inode/file

/bin/sh  50668939 TOTTEN OMVSSPA.SVT.SYSPLEX.ZFS /          r 1
                         USSZFS.TOTTEN.ZFS       /u/totten  c 1
                         OMVSSPA.SVT.S2.TMP.TFS  /J50/tmp   17011 /tmp/fl3e
                         OMVSSPA.SVT.U.ZFS       /u         11312 /u/.sh_histor
                         OMVSSPA.SVT.S2.TMP.TFS  /J50/tmp   22065 /tmp/flaccess
.J50.var3.0.sh2
                         OMVSSPA.SVT.S2.TMP.TFS  /J50/tmp   17010 /tmp/fl3

zlsof End of output
```

zlsof – example output for pipe usage

```
J50 /u/totten# zlsof -p  16851259,16842835  pipe
zlsof version=130111
Searching for use of PIPE by PID 16851259 16842835
Command              PID User  File System Mountpoint  Inode/file
/usr/sbin/sshd 16842835 HOMER             pipe: 2
/bin/ssh       16851259 HOMER             pipe: 4
zlsof End of output
```

Unicode Enhancements

The Unicode Standard provides a unique number for every character, regardless of platform, language, or program.

Using the Unicode Standard, you can develop a software product that works with various platforms, languages, and countries. The Unicode Standard also allows data to be transported through many different systems. Modern systems provide internationalization solutions based on the Unicode Standard.

The objective is to allow any z/OS Unix thread be set to any code page, z/OS Unix files be tagged with any code page, and auto conversion be enabled so that text conversion will occur when the files are read or written by that thread. It is not an objective to enable programs to be compiled or run as if they were natively built with any code page.

See: Unicode Services User's Guide and Reference Version 2 Release 1

Unicode Enhancements, continued

Files that are tagged can be converted between any coded character set identifier (CCSID) of the program or user and the CCSID of the file, if Unicode Services supports that conversion.

Unlike Enhanced ASCII, which affects conversion of regular file, pipes, and character special files, an environment enabled for Unicode Services environment affects regular files and pipes only.

No character special support beyond that provided for Enhanced ASCII is included.

See: Unicode Services User's Guide and Reference Version 2 Release 1

Three “to do’s” for UNICODE

1. Set up Unicode Services (enable conversion)

2. Assign the appropriate file tag for each file that is to be converted.

3. Assign a coded character set identifier (CCSID) to each program or thread. By default, the initial CCSID for every thread is IBM-1047 (EBCDIC).

Enabling automatic conversion for the z/OS UNIX environment.

- Use AUTOCVT(ALL) in BPXPRMxx to enable Unicode Services support for all programs and users. When AUTOCVT(ALL) is set, every read and write operation for a file is checked to see if conversion is necessary. *AUTOCVT(ON) still uses the unchanged Enhanced ASCII.*

- SET OMVS and SETOMVS AUTOCVT=ALL operator commands will now be supported to enable UNICODE conversion.

- Enable conversion (regardless of any environment variables or parmlib values) in a C program by using fcntl() with the F_CONTROL_CVT command and the SetCvtAll option. This also allows you to tag files and the program.

Unicode Enhancements, c/c++ applications

1. Use the FILETAG runtime option with the _BPXK_AUTOCVT environment variable set to ALL. (Set up Unicode Services.)

2. Issue the F_SETTAG subcommand of the BPX1FCT (fcntl) callable service from a program to permanently or temporarily tag the specified file.

or

Issue BPX1CHR (chattr) callable service from a program to permanently tag the specified file.

or

Mount the file system with the TAG parameter to temporarily tag the specified file.

3. Initialization of a new thread, fork() related functions, the setenv(), putenv(), clearenv() and __ae_autoconvert_state() have been enhanced. A new state, _CVTSTATE_ALL, was added.

Unicode Enhancements, z/OS UNIX shell or BPXBATCH

1. Use the _BPXK_AUTOCVT environment variable. (Set up Unicode Services.)
2. Issue the chtag command to permanently tag the specified file.

or

Mount the file system with the TAG parameter to temporarily tag the specified file.

3. Assign a coded character set identifier (CCSID) to each program or thread in the shell. Set environment variable _BPXK_PCCSID. This identifies the program CCSID for the running thread or user. It can be used to override the internal default of 1047 (EBCDIC).

Restriction: Only SBCS sh scripts are supported to tag and run when automatic conversion is enabled and the locale is SBCS. Non-SBCS sh scripts (e.g DBCS, MBCS) are not supported.

REXX Enhancements

- BPXWDYN
- BPXWUNIX
- Address Syscall
- Address TSO

BPXWDYN

- Enhancements
  - Extend BPXWDYN so it may be called from non-REXX environments for the info retrieval functions.
  - Add additional SVC99 keys that have been requested over the years

- Benefits

- BPXWDYN enables a simple interface for non-REXX applications to have access to the SVC99 info retrieval functions.
  - Additional keys enable BPXWDYN to be more broadly used.

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

BPXWDYN

▪ Almost all info retrieval keys are now supported, see publication for complete list.

– Build your own TU (non-info retrieval)
  - "alloc tu(000100010002c1c2)" same as "alloc fi(ab)"

  TU can be used multiple times in an allocation request.

– Debug aids:
  - DUMP key can be used to obtain a TDUMP before / after SVC99
  - DIAG key can be used to show:

    Version information — BPXWDYN input
    Key parsing information — Generated text units

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

BPXWDYN

- Calling BPXWDYN outside of REXX
  - BPXWDYN has always supported non-REXX calls with the restriction that information is not returned to the application for keys that return information
  - Multiple parameters can now be specified
    - On input the additional parameters specify the length of the parameter area and the text key
    - On output the parameters contain information returned by SVC99 for the corresponding key
  - See the z/OS Using REXX book for the parameter format

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

BPXWUNIX

- Problem Statement / Need Addressed
  - It is very difficult to run a login shell or background process through BPXWUNIX. Two shells would have to process the command line which complicates substitution strings.
  - BPXWUNIX line length is too restrictive

- Solution
  - Two additional arguments are added to BPXWUNIX to direct it to run a login shell or a background process.
  - Increase BPXWUNIX line length from 2K to 16K

- Benefit / Value
  - Simplifies starting an async process from REXX
  - Allows for long output lines

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

BPXWUNIX

- Arg 6 (login):
  0: run /bin/sh -c (A login shell is not used, this is the default)
  1: run /bin/sh -Lc (A login shell is used)

- Arg 7:
  0: run /bin/sh (this is the default)
  1: run /bin/nohup /bin/sh -c or -Lc
  The argument for /bin/sh (-c or -Lc) is determined by arg 6

  Note: you will not be able to capture output for a background process through BPXWUNIX. Consider redirecting stdout/stderr files.

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

additional REXX services

- Problem Statement / Need Addressed
  - Several sets of useful services are not available to REXX programs
- Solution
  - The following sets of services are now available
    - Message queues
    - Shared memory
    - Shared memory locks
    - File shares via the file server interfaces
- Benefit / Value
  - Extends the type of applications that can be implemented in REXX

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

address TSO

- Problem Statement / Need Addressed
  - Open file descriptors are not inherited by the Address TSO co-process.

- Solution
  - Add an environment variable that directs address TSO to pass open file descriptors. It's enabled by setting environment variable BPXWRFD=YES. Can be set outside of the rexx program or the rexx program can use the environment() function to set the variable.

- Benefit / Value
  - Allows for a TSO program run via address TSO to access file descriptors opened by the calling rexx program.

See: Using REXX and z/OS UNIX System Services, Version 2 Release 1

System symbols cache update

- For z/OS V2R1,

  `SETLOAD xx,IEASYM`

  console command will dynamically replace the running system symbol table with a new one.

- z/OS UNIX support was enhanced to exploit this new function:

  - **When a system symbol is used in a symlink and that symbol is dynamically changed, the next access to that object resolves correctly.**

  - **This is per sysplex image.**

DISPLAY OMVS,STORAGE

- New **`DISPLAY STORAGE`** or **`ST`**

- Displays storage usage information for the z/OS UNIX System Services kernel address space. The display includes a summary of system-wide kernel stack cell pool cell usage, a list of processes using 50 or more stack cells, and private below the bar storage usage.

See: z/OS MVS System Commands Version 2 Release 1

and

z/OS MVS System Messages Volume 3 (ASB – BPX) Version 2 Release 1

DISPLAY OMVS,STORAGE

```
D OMVS,STORAGE
BPXO075I 13.04.42 DISPLAY OMVS 703
OMVS      0011 ACTIVE                OMVS=(ST,R1,MM,ZM)
                        KERNEL STORAGE USAGE
PRIVATE STORAGE:
CURRENT USAGE      MAXIMUM AVAILABLE      HIGH WATER      REGION SIZE
      50429950           290850406           67850238      1467981824
STACK CELLS:
CURRENT USAGE      MAXIMUM CELLS          HIGH WATER
           747              31559                2146
                     PROCESS STACK CELL USAGE
USER      JOBNAME  ASID         PID        PPID STATE    THREADS STACKS
DOMINO    DOMINO9  0191    50463207      131551 HS------      155     156
WEBSRV    IMWEBSRV 013B    16908322           1 HK------       46      85
```

kernel private storage usage health check

USS_KERNEL_PVTSTG_THRESHOLD

Description:

This check monitors the current usage of UNIX System Services kernel private storage below the bar against a suggested threshold.

Reason for check:

At kernel initialization 80% of the available region is designated for stack cell pool cells and the remaining 20% for kernel and system usage. Because the stack cell pool is only extended as needed all of the storage that is designated for stack cell might not be allocated. If the allocation of storage designated for kernel and system usage overflows into the storage designed for stack cells, the number of stack cells that can be allocated is reduced. Reducing the number of stack cells that can be allocated can reduce the system-wide UNIX workload capacity. This check notifies the installation of the potential reduced capacity.

monitor stack cell pool usage health check

USS_KERNEL_STACKS_THRESHOLD

Description:

This check monitors the current usage of z/OS UNIX kernel stacks cell pool cells against a suggested threshold.

Reason for check:

If the number of kernel stack cell pool cells in use reaches the system maximum, additional cells cannot be allocated and system calls that use the kernel address space are disallowed. By monitoring stack cell usage installations might be able to prevent impacts to critical workloads by quiescing noncritical workloads before the supply of stack cells is exhausted.

RAS Enhancements

- Enhance Member Gone processing (in a Shared File System configuration) to initiate a sysplex-wide dump should the sysplex root fail to recover when the current file system owner fails and there is no ALTROOT configured. Also initiate a sysplex-wide dump if the ALTROOT hot swapin fails. This is provided for better problem determination should this type of failure occur.
- New EC6 ABEND reason – xxxx0796 (NoRootFileSystem)

- Enhance F OMVS,NEWROOT processing to audit its processing so that customers can determine the progress of the NEWROOT processing should the processing be delayed.

F OMVS,NEWROOT auditing

- Each system issues hardcopy message BPXF267I when conversion for a file system begins:

  BPXF267I SYSPLEX ROOT REPLACEMENT PROCESSING IS CONVERTING FILE SYSTEM *fsname*.

- Each system issues hardcopy message BPXF268I when NEWROOT conversion completes:

  BPXF268I SYSPLEX ROOT REPLACEMENT PROCESSING HAS COMPLETED CONVERTING ALL FILE SYSTEMS.

Security Enhancement-necessary for customers to pass security audits.

- New BPXPRMxx statement: PWT
  - PWT(SMF|ENV|SMFENV)
- Allow automatic log off of unattended terminals after a period of inactivity.
- The options on this statement provide the ability to indicate if the SMFPRMxx JWT/SWT/TWT values are to be honored.
- new kernel environment variable: _BPXK_TIMEOUT, allows job timeout value to be overridden on an individual process basis

See *z/OS V2R1.0 UNIX System Services Planning* for details

Miscellaneous Enhancements

- Removal of BPX.DEFAULT.USER -- you should use BPX.UNIQUE.USER instead.

- __login() service was be modified to support SURROGAT class profiles.
  - This allows applications with appropriate privileges to issue the __login() service without specifying a password.

- Boost the limit on the number of mutexes and condition variables.
  - New resource in the UNIXPRIV class: SUPERUSER.SHMMCV.LIMIT
    - With READ access, allows the user to create up to 4,194,304 mutexes or condition variables to be associated with a single shared memory segment.
    - With READ access, allows 134,217,728 (128M) for the system.

References

- SA23-2283-00 z/OS V2R1.0 Using REXX and z/OS UNIX System Services
- SA23-2280-00 z/OS V2R1.0 UNIX System Services Command Reference
- SA23-2285-00 z/OS V2R1.0 UNIX System Services File System Interface Reference
- SA23-2284-00 z/OS V2R1.0 UNIX System Services Messages and Codes
- GA32-0884-00 z/OS V2R1.0 UNIX System Services Planning
- SA23-2282-00 z/OS V2R1.0 UNIX System Services Programming Tools
- SA23-2281-00 z/OS V2R1.0 UNIX System Services Programming: Assembler Callable Services Reference
- SA23-2279-00 z/OS V2R1.0 UNIX System Services User's Guide
- SC23-6887-00 z/OS V2R1.0 Distributed File Service zFS Administration
- SC23-6885-00 z/OS V2R1.0 Distributed File Service Messages and Codes

Trademarks and Disclaimers

- **See http://www.ibm.com/legal/copytrade.shtml for a list of IBM trademarks.**
- **The following are trademarks or registered trademarks of other companies**
  - UNIX is a registered trademark of The Open Group in the United States and other countries
  - CERT® is a registered trademark and service mark of Carnegie Mellon University.
  - ssh® is a registered trademark of SSH Communications Security Corp
  - X Window System is a trademark of X Consortium, Inc
- **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.

Connect with IBM System z on social media!

Subscribe to the new *IBM Mainframe Weekly* digital newsletter to get the latest updates on the IBM Mainframe!

LinkedIn

System z Advocates **
IBM Mainframe- Unofficial Group
IBM System z Events
Mainframe Experts Network
SHARE

facebook

IBM System z **
IBM Master the Mainframe Contest
IBM Destination z
SHARE Inc.

twitter

IBM System z **
IBM System z Events
Destination z
SHARE

System z SMEs and Executives:
Deon Newman - @deonnewm
Steven Dickens - @StevenDickens3
Michael Desens - @MikeDesens
Patrick Toole - @Pat_Toole_II
Kelly Ryan - @KellykmRyan
Richard Gamblin - @RichGx

Blogs

IBM Mainframe Insights **
Millennial Mainframer
#MainframeDebate blog
SHARE blog
IBM Destination z

YouTube

IBM System z **
Destination z

tumblr.

IBM Mainframe50

Include the hashtag #mainframe in your social media activity and #mainframe50 in 50th anniversary activity