ibm.com

Real Storage Management (RSM) BCP Allocation

Redbooks

International Technical Support Organization

Trademarks

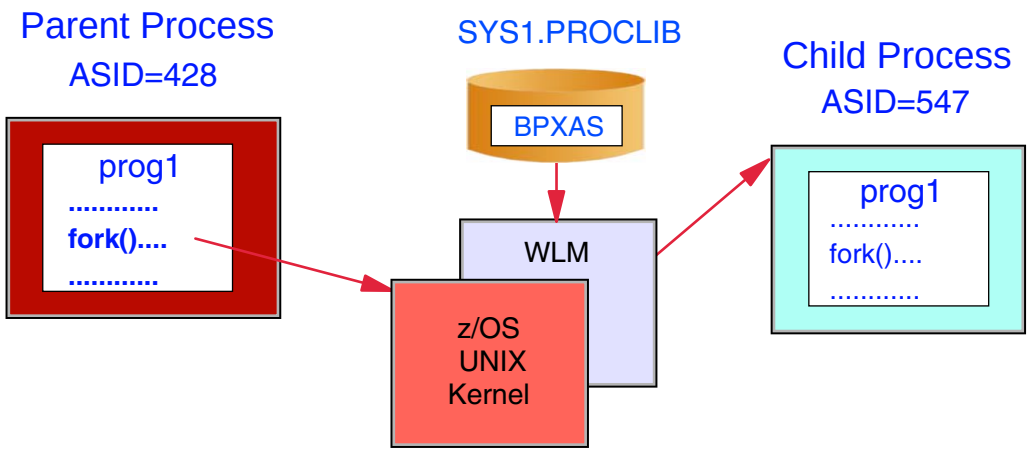
eNetwork	DFSMS/MVS	IMS	RMF
geoManager	DFSMSdfp	IMS/ESA	RS/6000
AD/Cycle	DFSMSdss	IP PrintWay	S/390
ADSTAR	DFSMShsm	IPDS	S/390 Parallel Enterprise Server
AFP	DFSMSrmm	Language Environment	SecureWay
APL2	DFSORT	Multiprise	StorWatch
APPN	Enterprise System 3090	MQSeries	Sysplex Timer
BookManger	Enterprise System 4381	MVS/ESA	System/390
BookMaster	Enterprise System 9000	Network Station	System REXX
C/370	ES/3090	NetSpool	SystemView
CallPath	ES/4381	OfficeVision/MVS	SOM
CICS	ES/9000	Open Class	SOMobjects
CICS/ESA	ESA/390	OpenEdition	SP
CICS/MVS	ESCON	OS/2	VisualAge
CICSPlex	First Failure Support Technology	OS/390	VisualGen
COBOL/370	FLowMark	Parallel Sysplex	VisualLift
DataPropagator	FFST	Print Services Facility	VTAM
DisplayWrite	GDDM	PrintWay	WebSphere
DB2	ImagePlus	ProductPac	3090
DB2 Universal Database	Intelligent Miner	PR/SM	3890/XP
DFSMS	IBM	QMFr	z/OS
	IBM System z	RACF	z/OS.e

Domino (Lotus Development Corporation)
DFS (Transarc Corporation)
Java (Sun Microsystems, Inc.)
Lotus (Lotus Development Corporation)
Tivoli (Tivoli Systems Inc.)
Tivoli Management Framework (Tivoli Systems Inc.)
Tivoli Manger (Tivoli Systems Inc.)
UNIX (X/Open Company Limited)
Windows (Microsoft Corporation)
Windows NT (Microsoft Corporation)

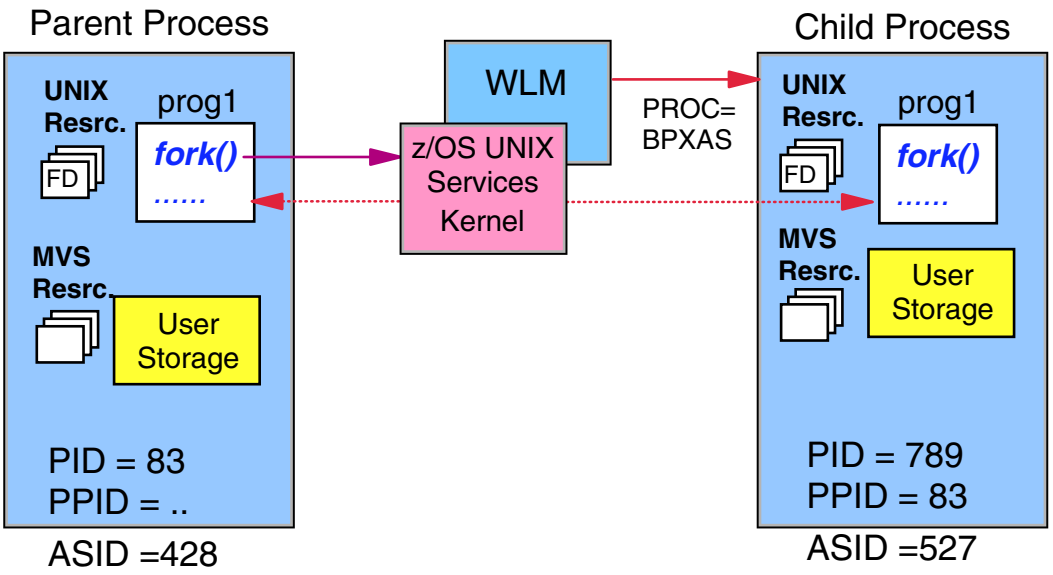
z/OS UNIX fork() Processing

- Fork() is a POSIX/XPG4 function that creates a duplicate process referred to as a child process
  - The process that issues the fork() is referred to as the parent process
- With z/OS UNIX, a program that issues a fork() function creates a new address space that is a copy of the address space where the program is running
- The fork() function does a program call to the kernel, which uses WLM/MVS facilities to create the child process
- The contents of the parent address space are then copied to the child address space

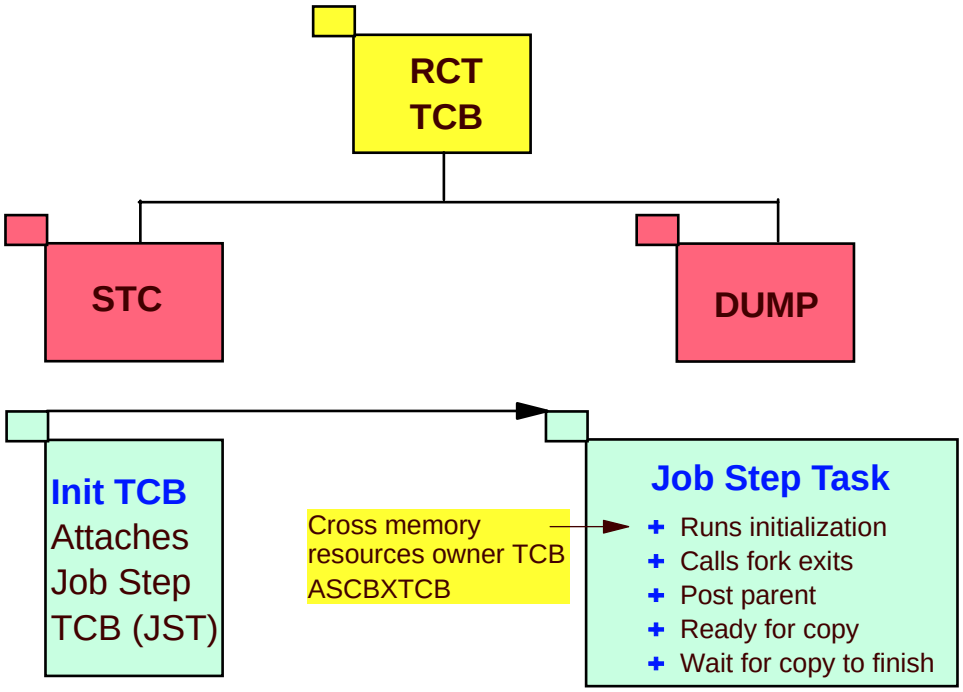
Create a Process



Create a Process Using Fork



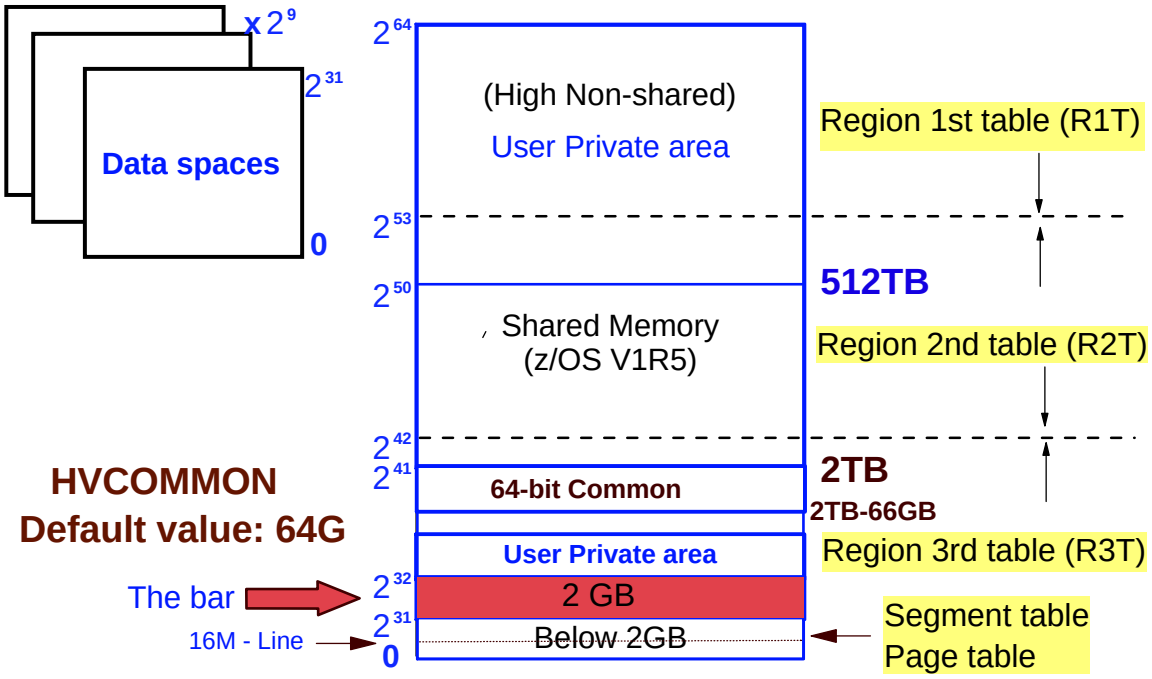
TCB Structure in Child Address Space



Current Problems with fork() Processing

- fork() 64-bit copy processing fails when the high virtual storage is allocated in the child address space at the time when the copy of the 64-bit parent address space is copied into the child space
- This causes the following to occur:
  - RSM fails the copy with Return Code 8 Reason code '6E000300'x and the child address space contains memory objects that were previously allocated.
  - The fork() function fails with a Return code 70, Reason code '0B1505C1', as the invocation of the IARV64FC service fails - The documented action for this failure was to retry the operation at a later time

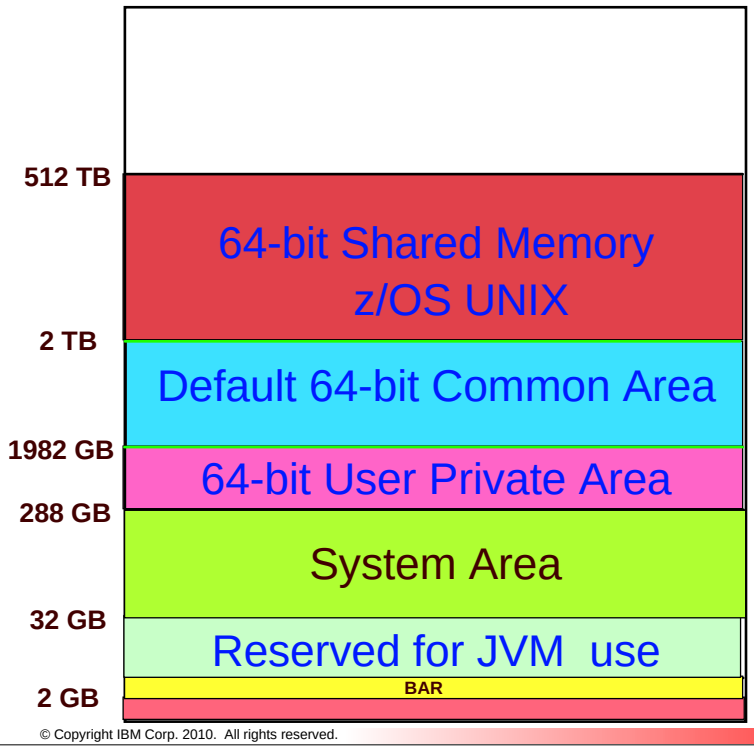
z/OS V1R10 Addressability with HVCOMMON in IEASYSxx Member



z/OS V1R12 Enhancements

- Support a new system area in the storage above the 2 GB bar
- Storage is designed to be equivalent to LSQA below the 2 GB bar
- Storage is this area will not be copied during the fork() process when RSM copies the parent storage to the child address space
- New keyword is added to the:
  - IARV64 REQUEST=GETSTOR, **LOCALSYSAREA=NO|YES**
    - Indicate that the memory object should be allocated from the system area of the 64-bit address space map

Storage Map



RSM Support for SDUMP

- Real storage constraint problems with SVC DUMPs
  - When large amount of data paged-in from auxilliary storage
  - When capturing large amount of component exit data
  - Auxilliary storage causes pressure on real memory and may force page-out o important data
- z/OS V1R12 SDUMP improvements
  - Reduce the dump capture time and system non-dispatchable time
  - SDUMP target area can be a dataspace or an area within a 64-bit memory object

VERBX IEAVTSFS Command

- VERBX IEAVTSFS command has been enhanced to display the SDUMP data capture statistics
- The IPCS VERBX IEAVTSFS command displays the statistics for various phases of the dump capture process:
  - A set of statistcs for the global storage capture phase
  - A set of statistcs for each address space included in the dump
  - A set of statistcs for the dump exit phase of the dump

Output from VERBX IEAVTSFS Command

- SDUMP statistics for the global data capture phase of the dump and for the capture phase of ASID X'13'

```
Global storage start              04/30/2010 12:06:31.726449
Global storage end                04/30/2010 12:06:31.808150
Global storage capture time       00:00:00.081700

  Defers for frame availability              0
  Pages requiring input I/O                 59
  Source page copied to target            9966
  Source frames re-assigned                 59
  Source AUX slot IDs re-assigned            0
Asid 0013:
  Local storage start             04/30/2010 12:06:31.851937
  Local storage end               04/30/2010 12:06:31.903105
  Local storage capture time      00:00:00.051167
  Tasks reset dispatchable        04/30/2010 12:06:31.903248
  Tasks were nondispatchable      00:00:00.051310

  Defers for frame availability              0
  Pages requiring input I/O                 50
  Source page copied to target            1708
  Source frames re-assigned                 66
  Source AUX slot IDs re-assigned            0
```


VERBX IEAVTSFS Dump Exit Statistics

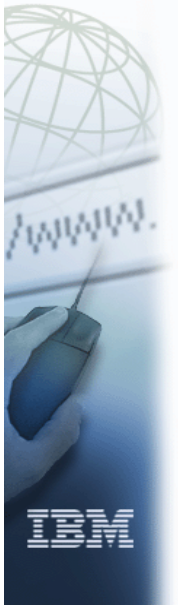
- Sample output from VERBX IEAVTSFS command showing the SDUMP statistics for the SDUMP exits phase of the dump

```
Dump Exits
Exit address                      0755E6A0
  Home ASID                       0005
  Exit start                      04/30/2010 12:06:31.740223
  Exit end                        04/30/2010 12:06:31.740226
  Exit time                       00:00:00.000002
  Exit attributes:  Sdump, Early Global

  Defers for frame availability              0
  Pages requiring input I/O                  3
  Source page copied to target            5239
  Source frames re-assigned                  3
  Source AUX slot IDs re-assigned            0
```

ibm.com

e-business



BCP Alloation Improvements

Redbooks

International Technical Support Organization

New MEMDSENQMGMT Keyword

- Allows jobs and subsystems to use memory-based data set ENQ management for dynamically allocated data sets
  - This parameter moves data set ENQ information created by allocation out of SWA and into efficient memory structures
  - Memory-based data set ENQ management is faster than the other option, SWA-based data set ENQ management
    - For jobs that allocate a large number of data sets, such as DB2

New MEMDSENQMGMT Keyword

- ALLOCxx parmlib member - MEMDSENQMGMT specifies whether the feature is available for exploitation by jobs and subsystems as follows:
  - MEMDSENQMGMT(ENABLE | DISABLE)
  - **ENABLE** - Allows jobs and subsystems to use memory-based data set ENQ management for dynamically allocated data sets
  - **DISABLE** - Disables jobs and subsystems from using memory-based data set ENQ management for dynamically allocated data sets

Using MEMDSENQMGMT Function

- New macro service - IEFDDSRV macro instruction
  - When MODIFY and TYPE=FEATURE are specified
    - These are required parameters that indicates how allocation should manage ENQs on the data set name when you specify - DSENQMGMT=MEMORY

```
IEFDDSRV MODIFY
,TYPE=ALLOCATION Default: TYPE=ALLOCATION
,DDNAME=ddname ddname: RS-type address or register (2) - (12) ASM only.
,DSABPTR=dsabptr dsabptr: RS-type address or register (2) - (12) ASM only.
,NEWDDNAME=newddname newddname: RS-type address, or register (2)-(12).
,TYPE=FEATURE
,DSENQMGMT=NO_CHANGE
,DSENQMGMT=MEMORY
```

New MEMDSENQMGMT Benefits

- Heavy data set allocation users should see a significant reduction of CPU time spent to manage serialization of access to data sets
  - Also a reduction of time to recovery after outages, when it becomes necessary to restart the job or subsystem
  - Better performance from subsystems when many data sets are used
    - Provides a better Mean Time To Recover from errors affecting subsystems like DB2
    - Allows installations to consolidate more work load onto z/OS with less performance impact

Migration Considerations

- When exploiting this feature, consider these options:
  - Programs exploiting the DD accounting suppression function will have less data in the SMF Type 30 EXCP section or Type 40
  - VSAM-related SMF record types are not affected
  - Programs should not request this function for non-VSAM data sets
- This new function requires the addition of MEMDSENQMGMT(ENABLE) in the ALLOCxx parmlib member for exploitation
- **Note:** When using this function for DB2, an IPL is required
  - Also it is preferable when you are running with DSMAX at 64K or 100K

Migration Considerations

- There is an option to **suppress DD accounting** in the SMF Type (14, 30 and 40) records by:
  - A program request when using the S99DASUP flag on DynAlloc request
    - You can suppress creation of SMF Type 30 EXCP section data on a per-DD basis
- S99DASUP - Used by authorized programs to
  - Setting this bit can affect the SMF data created for:
    - The EXCP section of SMF Record Type 30
    - SMF Record Type 40
    - SMF Record Type 14 - fields SMF14NTR and SMF14NER

Recommended for programs allocating VSAM data sets with generated DD names

Duplicate Temporary Data Set Name

- In previous z/OS versions, when two jobs, with the same jobname, using named (&&mydsn) temporary data sets
  - Scheduled in the same system, at the same time
  - The second would be canceled with a JCL error
  - WHY: - duplicate temporary data set name
- With z/OS V1R12, the behavior to generate data set names using the label or unique number can be controlled via the ALLOCxx parmlib member
  - SYSTEM TEMPDSFORMAT setting

Temporary Name - DSN=&&LABEL

- SYSTEM TEMPDSFORMAT(UNIQUE | INCLUDELABEL)
  - **UNIQUE** - Indicates that jobs with same jobname that includes temporary data sets with DSN=&&LABEL running simultaneously, NO temporary data sets with same names
    - SYSyyddd.Thhmmss.RA000.jjobname.Rggnnnnn
    - SYS10141.T143444.RA000.HEUSERJ.R0100590
  - **INCLUDELABEL** - Indicates that when the system processes JCL that includes temporary data sets with DSN=&&LABEL, the generated data set name will include the &&label specified in the JCL
    - SYSyyddd.Thhmmss.RA000.jjobname.dsetname.Hgg
    - SYS10141.T151047.RA000.HEUSERP.MYTEMP.H01

```
gg=01 - means in a sysplex
dsetname - 1 to 8 character DSNAME following the two ampersands (&&)
nnnnn - a number that is unique within a system
```


Commands Temporary Name Support

- Display ALLOCxx parmlib member

```
D ALLOC,OPTIONS
.................
 VERIFY_VOL      POLICY:            YES
 SYSTEM          IEFBR14_DELMIGDS:  LEGACY
                 TAPELIB_PREF:      EQUAL
                 REMIND_INTV:       90
                 VERIFY_UNCAT:      FAIL
                 TEMPDSFORMAT:      UNIQUE
                 MEMDSENQMGMT:      DISABLE
```

- Change TEMPDSFORMAT parameter either on the ALLOCxx parmlib member or by SETALLOC command

```
SETALLOC SYSTEM,TEMPDSFORMAT=INCLUDELABEL
IEFA010I SETALLOC COMMAND SUCCESSFUL
TEMPDSFORMAT SET TO INCLUDELABEL.
```