

z/OS V2R1 zFS Function Update

Vivian W Morabito

Wednesday March 12, 2014 3:00 – 4:00 PM
Grand Ballroom Salon C



morabito@us.ibm.com

Agenda

New V5 Filesystems
Salvage/ Repair Updates
File System Quiesce
File System Backup
Defaults Changed
Discontinued Support

Terminology/Abbreviations

- **z/OS 13 = z/OS Version 1 Release 13**
- **z/OS 2.1 = z/OS Version 2 Release 1**
- **File System = Aggregate**

Because zFS has removed multi-file system aggregates and clones in z/OS 2.1, the term file system and aggregate can be considered synonymous.

This presentation uses the term file system where the publications might use the term aggregate and/or file system but they are equivalent now with z/OS 2.1.

- zFS file systems (aggregates) have a major and minor version stored inside them. The **zfsadm aggrinfo -long** command will show you these two numbers today:

example: version 1.4 will be shown if you issue this command on z/OS 13.

- **V4 = Version 1.4**, and **V5 = Version 1.5**

z/OS 2.1 introduces an optional version 1.5 aggregate (file system).

This presentation simplifies the terminology to v4 file system (current version in the field) and v5 file system (new optional file system with z/OS 2.1).

- A v5 file system has the option to store directories in a new format (a tree) for fast lookup/insert/delete. The publications refer to directories stored in this new format as “extended (v5) directories”. This presentation simplifies it to v5 directory and refers to the original directory format available today in the field as v4 directory.

V4 / V5 Filesystems Comparison

V4 filesystems (in z/OS V1R13 and prior):

- Has linear (v4) directory formats on disk
- V4 directories with thousands of names or more suffer significant performance degradation.
 - **>50,000 names and IBM recommends that HFS contain the directory** (but HFS is significantly worse for file IO)
- **V4 filesystems are still fully supported and will be for a long time (possibly forever)**

V5 filesystem (optional, in z/OS V2R1+ only):

- Provides new tree (v5) directory format: **faster insert/search/delete**
- Can contain both v4 (linear) and v5 (tree) directories
- Optional, not the default on z/OS V2R1
- Can only be mounted on z/OS V2R1
- IBM recommends you do not start to use V5 filesystems until you are fully on z/OS V2R1 with no plans to go back to prior releases.

V5 Filesystems:

- **4 billion+ limit of sub-directories in a directory**
 - (v4 limit is 65535)
- **16TB file system size limit** (file size still limited to 4TB)
 - (v4 limit is 4TB)
- **Can create a new v5 file system**
 - **zfsadm format/IOEAGFMT/IOEFSUTL**
- **New facilities to convert existing v4 file systems** (and their directories) to v5:
 - Offline conversion facility (**IOEFSUTL**):
 - *can convert v4 to v5 and also v5 to v4*
 - Online conversion facilities (**zfsadm convert / CONVERTTOV5 ...**)
 - *v4 to v5*
 - New query and APIs to determine directory version (4 or 5) (**zfsadm fileinfo**)

Version 5 File Systems- Performance Preliminaries



z/OS 2.1 Improves zFS Directory Performance:

New v5 file system directory trees make directory name lookup, insertion, & removal fast for larger directories.

- Directory searches scale well as names are added to the dir
- Directory searches can be performed in parallel (most of the time) with directory updates to the same directory.

For **both** v4 and v5 file systems:

z/OS UNIX and zFS teams improved readdir+ performance for large directories.

→ (used by **ls -l** and NFS server for example)

zFS greatly improved sysplex client directory operation performance ... by reducing communications between clients and servers for readdir operations.



Performance Workload Descriptions:



- Performance results (on subsequent slides) were obtained using 3 workloads created by IBM:
 - All workloads involved many tasks processing 2000 objects in each directory, for multiple directories, on multiple file systems (see slide notes).
 - **ptestDL2** – This workload did repeated lookup (name searches) by the tasks.
 - **ptestDL** – The tasks did repeated lookup and readdir functions.
 - **ptestDU** – The tasks performed directory create/update/remove/readdir/search.
- Definitions:
 - **External Throughput (E)** – Number of operations per unit of time.
 - higher the number → lower the average operation response time.
 - **Internal Throughput (I)** – Number of operations per unit of processor time.
 - higher the number → less CPU time each operation took.
- Tests were run varying the sizes of the involved directories to test scalability.

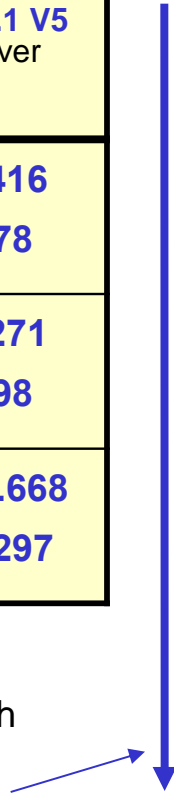
Version 5 File Systems – Performance Results ptestDL2



- ptestDL2 (Directory Search) Results on zEC12 / FICON connected DASD

	Monoplex Results			Sysplex Client Results		
Directory Sizes	R13 Operations / Second	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13	R13 Operations / Second I per processor	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13
0 Base Names (2000 names per directory)	E=307703 I=307703	EΔ 1.005 IΔ 1.005	EΔ 1.197 IΔ 1.197	E=232427 I=55067/proc	EΔ 0.996 IΔ 0.980	EΔ 1.416 IΔ 1.378
18k Base Names (20000 names per directory)	E=80840 I=80840	EΔ 0.964 IΔ 0.964	EΔ 4.536 IΔ 4.536	E=44532 I=10536/proc	EΔ 0.933 IΔ 0.933	EΔ 7.271 IΔ 7.098
48k Base Names (50000 names per directory)	E=34598 I=34598	EΔ 0.964 IΔ 0.964	EΔ 10.026 IΔ 10.026	E=18308 I=4333/proc	EΔ 0.918 IΔ 0.918	EΔ 16.668 IΔ 16.297

- V5 file system search performance scales almost linearly with directory size
- V5 Sysplex client performance improves 40% even for small directories, 16X for directories with 50,000 names.
- **vnode_cache_size=400000**, **meta_cache_size=100M**, **metaback_cache_size=2G**
- V5 file system monoplex performance improves 20% for small directories, 10X for directories with 50,000 names.



Version 5 File Systems – Performance Results ptestDL



- ptestDL (Dir. Search+Readdir) Results on zEC12 / FICON connected DASD

	Monoplex Results			Sysplex Client Results		
Directory Sizes	R13 Operations / Second	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13	R13 Operations / Second I per processor	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13
0 Base Names (2000 names per directory)	E=297285 I=297345	EΔ 1.002 IΔ 1.002	EΔ 1.167 IΔ 1.167	E=216433 I=50474/proc	EΔ 1.064 IΔ 1.078	EΔ 1.454 IΔ 1.452
18k Base Names (20000 names per directory)	E=33908 I=33918	EΔ 2.032 IΔ 2.032	EΔ 5.978 IΔ 5.977	E=2620 I=638/proc	EΔ 12.834 IΔ 8.840	EΔ 77.549 IΔ 51.830
48k Base Names (50000 names per directory)	E=12717 I=12720	EΔ 2.258 IΔ 2.258	EΔ 9.160 IΔ 9.160	E=384 I=97/proc	EΔ 35.490 IΔ 22.237	EΔ 276.67 IΔ 174.81

- Since readdir is in the mix, its response time is dependent on directory size
- V5 monoplex performance: improves 17% for small, 9X for larger directories.
- V5 sysplex client performance improves 45% for small, 277X for larger directories
- Sysplex clients do server communication for attributes of base files

▪ → This is why ETR differs from ITR, we improved performance for client to server communications for both v4 and v5 file systems, hence the large improvement.



Version 5 File Systems – Performance Results ptestDU



- ptestDU (Dir. Reading and Writing) Results on zEC12 / FICON connected DASD

	Monoplex Results			Sysplex Client Results		
Directory Sizes	R13 Operations / Second	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13	R13 Operations / Second I per processor	z/OS 2.1 V4 Ratio over R13	z/OS 2.1 V5 Ratio over R13
0 Base Names (2000 names per directory)	E=109584 I=119372	EΔ 1.055 IΔ 1.022	EΔ 1.167 IΔ 1.167	E=65384 I=11076/proc	EΔ 1.053 IΔ 1.059	EΔ 1.249 IΔ 1.226
18k Base Names (20000 names per directory)	E=31688 I=31943	EΔ 1.566 IΔ 1.483	EΔ 3.313 IΔ 3.586	E=7441 I=1675/proc	EΔ 3.528 IΔ 2.780	EΔ 8.940 IΔ 6.822
48k Base Names (50000 names per directory)	E=12667 I=12682	EΔ 1.741 IΔ 1.751	EΔ 6.158 IΔ 7.069	E=830 I=205/proc	EΔ 17.472 IΔ 12.568	EΔ 65.110 IΔ 47.891

▪ V5 monoplex performance: improves 17% for small, 7X for larger directories.

▪ V5 sysplex client performance improves 25% for small, 65X for larger directories

▪ Results from last two rows in table hurt by small meta cache size:

▪ → Due to zFS storage constraints, it was not possible to run with larger cache

▪ → IBM working on solution.

*
*



■ Typical Customer Usage Pattern:

1. Directory search (lookup) most common operation, or at least one of the most frequent.
2. File Open/Read/Write/Close the next most common operations
→ **Note: z/OS 2.1 file IO performance is equivalent to z/OS 13 file IO performance.**
3. Directory update operations generally a much lower percentage of calls to zFS.
→ **F ZFS, QUERY, KNPFS** will show you your workload characteristics in terms of what operations are most common for you.

Benefits for your site (cont'd)

■ The Bigger the Directory...

- As shown in the prior slides, directory performance is improved even for smaller directories (2000 names) but directories over 10,000 names will likely see non-trivial reduction in directory operation time inside zFS.
- The **F ZFS,QUERY,FILESETS** can be used to identify your most active file systems.
- And the **ls -slk** will show your directory size on disk in kilobytes.
 - Any directory over 160K in size is going to get a benefit with v5

Note: Above commands can be used to help determine v4→v5 conversion candidates.

Benefits for your site: Summary

- **Sysplex Client – Nice Performance Gain with V5 (and even V4)**
- **Summary:**
 - **If zFS (or HFS) is heavily used in your installation, AND you have:**
 - **High usage zFS or HFS file systems, or file systems with active larger directories AND/OR**
 - **You are using zFS sysplex RWSHARE support and access data on a non-owner.**
 - **Then zFS V5 File System and the z/OS 2.1 enhancements are going to help your environment.**

Disk Space Usage:

- V5 file system pages use a prefix/postfix scheme to pack more names in a directory page than v4 if the names fit a common pattern at the beginning or end of the name.
 - Can result in 4X more names in a page over a v4 file system.
- V4 file systems can place a name in any directory page that can contain the name.
 - V5 directories must put the name in the page in the tree that is supposed to contain the name, thus leaving less flexibility in placement.
- Thus there are cases where a v5 directory could use less space than a v4 directory and situations where it could use more.
- **IBM expects that the disk space used by v5 directories to be roughly equivalent to v4 directories on average.**

Version 5 File Systems – More Details

- A file system that is v5 can be up to 16TB large and any **new** directories created in it will be v5.
- A v5 filesystem can contain both v4 and v5 directories.
 - To support migration of an existing v4 filesystem to v5.
- Options for conversion of existing directories:
 - Leave them all v4 (new dirs will be v5)
 - Convert them as they are accessed
 - Convert them at the time the filesystem is converted (use IOEFSUTL off-line utility)

Version 5 File Systems – More Details

V5 File Systems Can Only Be Processed on z/OS 2.1

- zFS will not allow creation of a v5 or conversion of a v4 to v5 if R13 or prior members exist in a sysplex.
- R13 and prior systems will fail MOUNTs of v5 file systems.
- Do not use v5 file systems until all systems at your site are z/OS 2.1 or later.
- **IOEFSUTL converttov4** can convert a v5 file system back to v4 in most cases.

Version 5 File Systems – Conversion Details



- **Changing File System from v4 to v5:**
 - Fast operation, simply updates some global fields on disk.
 - After this conversion, any future directories created are v5
 - File system can be up to 16TB in size
 - And allows the individual directories to be converted from v4 to v5.



V5 File Systems – Conversion Details

- **Converting a v4 Directory to a v5 Directory:**
 - Stops user activity to the directory (if the file system is mounted)
 - Creates a new hidden v5 directory (gets deleted if system crashes at next mount time)
 - Copies names from original v4 directory to hidden v5
 - This could require disk space, and if file system space exhausted will dynamically grow file system if dynamic grow is allowed for the file system.
 - If prior step successful, atomically swaps the hidden v5 with the v4 making the v4 hidden.
 - A crash after this point in time will delete original hidden v4 and the new v5 remains.
 - Delete hidden v4 directory
 - Resume user activity
 - A conversion from v5 to v4 works the same way, just swap the terms v4 and v5.

Version 5 File Systems – Kernel Parameters and MOUNT



- **Kernel Conversion Parameters (IOEPRMxx)**
 - **CHANGE_AGGRVERSION_ON_MOUNT = ON/OFF**
 - Will set file system version to 5 at R/W mount time, fast operation.
 - **CONVERTTOV5=ON/OFF**
 - Will set file system version to 5 (if not already) and convert root directory at R/W mount time (if not already in the new format)
 - Will convert each old-format v4 directory to the new format *upon first access*.
 - **FORMAT_AGGRVERSION = 4/5** – controls whether zfsadm format, ioefsutl and ioeagfmt, by default create a version 4 or 5 file system.
 - This avoids the need to update existing programs and JCL to specify that you want a v5 file system.
 - Automount, ISHELL use this parameter.
- **MOUNT parameter**
 - **CONVERTTOV5/NOCONVERTTOV5** – Overrides IOEPRMxx **CONVERTTOV5** setting for file system. If not specified then IOEPRMxx **CONVERTTOV5** option takes effect.
- **IOEAGFMT/zfsadm format – existing format programs**
 - **-version4/-version5** – New parameters specify the version of the file system to create.
 - Default is to get value of **format_aggrversion** parameter from zFS kernel to determine file system version to create. zFS kernel must be active.



Version 5 File Systems – IOEFSUTL and salvage



- **IOEFSUTL** – new batch program with 4 commands for non-mounted file systems:
 1. **Format** - Can format/create new file system (v4 or v5).
 2. **Converttov5** – Converts a v4 file system and all its directories to v5.
 3. **Converttov4** - Can convert existing version 5 file systems back to version 4 format:
 - IFF – no directory has more than 65535 sub-directories
 - IFF – file system is < 4TB in size.
 4. **Salvage** - Can verify and repair both v4 and v5 file systems like **ioeagslv** does.
 - **IOEFSUTL** is the IBM future direction for zFS batch programs, although **ioeagfmt** and **ioeagslv** will remain supported, new batch function will be placed in **ioefsutl**.

- **IOEAGSLV** – completely re-written to support both v4 and v5 file systems.
 - Scales to largest zFS file systems
 - Has new messages that provide more details on the progress and results of the verification and repair.
 - Designed to repair in one pass, though two recommended if a repair required.



Version 5 File Systems –zfsadm convert



- **zfsadm convert** – new command, operates on mounted file systems:

- z/OS UNIX shell command that can:

`zfsadm convert –aggrversion filesystemname`

- Only changes the v4 filesystem to v5 (no directories)

`zfsadm convert –path pathname`

- Convert a specific directory to v5
- If filesystem not already changed to v5, it will change to v5
- Makes sure local application do not have directory open, but cannot check NFS/SMB remote applications, so be careful.



Version 5 File Systems – zfsadm fileinfo

- **zfsadm fileinfo** – displays detailed file information:
 - **zfsadm fileinfo -path <pathname> [{-globalonly | -localonly | -both}]**

```

path: /home/suimghq/lfsmounts/PLEX.ZFS.FS2/DCEIMGHQ.ftestLD.pb/.
***  global data  ***
fid                7,174515          anode              130931,2028
length            33546240          format            BLOCKED
1K blocks         21664            permissions       755
uid,gid           0,10             access acl        33,72
dir model acl     33,72           file model acl    32,72
user audit        F,F,F           auditor audit     N,N,N
set sticky,uid,gid 0,0,0          seclabel          none
object type       DIR             object linkcount  39686
object genvalue   0x00000000      dir version       5
dir name count    39686          dir data version  160308
dir tree status   VALID          dir conversion    na
file format bits  na,na,na       file charset id   na
file cver         na             charspec major,minor na
direct blocks     0x000D91FE     0x000D91FF       0x000D9200       0x000D9201
                  0x000D9202     0x000D9203       0x000D9204       0x000D9205
indirect blocks   0x000D9206     0x00052199
mtime            May 28 17:09:45 2013  atime              May 28 17:09:19 2013
ctime            May 28 17:09:45 2013  create time        May 28 15:14:48 2013
reftime         none

```

Version 5 File Systems - Exploitation Strategy



1. **Delay converting remaining HFS to zFS file systems until z/OS 2.1:**
 - Unless you have an immediate need to convert an HFS to zFS (typically due to file IO performance issues with HFS):
 - Wait until your site is at z/OS 2.1 and later before migrating the file system to zFS
 - So you can use v5 format for improved directory performance and avoid a conversion from v4 to v5 format.
 2. **Do not use v5 until fully ready to commit to z/OS 2.1 for all systems.**
 - Going back to v4 via IOEFSUTL could be painful, wait until its safe before using v5.
 3. **Set `format_aggrversion=5`:**
 - Future file systems get created as version 5
 - And avoids having to change JCL and other programs.
 4. **Set `change_aggrversion_on_mount=ON`**
 - Safe since it is a fast operation and ensures future directories are v5 format.
 5. **Determine if `CONVERTTOV5` can be globally enabled**
 - This depends on how many directories are accessed at IPL time, the number of names in each and the known zFS conversion performance results and the expected amount of delay to the system IPL.
 - User has to decide if they can tolerate the expected one-time IPL delay
 - And if they can, simply specify `CONVERTTOV5=ON` in `IOEFSPRM`.
 6. **And if cannot globally enable, then:**
 - Determine highest usage file systems via RMF, or `F ZFS,QUERY,FILESETS` and commands shown on slide 9.
 - Also look at file systems with large directories, especially if they are high usage and then use the `CONVERTTOV5 MOUNT` parameter selectively.
- 23 Complete your session evaluations online at www.SHARE.org/AnaheimEval



New Salvager I - Summary



- **Completely rewritten in z/OS 2.1:**
 - Scales to largest file systems, avoiding abends present with the previous salvager.
 - Runs much faster
 - Note: Large file systems may still take a long time to verify and repair
 - Uses official z/OS messages for all communications to the user
 - The inputs to the program are mostly the same
 - Can specify a dataset that contains parameters much like the zFS kernel and/or use parmlib search.
 - But the output messages are completely different as shown in following slides.
- **Supports both V4 and V5 File Systems.**
- **Invoked via:**
 - **IOEAGSLV** – for compatibility with existing programs.
 - **IOEFSUTL salvage** – preferred method of invocation for the future.
- **Can Verify and Repair in One Pass**
 - If corrupt, will mark file system damaged, and if not repaired, a subsequent **MOUNT** would show message number **IOEZ00783E**.
 - However IBM recommends that if corruptions are found:
 - Perform a second repair pass to prove file system is truly corrected.
 - If not corrected then report the problem to IBM.
 - Should always be certain a file system is truly fixed before using it.



New Salvager II – Sample Verification

```
IOEZ00559I zFS IOEFSUTL: Initializing z/OS zFS
Version 02.01.00 Service Level s15.
Created on Tue Feb 5 09:15:55 EST 2013.
Address space asid x1C
IOEZ00178I CFCIMGNJ.PARMLIB(IOEFSPRM) is the configuration dataset currently in use
IOEZ00707I Log file size 107 8K blocks, verified correct
IOEZ00729I Verification of aggregate SUIMGNJ.HIGHRISK.TEST started
IOEZ00705I Formatted v5 aggregate size 10800 8K blocks, dataset size 10800 8K blocks
IOEZ00707I Log file size 107 8K blocks, verified correct
IOEZ00709I Bitmap size 2 8K blocks, verified correct
IOEZ00782I Salvage has verified 17 of 175 pages in the anode table.
IOEZ00782I Salvage has verified 34 of 175 pages in the anode table.
....
IOEZ00782I Salvage has verified 153 of 1530 directories in the directory tree.
IOEZ00782I Salvage has verified 306 of 1530 directories in the directory tree.
....
IOEZ00722I Primary file system size 465 8K blocks, verified correct
IOEZ00739I Salvage processed 1573 directory pages, 4128 anodes, 3 indirect blocks and 175
anode table pages.
IOEZ00730I Verification of aggregate SUIMGNJ.HIGHRISK.TEST completed, no errors found
```

- **Note that the new program has progress indicators to show its still running**
- **Shows how many objects were processed.**

New Salvager III – Sample Repair Case (File System was Corrupt)

```
....  
IOEZ00729I Verification of aggregate SUIMGNJ.HIGHRISK.TEST1 started  
IOEZ00705I Formatted v5 aggregate size 41040 8K blocks, dataset size 41040 8K blocks  
IOEZ00707I Log file size 107 8K blocks, verified correct  
IOEZ00709I Bitmap size 6 8K blocks, verified correct  
IOEZ00703E Corruption: Anode Table (T3): direct block index=6 physPage=8  
IOEZ00704E Expected: Anode Table (T3): physPage=8 not allocated to another object  
IOEZ00703E Corruption: Anode Table (T2): allocated=12032 visible=12032  
IOEZ00704E Expected: Anode Table (T2): calculated allocated=12024 visible=12024  
...  
IOEZ00723E Primary file system on aggregate SUIMGNJ.HIGHRISK.TEST1 is corrupted  
IOEZ00703E Corruption: Bitmap (B18): bits array in Bitmap physPage 00000001 logPage/offset/value=<0 2319  
000000FF>  
IOEZ00703E Corruption: Bitmap (B18): represented physPage 1386  
IOEZ00704E Expected: Bitmap (B18): bits array in Bitmap physPage 00000001 logage/offset/value=<0 2319  
00000000>  
.....  
IOEZ00739I Salvage processed 16916 directory pages, 47972 anodes, 5 indirect blocks and 1500 anode table  
pages.  
IOEZ00733I Verification of aggregate SUIMGNJ.HIGHRISK.TEST1 completed and the aggregate is in error  
IOEZ00735I Salvage found 16 minor errors of aggregate SUIMGNJ.HIGHRISK.TEST1  
IOEZ00753I Salvage is repairing anode table pages.  
IOEZ00755I Salvage is repairing the ZLC list.  
IOEZ00756I Salvage is repairing file, directory, and ACL objects.  
IOEZ00757I Salvage is repairing bitmap pages.  
IOEZ00758I Salvage repaired 1 anode table pages, 0 partially free pages, 0 zlc pages, 0 indirect blocks.  
IOEZ00787I Repair of aggregate SUIMGNJ.HIGHRISK.TEST1 completed
```

- **Message 703E/704E generally intended for IBM support.**
- **Messages 733I/758I/787I intended for user, shows if file system is corrupted and how many errors were repaired and if repair was successful.**
- You should run the program one more time and verify it indicates the file system is no longer corrupted if you see these.

Quiesce Updates I - Summary

- **Problem: D OMVS,F does not show quiesced status for zFS file systems:**
 - zFS was handling file system quiesce operations independent of z/OS Unix
 - z/OS Unix displays did not properly show file system quiesce status.
- **IBM has corrected problem in z/OS 2.1:**
 - NORWSHARE file system quiesce now handled by z/OS Unix
 - Guarantees z/OS Unix display command correct.
 - RWSHARE file system quiesce still handled by zFS
 - But zFS notifies z/OS Unix of the quiesce, and z/OS Unix shows it in a “PFS Exception” string in its display commands.
 - zFS has also updated its **zfsadm agrinfo –long** command to show more information about the quiesced file system.
 - Example shown here:

```
USSZFS.PLEX01.ZFS (R/W COMP QUIESCED): 2134143 K free out of total 2160000
version 1.4
auditfid D6C5F9F6 F7F70010 0000
266767 free 8k blocks; 7 free 1K fragments
21616 K log file; 56 K filesystem table
312 K bitmap file
Quiesced by job TOTTEN26 on system NP5 on Wed Feb 6 13:31:07.687179 2013
```

Quiesce Updates II – z/OS UNIX Display Examples

- NORWSHARE Examples:**

- D OMVS,F,N=USSZFS.PLEX01.ZFS**

```
PX0045I 15.14.39 DISPLAY OMVS 252
OMVS 0011 ACTIVE OMVS=(ST,R1,X9)
TYPENAME DEVICE -----STATUS----- MODE MOUNTED LATCHES
ZFS 425 QUIESCED RDWR 01/30/2013 L=245
NAME=USSZFS.PLEX01.ZFS 15.13.24 Q=245
PATH=/u/totten/testdir
MOUNT PARM= norwshare
OWNER=NP5 AUTOMOVE=Y CLIENT=N
QSYSTEM=NP5 QJOBNAME=TOTTEN26 QPID= 50595795
```

- df -Pkv /u/totten/testdir**

```
Filesystem      1024-blocks      Used      Available      Capacity      Mounted on
USSZFS.PLEX01.ZFS      0          0          0              0%
ZFS,      Read/Write,      Device:401,      ACLS=Y
norwshare
File System Owner : NP5      Automove=Y      Client=N
Quiesce Owner :      NP5 Quiesce      Jobname : TOTTEN26      Quiesce PID :50595795
Filetag : T=off codeset=0
Aggregate Name : USSZFS.PLEX01.ZFS
```

Quiesce Updates III – z/OS Unix Display Examples

- **RWSHARE Examples:**

- **D OMVS,F,N=USSZFS.PLEX01.ZFS**

```

BPXO045I 15.07.53 DISPLAY OMVS 248
OMVS 0011 ACTIVE OMVS=(ST,R1,X9)
TYPENAME DEVICE -----STATUS-----          MODE      MOUNTED  LATCHES
ZFS  423  ACTIVE          RDWR      01/30/2013  L=245
NAME=USSZFS.PLEX01.ZFS 15.05.59 Q=0
PATH=/u/totten/testdir
MOUNT PARM= rwshare
OWNER=NP5 AUTOMOVE=Y CLIENT=N
PFS INFO:
PFS EXCP: QUIESCED

```

- **df -Pkv /petzfs1**

```

Filesystem          1024-blocks          Used      Available  Capacity  Mounted on
OMVSSPT.PETZFS1.ZFS 2492640             973353    1519287    40%       /petzfs1
ZFS, Read/Write, Device:8793, ACLS=Y
File System Owner : Z1      Automove=U    Client=N
Filetag : T=off  codeset=0
Aggregate Name : OMVSSPT.PETZFS1.ZFS
PFS EXCP: QUIESCED

```

File System Backup Improvements

■ File system backup not quite right in prior releases:

- zFS was setting the change-activity flag after a file system was mounted in R/W mode even though nothing changed
- zFS did not set the change-activity flag after the first write after a successful backup
- Hence backups not taken when required, or taken when not required.

■ Change-Activity flag now set correctly:

- Will set the flag for first write after a R/W mount
- Will set the flag for first write after a backup
 - Backup program now tells zFS if backup successful
 - zFS will always set the flag for first write after a file system unquiesce (quiesce/unquiesce is used by backup).
- Will set the flag for log-file recovery at mount time (since that updates the file system).
- Salvage repair sets the flag.
- File system format commands with the **–overwrite** option also set the flag.

Defaults Changed I – Auditfid

- SMF type-80 records and RACF message ICH408I use a 16 byte identifier to identify an object that had some sort of authorization failure or state change of interest to the operator.
- zFS audit identifiers take one of two formats:

inode	unique	0	0
-------	--------	---	---

Standard version

volser	CCHH	inode	Un
--------	------	-------	----

Unique version

- The standard (prior default) version is not acceptable for truly identifying the object.
 - The unique version uses the volume serial and CCHH of the first extent of the file system, plus the inode and the low order part of the uniquifier to identify the object
- www.ibm.com/servers/eserver/zseries/zos/unix/bpxa1ty2.html - **auditid** tool available on z/OS Unix Tools website to display the object's pathname but needs the unique audit id version.
- Since proper auditing is desired, and because a proper auditid causes no system overhead, **z/OS 2.1 now has the following defaulted:**
 - **zfsadm format -newauditfid** – This option will store the proper unique audit id inside the new file system and it will be used in auditing records and messages.
 - **convert auditfid=on** – zFS startup parameter will auto-convert any mounted file system not using the unique method to begin using it.
- **zfsadm aggrinfo -long** – Note that this command will show you the audit fid of the file system (the first 10 bytes of audit id), this was not changed for z/OS 2.1.

Defaults Changed II – Cache Sizes

- **z/OS 13 and prior cache defaults:**
 - **user_cache_size=256M**
 - **meta_cache_size=64M**
 - **metaback_cache_size=0M**
 - These values are not representative of a typical customer need nor reasonable in current environments.

- **z/OS 2.1 changes above defaults (next slide)**
 - Because this represents a migration action, IBM has provided health checks:
 - **ZOSMIGV2R1_ZFS_VERIFY_CACHESIZE** – Used for z/OS 12 and 13 to warn you if you are affected by the change, and provide guidelines in case you want to accept the new defaults, or tells you how to keep the old sizes.

 - **ZFS_VERIFY_CACHESIZE** – Use for z/OS 2.1 and later. A best-practices check to alert you if you could get better performance by raising one of the cache sizes listed above.

- **z/OS 2.1 still honors your settings:**
 - If you took the time to tune zFS and specify a value for these in IOEFSPRM or in parmlib, then we use that value.

Defaults Changed III – New Cache Defaults



- **z/OS 2.1 USER_CACHE_SIZE default algorithm:**
 - If the **IOEFSPRM** (or parmlib search) does not specify a **user_cache_size** value:
 - Let $X = 0.10 \times \text{Real Storage Size (at zFS initialization time)}$
 - If $X < 256\text{M}$ then set **user_cache_size=256M**
 - Else if $X > 2\text{G}$ then set **user_cache_size=2G**
 - Else set **user_cache_size=X**

- **z/OS 2.1 META_CACHE_SIZE/METABACK_CACHE_SIZE default algorithm:**
 - If the **IOEFSPRM** (or parmlib search) does not specify either **meta_cache_size** or **metaback_cache_size**:
 - Let $X = 0.10 \times \text{Real Storage Size (at zFS initialization time)}$
 - If $X < 64\text{M}$ then set $X = 64\text{M}$
 - Else if $X > (2\text{G}+100\text{M})$ then set $X = 2\text{G} + 100\text{M}$
 - If $X \leq 100\text{M}$ then **meta_cache_size=X** AND **metaback_cache_size=0M**
 - Else **meta_cache_size=100M** and **metaback_cache_size=X-100M**
 - Else if the **IOEFSPRM** (or parmlib search) only specified **metaback_cache_size**:
 - Set **meta_cache_size=64M**
 - Else if the **IOEFSPRM** (or parmlib search) only specified **meta_cache_size**:
 - **metaback_cache_size is not created**



Defaults Changed IV – Health Check Details

- **Migration Health Check to assist you on z/OS V1R12 and V1R13 systems:**
 - **ZOSMIGV2R1_ZFS_VERIFY_CACHESIZE**
 - Verifies: `user_cache_size` & the sum of `meta_cache` and the `metaback_cache`.
 - Verifies the cache sizes you are running with against the defaults or
 - Will allow you set a PARM within the check to override the defaults.
 - Inactive, one time check, low severity. Provided in an APAR, marked with **IBM.Function.HealthChecker** FIXCAT.

- **Best practice Health Check to verify and notify of differences on z/OS V2R1 and higher:**
 - **ZFS_VERIFY_CACHESIZE**
 - Exactly the same check as the Migration Health Check, except that the messages are different. (No “migrating to z/OS V2R1” phrase necessary!), and it is Active.
 - This check will help you to know if your cache sizes fall below a desired level (either system default, or some specified value you have determined).

Discontinued Support

- **z/OS 2.1 does not support multi-file system aggregates or cloning or file systems containing cloned (.bak) file systems.**
 - Hence the terms file system and aggregate are now synonymous.

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!



[System z Advocates](#) **
[IBM Mainframe- Unofficial Group](#)
[IBM System z Events](#)
[Mainframe Experts Network](#)
[SHARE](#)



[IBM System z](#) **
[IBM Master the Mainframe Contest](#)
[IBM Destination z](#)
[SHARE Inc.](#)



[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](#)



[IBM System z](#) **
[Destination z](#)



[IBM Mainframe50](#)

Include the hashtag [#mainframe](#) in your social media activity and [#mainframe50](#) in 50th anniversary activity