

 #SHAREorg

# zFS Diagnosis II: Problem Determination and File System Monitoring

Scott Marcotte (smarcott@us.ibm.com)

IBM

August 8, 2012 4:30PM

Session Number 11790



---

## Topics

<b><u>Title</u></b>	<b><u>Slides</u></b>
File System Monitoring	3-4
Auditing	5
Disk Space Monitoring	6-9
File System Backup/Quiesce	10-12
File System Copy	13
Cancel/Force	14
System Outages	15-17
Disk I/O Errors	18
XCF Errors	19-20
Enabling Debugs	21
zFS Software Errors	22-25
System Hangs	26-28
Storage Usage	29-30

## Basic File System Information I: Mode, Owner, Sharing Mode:

- **zfsadm lsaggr** – shows mounted file systems, zFS owner and mount mode

OMVS.ZFS.REL19.DR16	DCEDFBLD R/O	zFS Owner
OMVS.ZFS.PROJ.DFS	DCEDFBLD R/W	Mount mode

- **zfsadm aggrinfo -long** - shows sysplex sharing mode, and mount mode

<p>OMVS.ZFS.PROJ.DFS (R/W COMP): 34447 K free out of total 36000          version 1.4          auditfid 00000000 00000000 0000</p> <p>4284 free 8k blocks;      175 free 1K fragments          360 K log file;            32 K filesystem table          8 K bitmap file</p> <p>ZFSAGGR.BIGZFS.FS1 (R/W COMP): 1737527 K free out of total 2000160          version 1.4          auditfid 00000000 00000000 0000          sysplex-aware</p> <p>217190 free 8k blocks;      7 free 1K fragments          32800 K log file;            56 K filesystem table          288 K bitmap file</p>	<p>zFS Sysplex Sharing State          (blank means not using          zFS sysplex sharing, hence          NORSHARE)</p> <p>Says sysplex-aware which          means it can use sysplex          sharing, hence RWSHARE</p> <p>Not using unique auditfid          (see slide 4)</p>
---	---

**NOTE:** This presentation assumes zFS compatibility mode file systems used exclusively, clones (.bak) file systems also not presented since they are discontinued after z/OS 13,

→ This means the terms file system and aggregate are synonymous.

## Basic File System Information II: System Sysplex Sharing Status

- **One might need to determine if a particular R/W mounted file system is using zFS sysplex sharing or z/OS Unix Sharing on a specific plex member.**
- **F ZFS,QUERY,FILESETS,ALL** – shows file systems locally mounted on that system
  - Need to issue on each plex member to see if it's a zFS sysplex client or not for a given file system. → If a file system not listed, zFS does not have it mounted on that system.

```

15.29.40 DCEIMGHR STC00043 IOEZ00438I Starting Query Command FILESETS
File System Name                Aggr # Flg Operations
-----
ZFSAGGR.BIGZFS.FS1              1  AMS      2
  
```

S – means file system is using zFS sysplex sharing

- **D OMVS,F,T=ZFS** – shows file system information from z/OS Unix view

```

15.42.01 DCEIMGHR      d omvs,f,t=zfs
15.42.01 DCEIMGHR      BPXO045I 15.42.01 DISPLAY OMVS 430
OMVS 000F ACTIVE      OMVS=(P0,HR)
TYPENAME  DEVICE -----STATUS-----  MODE MOUNTED  LATCHES
ZFS       28 ACTIVE      RDWR  05/30/2012  L=39
NAME=ZFSAGGR.BIGZFS.FS1          14.58.37  Q=39
PATH=/home/suimgqh/lfsmounts/PLEX.ZFS.FILESYS
OWNER=DCEIMGHQ AUTOMOVE=Y CLIENT=N
  
```

CLIENT=N  
.. Means zFS sysplex sharing being used or system is z/OS Unix owner

Since owner is HQ and command issued from HR, this is RWSHARE and zFS handling request for this system

- These commands should be issued from each system
- In rare cases, there could be a file system mounted RWSHARE but on some plex member z/OS Unix function shipping is being used
  - → check system log for messages to see why mount failed
- Try an unmount-remount without mode switch to clear this condition

## Auditing

- 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

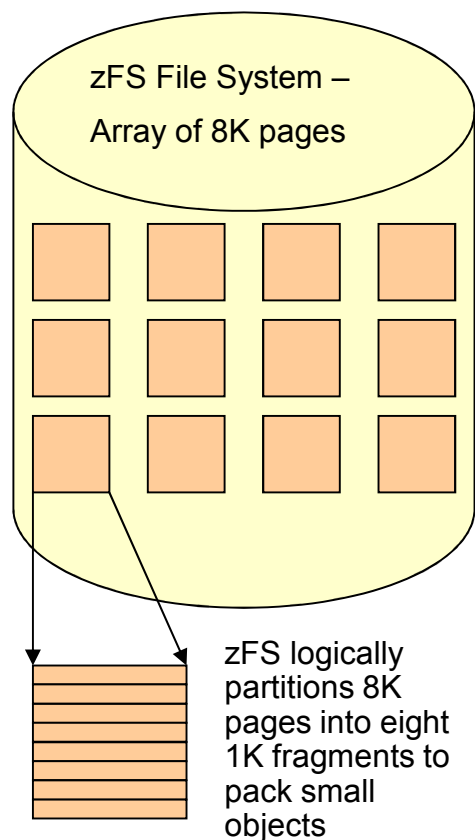
volser	CCHH	inode	Un
--------	------	-------	----

Unique version

- The standard (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](http://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.
- To ensure the unique version is used you can do the following:
  - **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.
  - **zfsadm setauditfid** – This command will store the proper unique audit id inside an existing file system.
  - **convert\_auditfid=on** – zFS startup parameter will auto-convert any mounted file system not using the unique method to begin using it (also: **zfsadm config -convert\_auditfid**).
- **zfsadm aggrinfo -long** – will show the audit fid of the file system (the first 10 bytes of audit id)

## Disk Space Monitoring I: Disk Layout

- zFS file system space monitoring more complicated due to way files stored on disk:



- **File system objects stored 3 ways:**

- **Inline** – Object never grew bigger than 52 bytes, stored inside the object's inode on disk.
- **Fragmented** – Object never grew bigger than 7K, can be put into fragments in a page and stored with other small objects:
  - R13 zFS only uses fragmented method for ACLs and symbolic links, aggressively packing them in the same pages. Prior releases stored small files and directories in same page but was not efficient at it.
- **Blocked** – Any object that grew bigger than 7K (or for R13 any files bigger than 52 bytes and all new directories) is simply a sparse array of pages. Must have empty pages to extend.
- **df** - command not always good to determine space usage, might indicate space free (free fragments) but might not show the fact that there are no free 8K pages.
  - Because of R13 allocation methods and aggressive packing of small objects, df will be more accurate and usable if the file system was created and used with R13 or later.

## Disk Space Monitoring II: Querying Space and Low Space Messages

- **zfsadm agrinfo -long** – shows more detailed space usage:

ZFSAGGR.BIGZFS.FS1 (R/W COMP): 1737527 K free out of total 2000160		Shows number of unallocated whole 8K pages
version 1.4		
auditfid 00000000 00000000 0000		Shows number of free fragments inside 8K pages
sysplex-aware		
217190 free 8k blocks;	7 free 1K fragments	
32800 K log file; 288 K bitmap file	56 K filesystem table	

- **Can monitor low-space via system log messages**

- **aggrfull(threshold, increment)** –Indicates a *threshold* percentage of file system space usage before zFS issues messages to log to warn of high space usage on file system and an *increment* of when zFS should repeat the message as more and more space is used.
  - zFS startup parameter is default for **MOUNT**s that do not specify this parameter
  - Can be over-ridden on **MOUNT** command via the **PARM** option.
- Sample Low-space Messages (units=8K blocks, example from aggrfull(80,5)):
  - IOEZ00078I zFS aggregate ZFSAGGR.BIGZFS.FS2 exceeds 85% full (17083725/20098500)
  - IOEZ00079I zFS aggregate ZFSAGGR.BIGZFS.FS2 is now below 80% full (16078800/20098500)
- If a file system completely runs out of space zFS will issue this message to the log:
  - IOEZ00551I Aggregate ZFSAGGR.BIGZFS.FS2 ran out of space.
  - Will repeat message every 10 minutes until aggregate no longer out of space.

## Disk Space Monitoring III: Dynamic Growth Process

- **Dynamic growth requirements:**
  - **aggrgrow** enabled – either via IOEFSPRM default or specified on MOUNT
  - File system has secondary allocations defined and free space on candidate volume
  - No prior dynamic grow failures for this file system since it was last mounted
- **If requirements met, zFS will:**
  - Issue message such as:
    - IOEZ00312I Dynamic growth of aggregate ZFSAGGR.ZFS.FS2 in progress (by user SUIMGHQ).
  - Grow aggregate by 25MB
    - Will extend aggregate by secondary extents if necessary, issuing message:
      - IOEZ00329I Attempting to grow ZFSAGGR.ZFS.FS2 by a secondary extent.
    - If the extension fails, will issue message:
      - IOEZ00445E – shows DFSMS return codes for extension attempt explained in text for message IEC161I
    - Issue a format-write for at most 25MB
      - IOEZ00326E – shows DFSMS Media Manager return code if the format-write fails.
  - Issue a final message showing success or failure:
    - IOEZ00309I Aggregate ZFSAGGR.ZFS.FS2 successfully dynamically grown (by user SUIMGHQ)
    - IOEZ00308E Aggregate ZFSAGGR.ZFS.FS2 failed dynamic grow (by user SUIMGHQ)
  - File system reading still allowed if read is not to a file/directory that is waiting for the grow to complete.
  - Writes generally made to wait unless they are only updating existing blocks of a file.



## Disk Space Monitoring IV : Wrap-up

- **zfsadm grow** – Can explicitly grow a file system.
  - Same as dynamic grow except user controls how much to grow file system.
- **Low space performance effects:**
  - zFS uses a distributed space reservation scheme
    - Each plex member gets a pool of blocks to use to allocate to files for RWSHARE file systems
    - Owner uses same logic as sysplex clients, has pool for local users that need to add blocks to a file.
    - The more space left in file system, the bigger the pool the owner can hand out to clients
    - **If less than 1MB in space, clients do not get pools**
      - **Must synchronously request blocks as needed if file writes require space**
      - ... which has a non-trivial performance effect**
    - **Dynamic growth is not cheap**
    - Try to minimize these low-space write conditions and minimize dynamic growth in peak usage times when possible.

## File System Backup I: Quiesce/Unquiesce Overview

- A R/W mounted file system should be quiesced before backup.
- A quiesce will do:
  - Stop all write activity sysplex-wide to the file system
  - Sync all dirty data and the zFS log file to disk
    - This makes it look like the file system was cleanly shutdown to zFS
- After a quiesce is complete a backup would be performed.
- Once the backup is complete, an unquiesce should be performed
  - This will allow application write requests to proceed.
- R/O mounted file systems do not require a quiesce prior to backup
  - But no harm in doing it.
  - No user activity is stopped, so should have no negative system affect.
- A quiesced file system can also be un-mounted if desired.
- If the file system owner, or the system initiating the quiesce terminates, the file system will be un-quiesced by zFS.
- **Mounting a restored R/W mounted file system that was not properly quiesced before backup will require log file recovery since it will look like an unclean shutdown to zFS, which delays mount time (worst case @ 1 minute, typical case a few seconds)**

## File System Backup II: ADRDSSU

- Full volume dumps require file systems to be un-mounted before dumping.
- If not using an **ADRDSSU** logical dump for a R/W mounted file system you must do:
  - **IOEZADM quiesce** to stop user activity and sync data to disk
  - Perform backup using your backup program
  - **IOEZADM unquiesce** to resume user activity
- **ADRDSSU** automatically performs a quiesce & unquiesce during a logical backup
  - Can often use flash-copy to keep users quiesced for a short time
- If the backup takes a long time, or there is a problem where the file system did not get unquiesced you will get the following message on operator's console:
  - **IOEZ00581E There are quiesced aggregates.**
    - zFS will check every 30 seconds for quiesced file systems and either issue or delete the message as appropriate.
  - If you see this message persistently on the screen, you may need to intervene.
    - If backup program abended or is stopped, you can issue:
      - ➔ **F ZFS,UNQUIESCE,FSNAME**
      - Or ➔ **zfsadm unquiesce FSNAME**

## File System Backup III: Finding Quiesced File Systems

- The following commands can be used to find which file systems are currently quiesced:
  - **F ZFS,QUERY,FILESETS,ALL**

```
10.43.12 DCEIMGHQ STC00044 IOEZ00438I Starting Query Command FILESETS
File System Name           Aggr # Flg Operations
-----
ZFSAGGR.BIGZFS.FS1        1 AMQSL      43
```

- **zfsadm lsaggr**

```
IOEZ00106I A total of 1 aggregates are attached
ZFSAGGR.BIGZFS.FS1           DCEIMGHQ R/W QUIESCE
```

- **zfsadm aggrinfo**

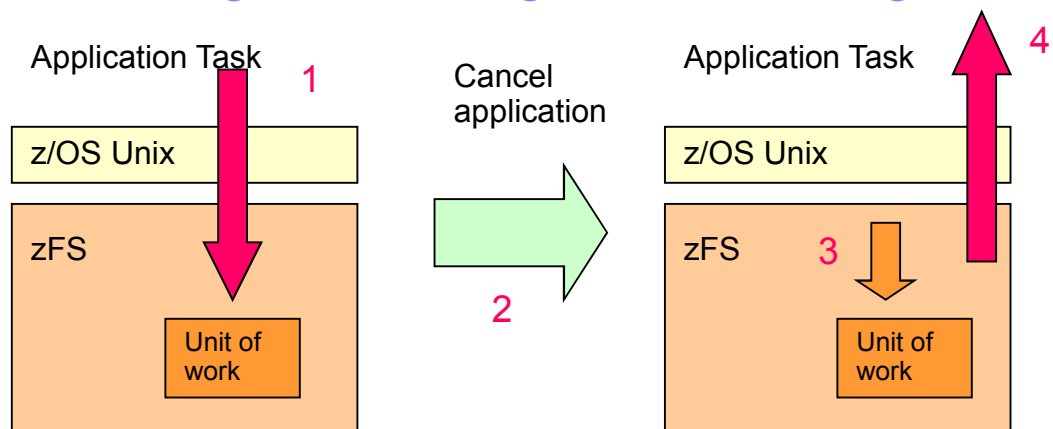
```
IOEZ00369I A total of 1 aggregates are attached to the sysplex.
ZFSAGGR.BIGZFS.FS1 (R/W COMP QUIESCED): 1737527 K free out of total 2000160
```

- **F ZFS,QUERY,FILESETS** often the best
  - Operator command, always available
  - zfsadm may be inaccessible because the file system containing it is quiesced
  - Issue it on the system that shows the message **IOEZ00581E**
- If repeated issuances of the query show the same file system is always quiesced AND the message **IOEZ00581E** is not going away (it is refreshed or deleted every 30 seconds) then:
  - Check to see if backup still running and if not:
    - Issue **zfsadm unquiesce** or **F ZFS,UNQUIESCE** to unquiesce the file system

## File System Copying and Multi-System R/W Access

- zFS uses a scheme to detect multiple R/W access to a file system from multiple systems not in the same sysplex.
  - zFS periodically writes information to first block of file system showing system usage:
    - Includes system name and plex name and timestamp.
  - At un-mount time and system shutdown time, zFS will clear this information to show the file system was cleanly un-mounted and not currently mounted R/W on a system.
  - At mount time, zFS will examine this information to see if another system might have access.
    - If it sees the first block is non-zero
      - Will delay mount for 65 seconds to determine if another system is updating this block.
      - Will issue message **IOEZ00807I** to indicate its waiting
- As a result, you can cause zFS to wait during mount unnecessarily and you can experience z/OS UNIX latch contention if you fail to unmount (detach) a zFS aggregate before copying it or moving it to another system not in the same sysplex.

## Canceling and Forcing Users Running in zFS



1. Application calls for the file system run on the application task
2. User decides to cancel application
3. zFS recovery is notified and switches work to zFS task
4. Application task immediately freed which allows rest of cancel processing to occur more quickly

→ **Unit of work has to be pushed forward or rolled back**

- Application calls into zFS run on the application tasks and there is a unit of work associated with that call.
- When an asynchronous abend, such as a cancel, is delivered to zFS it will dispatch the recovery onto a zFS task which frees up the application task and allows the application to end quickly.
- The unit of work must either be rolled back or pushed forward, it rarely can be ignored.
  - A shared file system such as zFS has common file system objects, an application task can touch global file system structures and other objects that are not its own
  - Pushing forward is common, especially if transactions are started, just as much work to push forward as to roll back
- **Canceling user tasks does NOT break a hang inside zFS if the hang is the fault of zFS.**

## System Outages I: zFS Recovery Fundamentals

- **zFS considers a system down when XCF notifies the group exit for the other members in the sysplex.**
- When zFS is told by XCF that another system is down, zFS knows there are no stray IOs in progress to the disks because:
  - If whole system down, the fencing support in the sysplex software/hardware guarantees no IO in progress from the down member.
  - If only zFS took an outage:
    - If zFS aborts, it will ensure no IOs in progress to disk before leaving its XCF group.
    - If zFS was cancelled, XCF and address space recovery ensures no IO is in progress before it tells other members that zFS went down.
- Because of IO fencing:
  - zFS on other plex members can assume ownership of RWSHARE file systems owned by the down member (z/OS Unix moves ownership for NORWSHARE file systems)
    - zFS members will race for ownership takeover, using GRS serialization to determine who wins race for a specific file system
    - The winner of the race assumes ownership of the file system, confidently performing log file recovery to put the file system in a consistent state

## System Outages II: zFS Recovery Processing

### ▪ **When a system goes down:**

- A zFS member implicitly releases any tokens held by the down system for file systems it owns.
- zFS members implicitly un-quiessces a quiesced file system if the file system was owned or the quiesce was initiated by the down member.
- Will make local calls to file systems owned by the down member wait until someone assumes ownership of the file system.
- Will race for ownership, and if a system wins the race for a specific file system:
  - Will perform log file recovery putting the file system in a consistent state.
  - Allow other plex members to re-obtain tokens for any open files they have.
  - Tell other members to allow their stopped application tasks to resume processing

### ▪ **Application point of view:**

- Applications running on other members are stopped by zFS until ownership of their target file system is assumed by one of the members
- Applications that had open files:
  - May see IO errors reported if zFS on that system thinks data may have been lost in the file; otherwise if zFS is confident of no errors, they will see no errors.
- Due to transaction rollback during log file recovery, newly created objects might not exist, in this case an application could see IO errors reported due to lost object during the crash.
- If no system could assume ownership (should be very rare), stopped applications will be resumed but they will get errors for all calls until some zFS can successfully take ownership of the file system (each zFS member will re-try periodically).



## System Outages III: zFS Messages Issued During a System Outage

**IOEZ00387E System DCEIMGHQ has left group IOEZFS, aggregate recovery in progress.**

IOEZ00388I Aggregate takeover being attempted for aggregate ZFSAGGR.BIGZFS.FS4

**IOEZ00388I Aggregate takeover being attempted for aggregate ZFSAGGR.BIGZFS.FS3**

IOEZ00388I Aggregate takeover being attempted for aggregate ZFSAGGR.BIGZFS.FS1

IOEZ00397I Recovery statistics for ZFSAGGR.BIGZFS.FS4:

elapsed time 2910 ms 72 log pages  
7033 log records, 177 data blocks modified  
6140 redo-data, 6 redo-fill records  
1 undo records, 0 unwritten blocks

IOEZ00044I Aggregate ZFSAGGR.BIGZFS.FS4 attached successfully.

IOEZ00388I Aggregate takeover being attempted for aggregate ZFSAGGR.BIGZFS.FS2

IOEZ00397I Recovery statistics for ZFSAGGR.BIGZFS.FS1:

elapsed time 3735 ms 114 log pages  
10969 log records, 248 data blocks modified  
9640 redo-data, 9 redo-fill records  
3 undo records, 0 unwritten blocks

IOEZ00044I Aggregate ZFSAGGR.BIGZFS.FS1 attached successfully.

**IOEZ00397I Recovery statistics for ZFSAGGR.BIGZFS.FS3:**

**elapsed time 4555 ms 138 log pages**  
**12837 log records, 345 data blocks modified**  
**11342 redo-data, 9 redo-fill records**  
**20 undo records, 0 unwritten blocks**

**IOEZ00044I Aggregate ZFSAGGR.BIGZFS.FS3 attached successfully.**

IOEZ00397I Recovery statistics for ZFSAGGR.BIGZFS.FS2:

elapsed time 5840 ms 308 log pages  
17154 log records, 418 data blocks modified  
15158 redo-data, 9 redo-fill records  
0 undo records, 0 unwritten blocks

IOEZ00044I Aggregate ZFSAGGR.BIGZFS.FS2 attached successfully.

Each zFS member issues message to show when it was told member went down

Any time a zFS member wins race to assume ownership of an RWSHARE file system, it will tell you.

And also tell you when its successfully assumed ownership

Log file recovery often required for file systems that were being updated when owner went down – undo records means transactions rolled back.

Not shown:

- **IOEZ00589E** – system could not open file system to become owner.
- **IOEZ00389E** – Unexpected error prevented system from assuming ownership.

## Disk I/O Errors

- If an IO error occurs when zFS is performing a disk read or write operation it will issue:
  - **IOEZ00001E zFS I/O Error XXX occurred for aggregate aggrname.**
    - XXX is the VSAM error code.
  - **IOEZ00002E MMRE error id=AA cis=BB if=CC of=DD buf=YY CI=ZZ**
    - This second message describes actual disk blocks affected, intended for IBM level-2.
    - zFS has built in limits to avoid a flood of messages to the system log.
- To avoid system hangs for lost channel paths for zFS file systems:
  - Specify in **IECIOSxx parmlib member:**
    - **MIH IOTIMING=00:30,DEV=(XXXX-YYYY)**
      - Sets IO timeout to 30 seconds, for devices XXXX-YYYY
- If IO error occurred reading or writing user file contents
  - Simply issues the above messages, file contents un-cached from memory.
- **If IO error occurred reading or writing metadata**
  - **Will disable access to the file system after issuing message:**
    - **IOEZ00422E Aggregate aggrname disabled.**
- If the file system is disabled:
  - Try and correct disk hardware issue
  - Might require a zFS salvage-repair if the hardware error caused corruptions or a restore from backup
  - Can also try un-mount-remount without mode switch but if corrupted, will become disabled again

## XCF Errors I: Overview

- **XCF error types:**
  - Communication error transmitting or replying to another member
    - This has been extremely rare during the life of zFS sysplex support.
  - More likely is that a system went down/going down just prior to message transmit OR
  - A hang or slowdown that exists on a plex member, these show as timeout errors in zFS.
- **File Operations:**
  - Client to Owner Transmit/Reply Error or Timeout
    - Generally speaking the individual operation will fail.
    - The client will un-cache any data for that object.
  - Owner to client callbacks
    - Owner might have to reclaim tokens held by clients for an object
    - If it cannot communicate to clients, tokens cannot be reclaimed and new accesses to object for all other members fail until communication can be established.
- **Administration Operations:**
  - Administration operations are operations like mount, un-mount, quiesce etc...
  - A lost transmit or reply could leave zFS name-spaces out of sync with each other.
    - In this case zFS will dynamically repair via validation and correction
    - Any new administration operation is made to wait until zFS name space issues corrected due to lost communication packets.
- Hangs and slowdowns appear to other members as a transmit or reply failure
  - Will treat similarly to the above
  - Administration operations have a 15 minute timeout and file operations a 10 minute timeout.

## XCF Errors II: Messages, Validation, Correction

- If an individual file operation has an XCF error or a timeout occurs you will see a message like:
  - **IOEZ00659E Timeout exceeded for the Csvi Create operation on ZFSAGGR.BIGZFS.FS1. A reply was not received from system DCEIMGHQ.**
- If an administration operation like a MOUNT has an XCF error or a timeout occurs you will see a sequence of messages similar to:

```

12.22.13 DCEIMGHR STC00106 *IOEZ00547I zFS has a potentially hanging XCF
request on systems: DCEIMGHQ.
12.22.15 DCEIMGHR STC00106 IOEZ00659E Timeout exceeded for the Connect
operation on
12.22.15 DCEIMGHR STC00106 ZFSAGGR.BIGZFS.FS2. A reply was not received
from system DCEIMGHQ.
12.22.17 DCEIMGHR STC00106 *IOEZ00613I zFS Name Space Validation is
running due to a detected XCF
communication failure or message timeout.
12.22.17 DCEIMGHR      BPXF221I FILE SYSTEM ZFSAGGR.BIGZFS.FS2
FAILED TO MOUNT LOCALLY.
RETURN CODE = 0000007A, REASON CODE = EFF3650C
THE FILE SYSTEM IS ACCESSIBLE ON THIS SYSTEM THROUGH
A MOUNT ON A REMOTE SYSTEM.
12.22.22 DCEIMGHR STC00106 IOEZ00621E There is only one cache entry for aggregate
12.22.22 DCEIMGHR STC00106 ZFSAGGR.BIGZFS.FS2 on system DCEIMGHQ but
this entry indicates there
12.22.22 DCEIMGHR STC00106 are other systems connected.
  
```

zFS hang detector highlights operator message warning of hanging message

zES issues timeout message for operation that timed out.

Because an admin operation timed out sysplex-wide zFS namespace validation is initiated

Mount fails because client dceimgqr could not talk to owner dceimgqr

→ z/OS Unix function shipping now occurs on dceimgqr for this file system, and it does not get benefit of zFS RWSHARE sysplex.

There was a correction performed for the DCEIMGHQ namespace, which ensures namespace consistency in this plex. The actual message text/correction intended for IBM level-2 if needed.

## Enabling Debug and Obtaining Dumps

- If you have a problem, IBM level-2 might ask for debug classes enabled.
  - Can have debugs enabled at startup by specifying dataset in IOEFSPRM:
    - **DEBUG\_SETTINGS\_DSN=SUIMGHQ.PRIVATE.PARMS(ZFS)**
    - Which would contain statements such as this:  
**IOEDEBUG=(C=DIR,P=M,L=127)**  
**IOEDEBUG=(C=CTKC,P=M,L=127)**
  - Can also dynamically enable or disable debug classes like this:
    - **F ZFS,IOED,C=DIR,P=M,L=127** -- this enables the DIR debug class
    - **F ZFS,IOED,C=DIR,P=N,L=0** -- this disabled the DIR debug class
- IBM level-2 might ask for you to increase your trace table size.
  - Can specify the size only in the startup parms
    - **trace\_table\_size=500M**
  - Cannot dynamically re-size the trace table.
  - Default is somewhat small, if possible, maybe make it larger in case you hit a bug.
- IBM level-2 or user might decide that dumps are needed for a situation (such as unexpected failure of a file command or operation):
  - Can obtain one at any time via:
    - **F ZFS,DUMP**
  - This dump command will do the following:
    - Dump all zFS sysplex members
    - Dump both z/OS Unix and zFS storage to allow for debugging of most problems.

## zFS Software Errors I: Overview

- zFS is continually checking for errors on all operations, and periodically:
  - Has thousands of active checks in the code looking for problems that might affect a file or file system, or the entire zFS system.
  - Issues 2C3 abend, with a reason code denoting the error type, if a problem found.
- zFS classifies internal errors in one of 3 classes:
  - **Transient** – No significant harm to the system, might affect only a single operation and/or a single file.
  - **Disabling** – An error has occurred that might affect the integrity of a file system. In this case zFS disables access to the file system in hopes to prevent bad data from being written to disk (this is very effective at preventing corruptions).
    - zFS will attempt an internal re-mount or ownership change in this case up to a maximum of 3 times (>3 assumed a permanent corruption)
  - **Fatal** – zFS has a severe error, or an error in error recovery itself, and will self-restart. With z/OS 13 the restart will be an internal-restart that will preserve the file system mount tree in most cases.
- If a file system is corrupted, or believed to be corrupted:
  - zFS provides a program called **ioeagslv** that will verify and repair an aggregate.
    - Will not scale to larger file systems though (IBM working on a solution for this)
    - If problems are found, will repair them.
    - Requires a re-run if problems are found, it is designed this way – re-run fixing some problems until a run of the program finds no errors (IBM fixing this too)

## zFS Software Errors II: Transient and Disabling Classes

- All errors show a similar initial sequence of messages:

```
IOEZ00337E zFS kernel: non-terminating exception 2C3 occurred, reason
EA150342 abend psw 77C1000 8C20DCD6
IOEZ00064I General Registers R0: 4000000 42C3000 6BCDD000 C117572
IOEZ00064I General Registers R4: C11B068 EA150342 62F67378 EA150342
IOEZ00064I General Registers R8: 7EB1A34C 0 1 0
IOEZ00064I General Registers R12: 6BCE0000 6BCE0C30 8C65ACFA EA150342 ..... (more
register dumps)
IEA045I AN SVC DUMP HAS STARTED AT TIME=16.10.54 DATE=06/01/2012 903
FOR ASID (004A) QUIESCE = YES
IEA794I SVC DUMP HAS CAPTURED: 904
DUMPID=002 REQUESTED BY JOB (ZFS )
DUMP TITLE=zFS abend 02C3 reason EA150342 Jun 1 20:10:44 in module
IOEFSCM at offset 0010DCD6
IOEZ00334I Return code and reason code for dump is 40000000
```

- Disabling errors for RWSHARE file systems in a sysplex also do:

```
IOEZ00422E Aggregate ZFSAGGR.BIGZFS.FS1 disabled
IOEZ00548I Requesting that DCEIMGHR takeover aggregate ZFSAGGR.BIGZFS.FS1
(requests: local 0, new owner 0 total 0)
```

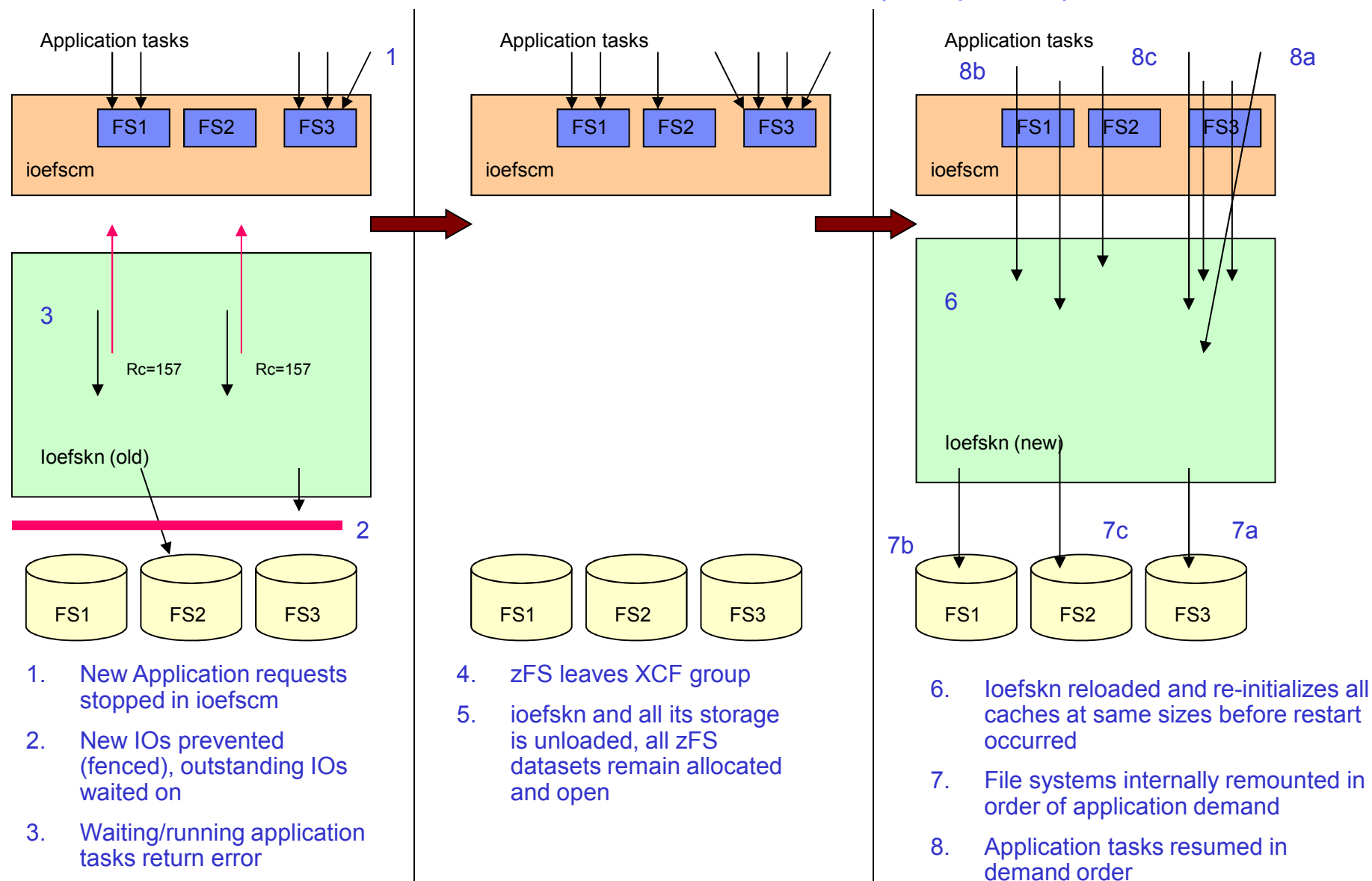
**With RWSHARE, zFS switches ownership to clear condition (fastest method)**

- Disabling errors for NORWSHARE file systems or single-system do:

```
IOEZ00422E Aggregate ZFSAGGR.BIGZFS.FS1 disabled
IOEZ00747I Automatically re-enabling file system ZFSAGGR.BIGZFS.FS1
IOEZ00048I Detaching aggregate ZFSAGGR.BIGZFS.FS1
IOEZ00725I Automatic re-enablement of file system ZFSAGGR.BIGZFS.FS1 complete.
```

**With NORWSHARE or single-system, zFS initiates a remount without mode switch**

## zFS Software Errors III: Internal Restart Process (Simplified)





## Software Errors IV: Internal Restart Example

```

13:52:21 F ZFS,ABORT
13:52:21 IOEZ00338A zFS kernel: restarting exception 2C3 occurred, reason
EA150384 abend psw 77C1000 8C20DCD6
13:52:21 IOEZ00334I Return code and reason code for dump is 00000000
13:52:56 IOEZ00051I An error occurred in program IOEFSKN
13:52:56 IOEZ00357I Successfully left group IOEZFS.
13:52:56 IOEZ00057I zFS kernel program IOEFSKN is ending.
13:52:57 IOEZ00579I Restarting zFS kernel, Restart count 1
13:52:57 IOEZ00559I zFS kernel: Initializing z/OS zSeries File System 703
Version 01.13.00 Service Level OA38177 - HZFS3D0.
Created on Mon Mar 5 13:16:13 EST 2012.
Address space asid x46
13:52:57 IOEZ00178I DCEIMGHQ.PARMLIB(ZFSPARMS) is the configuration dataset
currently in use.
13:52:57 IOEZ00727I Pre-processing 100000 vnodes for file system restart
13:53:32 IOEZ00617I zFS is running sysplex filesys,rwshare with interface level 4
13:53:32 IOEZ00350I Successfully joined group IOEZFS
13:53:32 IOEZ00646I *zFS Kernel is restarting 4 file systems 708
13:53:32 IOEZ00055I zFS kernel: initialization complete.
13:53:33 IOEZ00397I Recovery statistics for ZFSAGGR.BIGZFS.FS3:
elapsed time 352 ms 37 log pages
3527 log records, 40 data blocks modified
3075 redo-data, 0 redo-fill records
2 undo records, 0 unwritten blocks
13:53:33 IOEZ00728I Resuming user operations for file system ZFSAGGR.BIGZFS.FS3
Waking 7 waiting tasks
Re-cached 1435 vnodes for this file system
.....
13:55:32*IOEZ00646I zFS Kernel is restarting 1 file systems
13:55:39 IOEZ00731I zFS internal restart complete.
  
```

1. Fatal error occurred, issue diagnostics, fence IO, stop calls into zFS and wake waiting tasks.
2. zFS unloads and re-loads, indicates restarting.
3. Pre-processes cache of objects that applications were accessing
4. Rest of initialization completes and begin internally re-mounting file systems
5. File system FS3 recovered and application tasks resumed
6. Last of file systems re-mounted and restart complete

## System Hangs I: Overview

- Hangs that are the fault of zFS are rare in the field.
- zFS will often warn the administrator via highlighted operator messages when:
  - A target system taking a long time processing a message from a sending system
    - **IOEZ00547I zFS has a potentially hanging XCF request on systems: DCEIMGHQ.**
  - An external function call seems to be taking a long time
    - **IOEZ00604I Task asid=0059 tcb=6E8448 is delayed outside zFS while opening dataset.**
      - This message used for any unexpectedly long external call to DFSMS, XCF, GRS, waiting for an IO completion etc... by zFS.
  - A zFS task seems to not be making progress, or taking too long, or possibly looping:
    - **IOEZ00605I Task asid=0059 tcb=6E8448 appears to be delayed.**
      - If the task is a zFS task, zFS will restart since we assume zFS has a loop.
- zFS has a hang detection task that will:
  - Internally restart zFS if it finds a confirmed deadlock in the classic sense after first reporting the deadlock and obtaining proper dumps (extremely rare in the field).
  - Issue above warning messages any time it finds something taking a long time.
  - Warn the user if a potential possible hang exists in zFS:
    - **IOEZ00524I zFS has a potentially hanging thread caused by asid=0048 tcb=728440 asid=0068 tcb=6C6388**
      - Will list any tasks that appear to be possible culprits of the hang, it does not list every waiting task, just the potential suspects that may have caused a hang.

## System Hangs II: Diagnosis

- **F ZFS,QUERY,THREADS,OLDEST** (APAR #XXXXXX)
  - Lists the tasks (and system name) that is waiting the longest sysplex-wide
  - If repeated issuance of this command shows same tasks (nothing changing) you have a likely hang

```

19.42.59 DCEIMGHQ      f zfs,query,threads,oldest
19.42.59 DCEIMGHQ STC00158 IOEZ00438I Starting Query Command OLDEST.
Oldest zFS and z/OS UNIX Tasks
-----
STK/Recov  TCB      ASID Stack      Routine      State
-----
System DCEIMGHQ ←
612E8820 005FF188 004A 611843E8 ZFSRDDIR     WAITLOCK
612A4000
  since Jun 4 23:42:59 2012 Current DSA: 61185258
  wait code location offset=04E2 rtn=get_evp_locked
  lock=71FA79F0 state=E2F82001 owner=(62F82000 0077 5CEBF8)
  lock description=Vnode-cache access lock
  Operation counted for OEVFS=667A4150 VOLP=63162950

fs=ZFSAGGR.BIGZFS.FS3 ←

```

Shows system name, time it started waiting and file system its using

- zFS uses GRS latches and will ENQUEUE GRS resources for certain administration operations (such as MOUNT, QUIESCE, performance based aggregate movement),
  - **D GRS,C** – will include zFS ENQUEUEs and latches
    - Look for EXCL access on **IOEZJOIN** and/OR **IOEZNS** resources
      - Means a system is delayed joining or leaving sysplex or administration commands timed-out and a namespace validation is pending
- If message **IOEZ00604I** is shown, look at system log for problems with the catalog, DFSMS, XCF, GRS, or problems communicating with a sysplex member based on the message text.

## System Hangs III: Resolution

- **If the hang is due to a component that zFS calls due to message IOEZ00604I**
  - → Refer to that components documentation for a resolution procedure
    - ... fixing zFS is not going to fix the external component
    - Canceling the listed tasks/jobs may break the hang in some cases, depending on the component, unless the listed task is a zFS task, then you have to use hangbreak.
- **But if the hang looks to be in zFS itself then the recommended procedure is:**
  - Use **F ZFS,QUERY,THREADS,OLDEST** to determine oldest hanging system
  - Issue **F ZFS,HANGBREAK** on that plex member (or single-system)
    - → This will cause an internal restart after ensuring proper documentation obtained, this should get dumps and solve the problem.
    - (If for some reason an internal restart does not work or it gets hung (this should be rare), then perform a system IPL after obtaining a dump of the zFS and z/OS Unix address spaces)
  - Remember that canceling user tasks/jobs will normally not break a zFS hang
  - When the internal restart is complete, check to see that hang is resolved, if not then repeat these steps with next oldest system.
- **Can initiate a sysplex-wide internal restart:**
  - **F ZFS,ABORT,ALL**
    - → This will do an internal restart on all plex members which ensures every zFS starts with fresh storage.
    - Generally an approach of last-resort.

## Storage Shortages I: Overview

- zFS does cache in-memory copies of disk data in dataspace, but all of its data structures are in its primary address space.
- zFS keeps track of how much storage it is using and queries the operating system to determine how much storage it has available in its address space.
- **zFS has the following storage thresholds:**
  - **USS/External Limit: 60M below the address space limit**
    - If exceeded zFS will fail any operation that attempts to allocate memory for an object that is not already cached in memory  
→ Applications will see ENOMEM failures.
    - zFS issues the following highlighted operator message:  
**IOEZ00662I zFS is low on storage.**
  - **Non-critical Limit: 20M below the address space limit**
    - If exceeded only requests for critical storage allowed
    - Very dangerous, zFS could very well run out of memory  
→ Applications even more likely to see ENOMEM and other failures.
    - zFS issues the following highlighted operator message:  
**IOEZ00663I zFS is critically low on storage.**
  - **Address space limit (as determined by z/OS):**
    - If exceeded, zFS will restart.
    - Restart will be:
      - Internal restart, if most of storage not held by z/OS Unix
      - External restart, if a large portion of storage held by z/OS Unix access of zFS objects.

## Storage Shortages II: Diagnosis and Actions

- **F ZFS,QUERY,STORAGE** – shows zFS storage use

```
IOEZ00438I Starting Query Command STORAGE. 778
zFS Primary Address Space Storage Usage
-----

Total Storage Available to zFS: 1738539008 (1697792K) (1658M)
Non-critical Storage Limit: 1717567488 (1677312K) (1638M)
USS/External Storage Access Limit: 1675624448 (1636352K) (1598M)
Total Bytes Allocated (Stack+Heap+OS): 1669189632 (1630068K) (1591M)
Heap Bytes Allocated: 1587033610 (1549837K) (1513M)
Heap Pieces Allocated: 11445446
Heap Allocation Requests: 4
Heap Free Requests: 3
```

Address space limit determined by VSM

Non-critical limit

USS/External limit

Storage currently allocated

- **If storage dangerously close or exceeds USS/External Storage Access Limit:**
  - Try reducing cache sizes (**meta\_cache\_size/metaback\_cache\_size/user\_cache\_size**) to free some storage via **zfsadm config**.
  - If cache reduction does not solve problem, report it to IBM service.
- **zFS should be run with a region size of 0M**

## Publications of Interest

- z/OS UNIX System Services Planning (GA22-7800)  
[General Administration of z/OS UNIX file systems](#)
- z/OS Distributed File Service zSeries File System Administration (SC24-5989)  
[zFS Concepts and zfsadm command for zFS](#)
- z/OS Distributed File Services Messages and Codes (SC24-5917)  
[IOEZxxxxt messages and X'EFxxxxrrrr' reason codes for zFS](#)
- z/OS MVS System Commands (SA22-7627)  
[Describes D OMVS command](#)



← QR Code