Introduction to zSeries® File System (zFS)

Scott Marcotte

smarcott@us.ibm.com

April 24, 2007

Trademarks

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

- **DFS**
- **DFSMS**
- **DFSMSdss**
- **IBM**
- **MVS**
- **RACF**
- **RMF**
- **S/390**
- **z/OS**
- **zSeries**

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
UNIX is a registered trademark of The Open Group in the United States and other countries.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

* 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.

Agenda

- **Overview**
- **Usage**
- **Why zFS?**
- **Release Enhancements**
- **zFS Issues**
- **Future Trends**
- **Possibilities for the Future**

Data Set Type

- **HFS data set**
  - DD with DSNTYPE=HFS

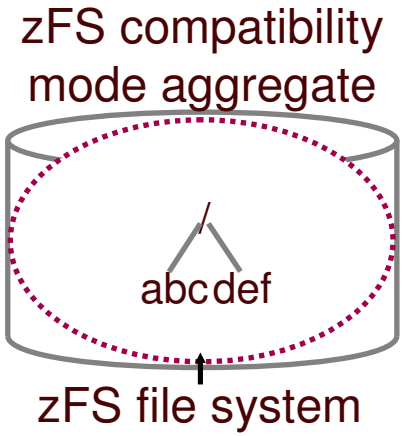
```
//USERIDA JOB
//STEP1   EXEC PGM=IEFBR14
//HFS1 DD DSN=OMVS.HFS.HOME,
//        SPACE=(CYL,(40,1,1)),
//        DSNTYPE=HFS,
//        DISP=(NEW,CATLG,DELETE),
//        STORCLAS=STANDARD
```

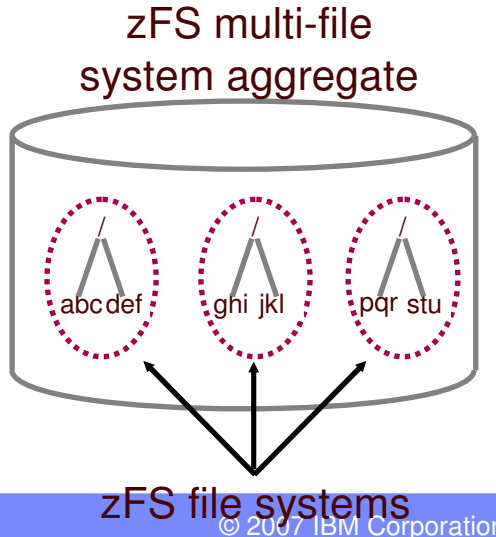
- **zFS data set**
  - define VSAM Linear Data Set (LDS)
  - format with IOEAGFMT
  - (Can also use zfsadm commands to define and format or pfsctl APIs)

```
//USERIDA  JOB
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=H
//SYSIN    DD *
 DEFINE CLUSTER (NAME(OMVS.ZFS1) -
        VOLUMES(PRV000) -
        LINEAR -
        CYL(40 1) -
        SHAREOPTIONS(3))
/*
//STEP2    EXEC PGM=IOEAGFMT,REGION=0M,
// PARM=('-aggregate OMVS.ZFS1 -compat')
//SYSPRINT DD SYSOUT=H
```

Data Set Type …

- **zFS data set (VSAM LDS) is called a zFS aggregate**
  - 2 types of aggregates:
    - Compatibility mode aggregate
      - One R/W file system per aggregate
      - R/W file system name matches LDS name
    - Multi-file system aggregate (multiple zFS file systems per aggregate)
      SUPPORT FOR THESE WILL BE REMOVED IN THE FUTURE – DO NOT USE THESE





zFS Usage

- **zFS is generally transparent to applications (looks the same as HFS)**
- **zFS administration is different than HFS administration**
- **HFS and zFS can be active at the same time and HFS and zFS file systems can be mixed in the same z/OS UNIX hierarchy**
- **zFS supports current z/OS UNIX file system sysplex (Shared HFS) environment only when using compatibility mode aggregates**
  - Since the term “Shared HFS” can be confusing, we are changing to use the term “Shared file system”

Why zFS? – Advantages of zFS over HFS

- **zFS generally provides improved performance over HFS:**
  - Substantial reduction in synchronous IO waits due to better caching and IO algorithms:
    - Fsync operation only syncs individual file, not entire file system
    - zFS aggressively writes data to keep DASD utilized, file system syncs are less disruptive.
  - Better usage of multi-volume file systems, zFS spreads IO amongst all volumes.
  - Reduced Lock Contention
    - Less locking of global file system structures
    - Parallel writes to same file
    - No per-file system locks, much more granularity which results in greater scale-ability.

Why zFS? – Advantages of zFS over HFS

- **zFS provides additional function:**
  - Cloning – Allows for a snap-shot of a file system at a given time, see later slide.
  - Multi-level security
  - ACL space saving techniques
- **zFS provides better error recovery for disk data:**
  - zFS uses logging which ensures disk always made consistent (via log file recovery) after a crash once system is restarted.
  - HFS can permanently lose file systems in rare situations: if a crash occurs at sync time. File system must be restored from backups.
  - zFS provides a salvage program (similar to fsck) to correct software or hardware errors that corrupt disk block contents, brings file system to a consistent state. (This is rarely required, but provides a safety net for you that HFS does not provide).
  - Aggressive writing and logging results in less data loss if a crash occurs.

Why zFS? - Performance

- **Following example has Windows clients continuously reading/writing to z/OS DFS/SMB server.**
- **HFS hits bottleneck at 44 workstations.**
- **zFS continues to scale, results in 62% improvement in measured throughput and even higher potential throughput.**

SMB Clients	HFS	ZFS
44 Clients ETR	274	327
ITR	328	344
60 Clients ETR	256	531
ITR	612	675

Netbench Workload: 9672-X87, Microsoft Windows NT Clients, Thruput in megabits/sec

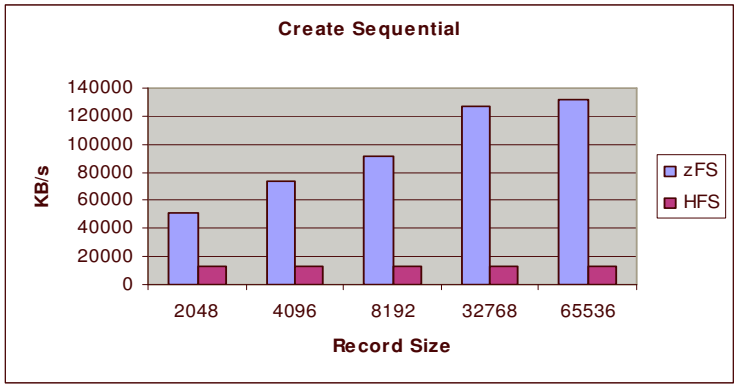
Why zFS – Improved Performance for files over 1M in size

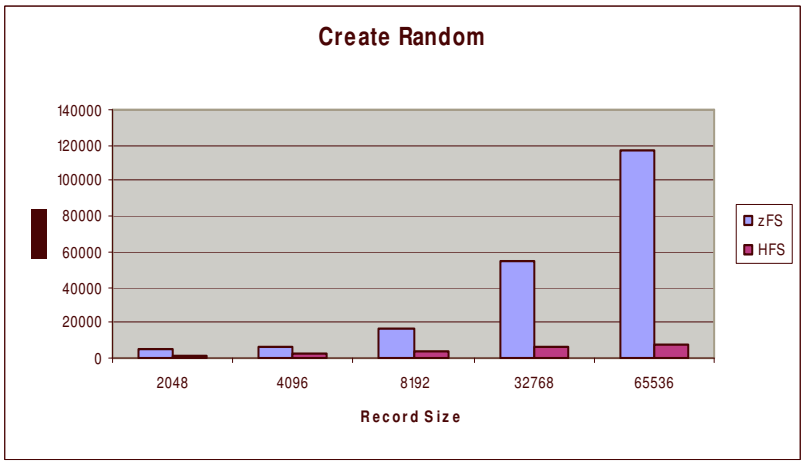
- **Simple database read/write simulation (FSPT) with zFS:**
  - Provides improvements of 1.1X (tiny zFS cache) to 60X (large zFS cache) for one thread reading/writing database file using zFS instead of HFS to store database file.
  - Provides improvement of 1.07X (tiny cache) to 90X (large cache) with 50 threads reading and writing same file.
- **Uncached sequential reads 60% faster with zFS.**
- **Cached sequential reads up to 36X faster with zFS.**
- **Large sequential file creates up to 10X faster with zFS.**
- **Random file creations up to 14X faster with zFS.**

Why zFS? – Performance

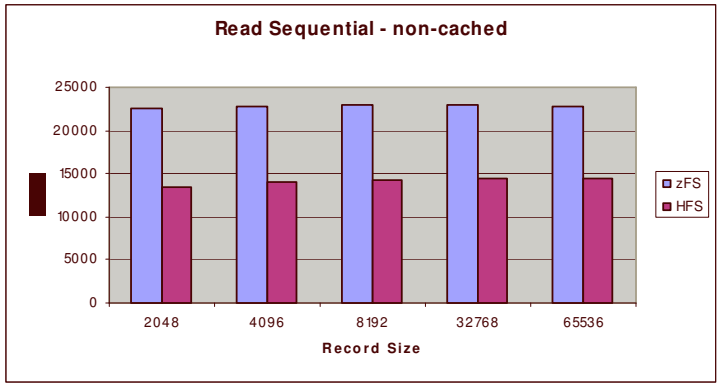
- **HFS**
  - Caches at most 1 MB of each file
  - Avoids writes to disk until sync interval
- **zFS**
  - Caches files regardless of size in an LRU cache
  - Begins asynchronous write to disk when block full

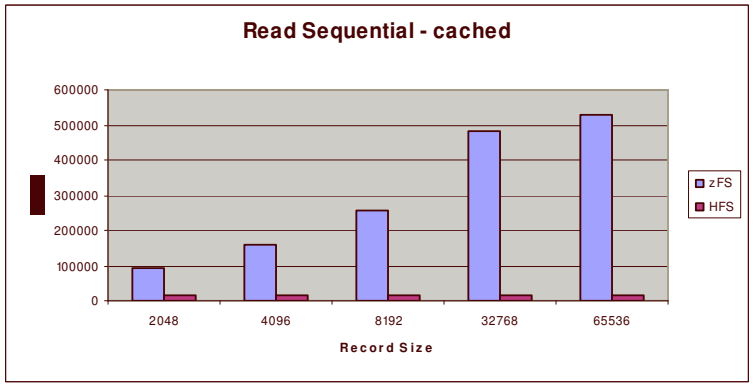
The processor is a z900, 2064-104 (4 processors) with Shark DASD.

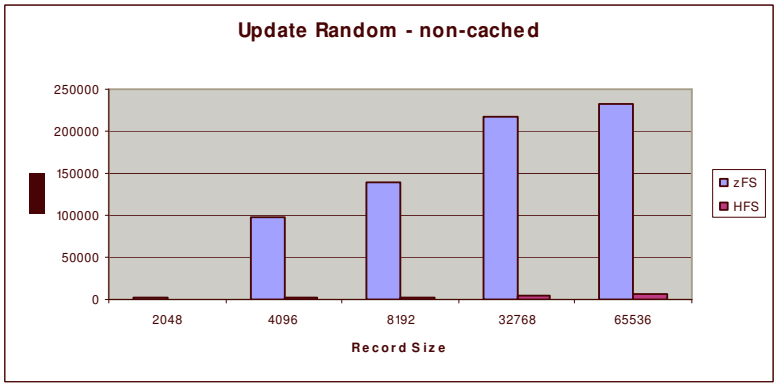


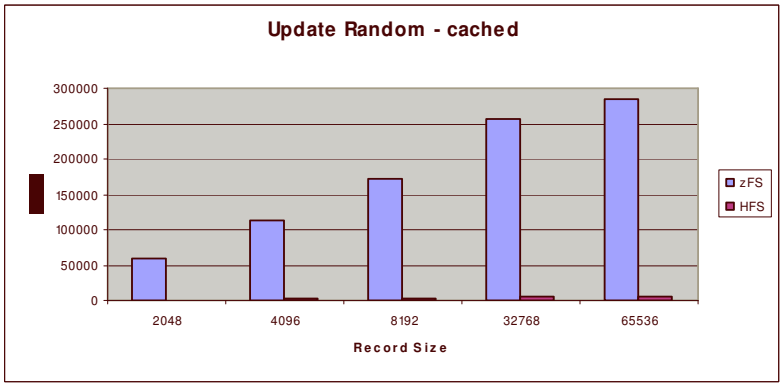


Why zFS? – Performance…



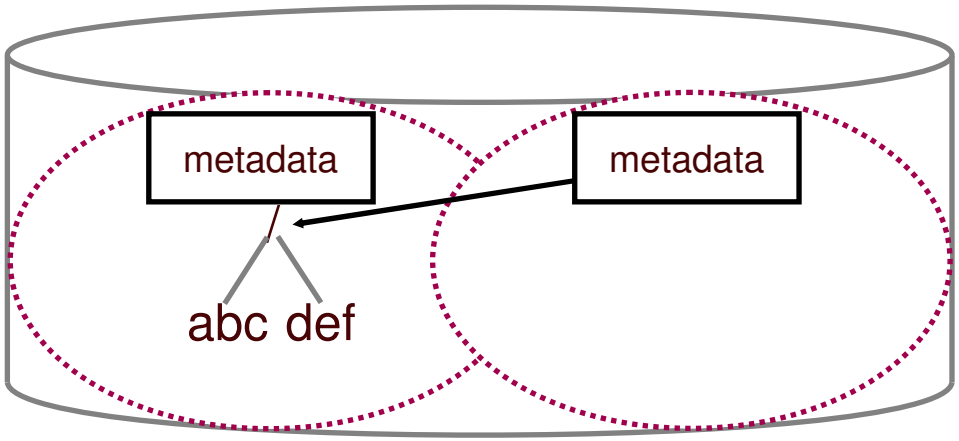






Why zFS? – zFS cloning

- **A clone operation makes a point-in-time read/only backup of a zFS file system in the same aggregate**
- **Only metadata is copied, not user data**
- **The backup file system has the name of the read/write file system with ".bak" appended**
- **Later, when a block of a file of the read/write file system is modified, a new page is allocated to maintain the backup copy of that block (called copy-on-write or COW)**



Why zFS? – zFS logging

- **zFS is a logging file system**
  - After a system failure, the next mount replays the log and brings the file system into a consistent state
  - fsck-like utility called IOEAGSLV (Salvager) in case recovery fails
  - R/O mount can fail if the log needs to be replayed
    - Must be mounted R/W, unmounted, then mounted R/O (last two steps could be remount) – fixing in future releases.

Why zFS? – HFS is stabilized

- **In February 2004, IBM announced**
  - *zSeries file system (zFS):* zFS is the strategic UNIX Systems Services file system for z/OS. The Hierarchical File System (HFS) functionality has been stabilized. HFS is expected to continue shipping as part of the operating system and will be supported in accordance with the terms of a customer's applicable support agreement. IBM intends to continue enhancing zFS functionality, including RAS and performance capabilities, in future z/OS releases. All requirements for UNIX file services are expected to be addressed in the context of zFS only.

zFS enhancements

- **z/OS V1.3**
  - Access Control Lists (ACLs)
  - Can put mount statements in BPXPRMxx
- **z/OS V1.4**
  - Dynamic extension
  - Dynamic configuration
  - Automatic quiesce/unquiesce on ADRDSSU backup (must be issued from owning system – restriction removed in z/OS V1.7)
- **z/OS V1.5**
  - Multi-level security (SECLABELs)
  - Customers can unload file systems shipped with ServerPac into zFS (except for root)

zFS enhancements …

- **z/OS 1.6**
  - RAS enhancements
    - When an internal problem is detected by zFS (Abend X'2C3'), rather than going down, zFS isolates the problem to the aggregate and marks it disabled for writing
    - The zFS Physical File System remains active and other aggregates remain active
    - A disabled aggregate can be made accessible by unmounting and mounting it
    - For safety, you can run IOEAGSLV (salvage) before re-mounting it
  - Parmlib search:
    - Specified in FILESYSTYPE in BPXPRMxx
    - Used when IOEZPRM DD not specifed in ZFS PROC
  - z/OS 1.6 zFS function rolled back to z/OS 1.4 and 1.5 via APAR OA11602

zFS enhancements …

- **In August 2004, IBM announced**
  - Earlier this year, IBM announced that Hierarchical File System (HFS) function is stabilized. The zSeries File System (zFS) is the strategic UNIX Systems Services file system for z/OS. IBM plans to enhance zFS function in z/OS V1.7 so that you can use zFS file systems at all levels within the file hierarchy.

zFS enhancements …

- **z/OS 1.7**
  - zfsadm commands work across the sysplex
    - You can issue a zfsadm command against any aggregate/file system from any member of the sysplex (does not need to be the owning system)
    - You can issue zFS pfsctl APIs from any member of the sysplex
    - Consequently, you can backup an aggregate from any system using DFSMSdss™ (ADRDSSU)
  - zFS aggregate/file names can now include @, #, $
    - Toleration APAR OA08611 for prior releases
  - Performance monitoring commands and APIs (exploited by RMF)

zFS enhancements …

- **z/OS 1.7 continued…**
  - Unquiesce Modify operator command
  - Recovery on End of Memory (cancel/force support)
  - Sysplex root can be zFS file system
  - Mount performance improvement
    - Toleration APAR OA11573 is required in prior releases

zFS enhancements …

- **Multi-file system aggregates are being deprecated**
- **In February 2005, IBM announced**
  - z/OS V1.7 is planned to be the last release to allow mounting zSeries File System (zFS) file systems contained in multi-file system aggregates that are to be shared across systems in a sysplex. IBM has previously recommended that these multi-file system aggregates not be shared in a sysplex environment. Once this support has been removed, attempts to mount zFS file systems contained in multi-file system aggregates will fail in a z/OS UNIX System Services shared file system environment. Mounting zFS compatibility mode aggregates, which have a single file system per data set, will continue to be supported in all environments.
- **and**
  - In a future release, IBM plans to withdraw support for zFS multi-file system aggregates. When this support is withdrawn, only zFS compatibility mode aggregates will be supported. (A zFS compatibility mode aggregate has a single file system per data set.)

zFS enhancements …

- **z/OS 1.8**
  - Cannot mount a file system contained in a zFS multi-file system aggregate in a shared file system environment
    - If you are using multi-file system aggregates, you must copy each file system into a separate zFS compatibility mode file system using z/OS V1R7 or earlier
    - You should plan to complete this copy before you IPL z/OS 1.8
  - z/OS UNIX bpxmtext command can be used to display the text and action for zFS reason codes (EFxxnnnn)
  - The stop zfs command is removed – use modify omvs,stoppfs=zfs

zFS Issues

- **Large directories:**
  - zFS currently stores directories in a linear format (HFS uses B+ trees). This can result in long lookup/create/remove times.
    - Workaround: define large directory_cache_size and apply recent service.
  - IBM is working on a solution to this that will ultimately make zFS perform better than HFS in this area.
- **Critical Errors in zFS:**
  - If error while in error handling or very severe error, zFS will restart which loses mount-tree.
  - IBM is working on a solution to preserve mount tree and auto-correct the error resulting in total availability.

zFS Trends

- **zFS will continue to enhance RAS**
- **zFS will improve sysplex support**
  - File access performance
- **zFS will provide additional performance and scalability improvements**

Publications

- **z/OS UNIX System Services Planning (GA22-7800)**
  General Administration of z/OS UNIX file systems
- **z/OS UNIX Command Reference (SA22-7802)**
  confighfs command for HFS
- **z/OS MVS System Messages Volume 9 (IGF-IWM) (SA22-7639)**
  IGWxxxt messages for HFS
- **z/OS UNIX System Services Messages and Codes (SA22-7807)**
  z/OS UNIX return codes, z/OS UNIX reason codes, X'5Bxxrrrr' reason codes for HFS
- **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)**
  IOEZxxxt messages and X'EFxxrrrr' reason codes for zFS
- **z/OS Distributed File Service zSeries File System Implementation (SG24-6580)**
  - Redbook available (updated February 2006 to include z/OS V1R7)
  - http://www.redbooks.ibm.com/abstracts/sg246580.html?Open
- **z/OS Version 1 Release 8 Implementation (SG24-7265)**
  - Redbook available (contains zFS updates for z/OS V1R8)
  - http://www.redbooks.ibm.com/abstracts/sg247265.html?Open
- **z/OS DFSMS™ Access Method Services for Catalogs (SC26-7394)**
  IDCAMS utility

Possible Enhancements for ZFS

- **This section describes features that are available on other platforms, and other possible enhancements to the zFS product. IBM is not committing to the delivery of any of these features; rather, IBM is trying to determine if there is customer interest in these features.**

Possible Enhancements -

- **Super Large File Systems – File systems whose size far exceeds 16TB, and all of the support that would go with it including:**
  - Salvage while mounted, due to large size of file system.
  - Fast format to ensure a large file system can be created quickly.
  - Possibly allow use of storage pools, file systems created in pools (an improved multi-file system aggregate). Pools could be enormous in size, file systems created quickly and easily. Reduces space usage on DASD.

Possible Enhancements …

- **Data Availability:**
  - Auto-correct disabled aggregates:
    - Ping DASD if it went offline (loss of channel paths) or had error, when back online allow full read/write.
    - Immediately correct software errors with internal mount refresh, possibly with salvage while in use.
  - File system software DASD mirroring (as opposed to hardware mirroring, detects hardware failures):
    - File system checksums all data
    - Provides ability of backup (multiples) of any disk block
    - File system detects bad block and picks up and uses a copy.

Possible Enhancements

- **HSM for files:**
  - Migrate/recall on a file basis.
  - Frees space for little used files.
- **More application control (APIs):**
  - Allow application to read-ahead
  - Allow application to bypass file system cache
  - Allow application to pre-allocate file blocks
  - And other interfaces for more function and better performance, features specific to customer environments.

Possible Enhancements

- **Data Encryption in the file system (rather than, or in addition to encryption in the DASD control unit)**
- **Caching Improvements**
  - Integrate zFS with BCP to dynamically size caches based on system storage usage, paging and zFS cache hit ratios.
- **Cloning Improvements:**
  - Clone while file activity allowed
  - Faster clone and unclone processing
  - Multiple backup file systems.
  - Automatic snapshots and end-user ability to access backup data.
- **Extended Attributes**