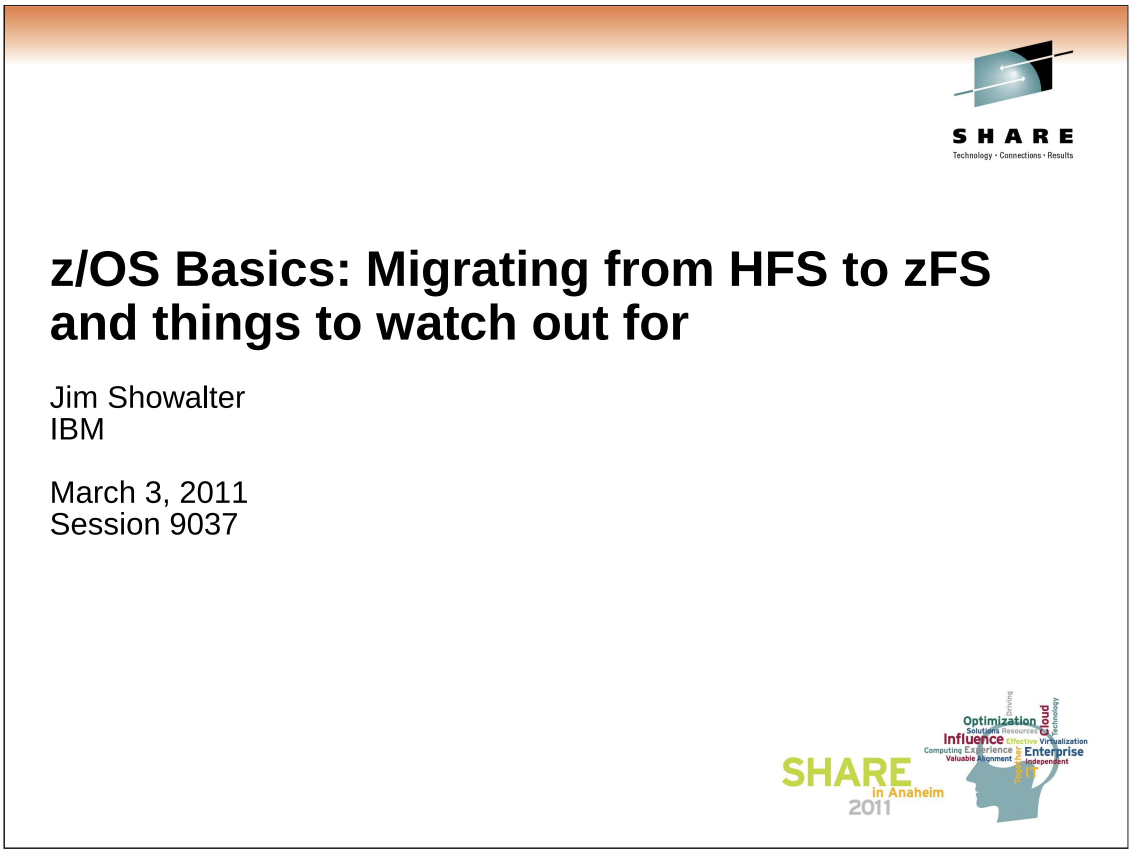
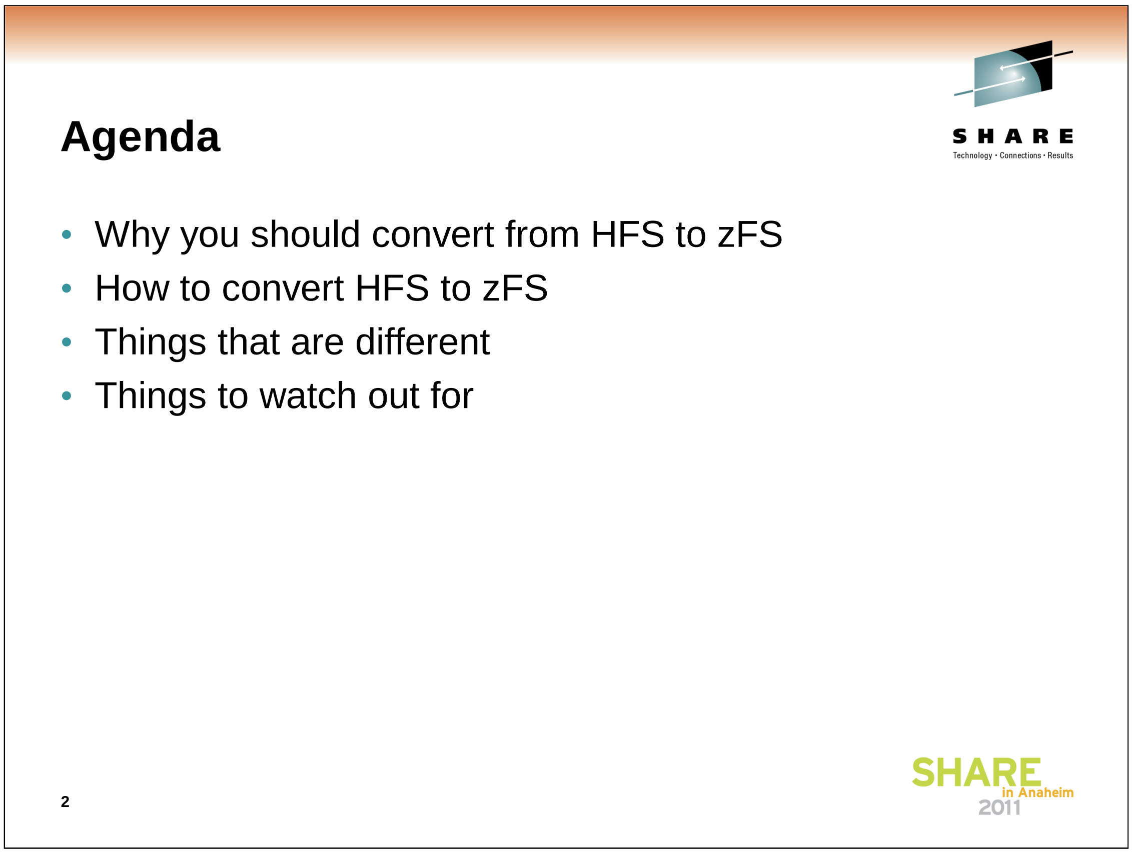


z/OS Basics: Migrating from HFS to zFS and things to watch out for

Jim Showalter
IBM

March 3, 2011
Session 9037

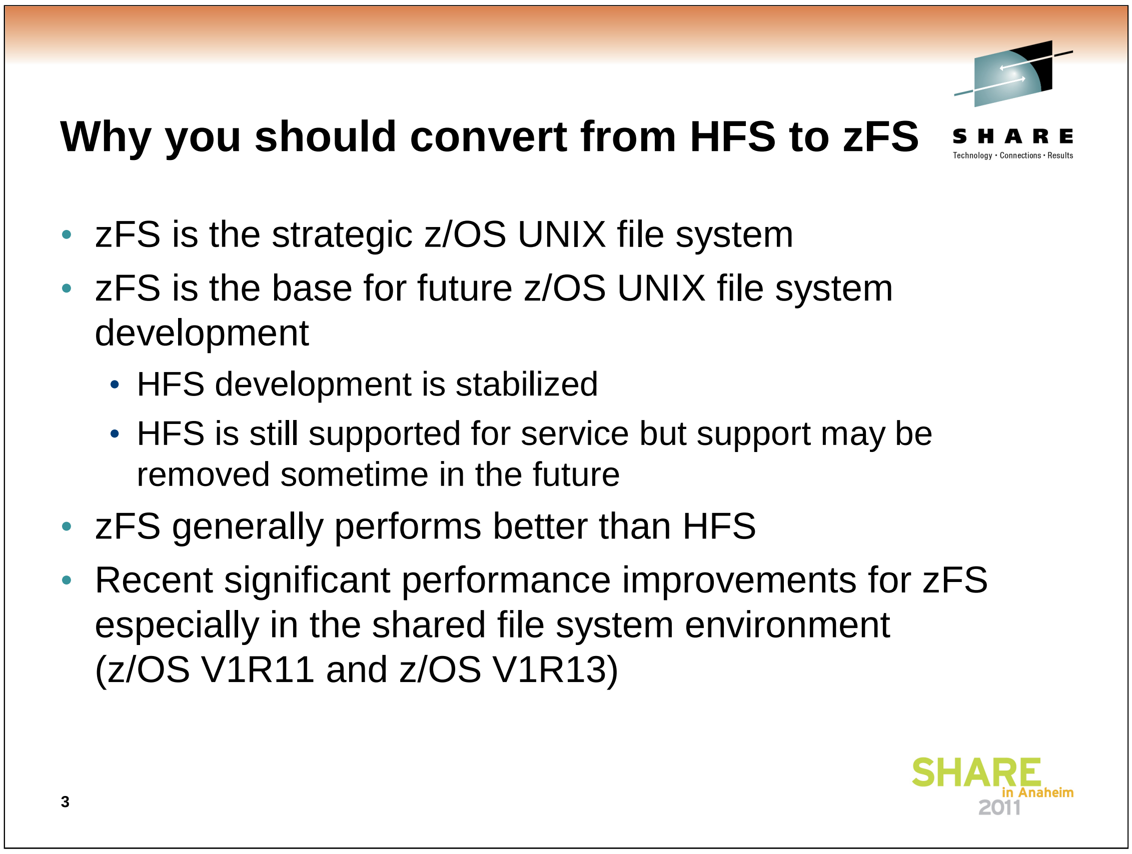


Agenda

- Why you should convert from HFS to zFS
- How to convert HFS to zFS
- Things that are different
- Things to watch out for

2

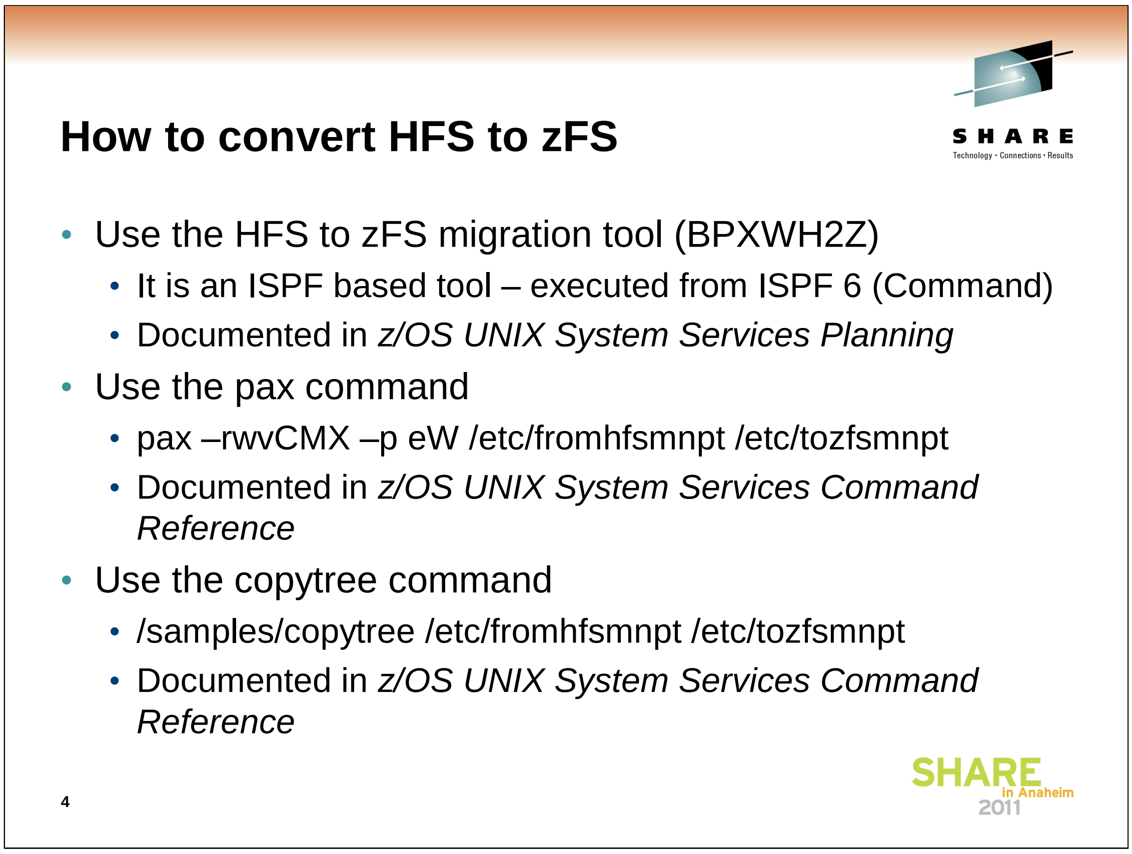
SHARE in Anaheim 2011



Why you should convert from HFS to zFS

- zFS is the strategic z/OS UNIX file system
- zFS is the base for future z/OS UNIX file system development
  - HFS development is stabilized
  - HFS is still supported for service but support may be removed sometime in the future
- zFS generally performs better than HFS
- Recent significant performance improvements for zFS especially in the shared file system environment (z/OS V1R11 and z/OS V1R13)

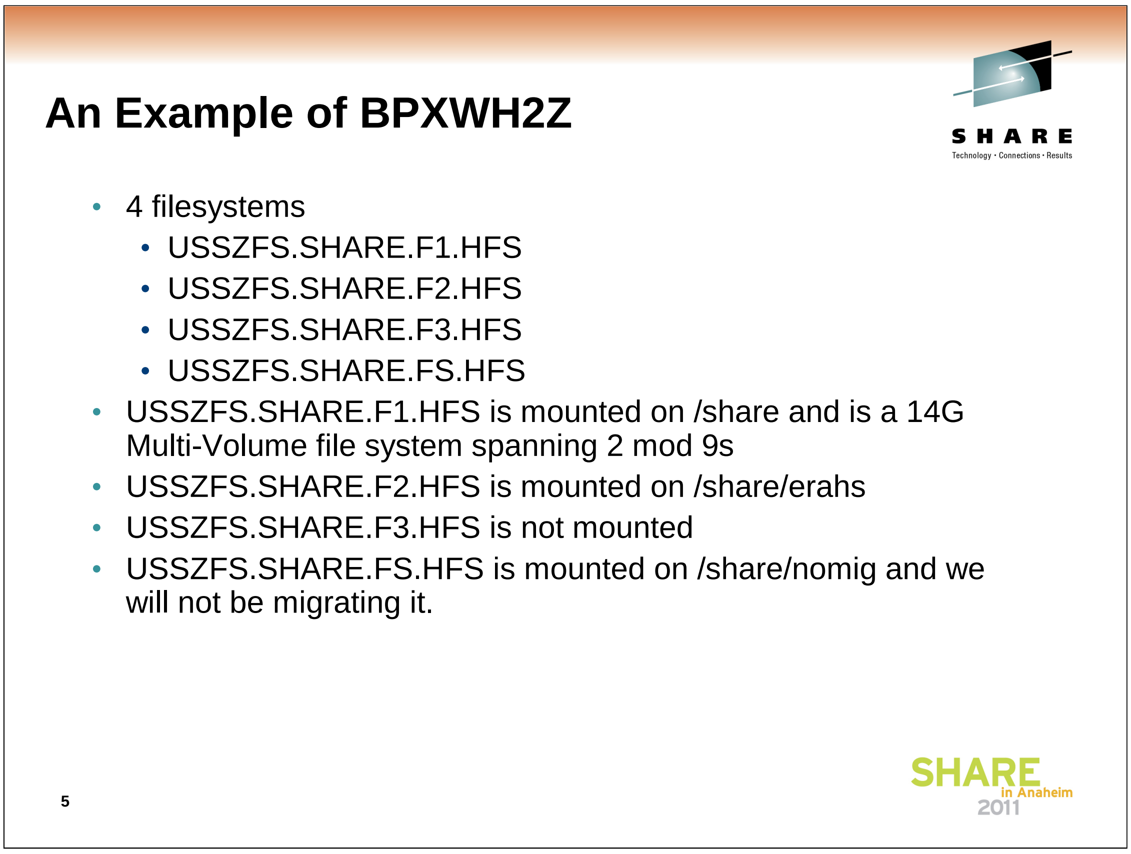
3



How to convert HFS to zFS

- Use the HFS to zFS migration tool (BPXWH2Z)
  - It is an ISPF based tool – executed from ISPF 6 (Command)
  - Documented in *z/OS UNIX System Services Planning*
- Use the pax command
  - pax –rwvCMX –p eW /etc/fromhfsmnpt /etc/tozfsmnpt
  - Documented in *z/OS UNIX System Services Command Reference*
- Use the copytree command
  - /samples/copytree /etc/fromhfsmnpt /etc/tozfsmnpt
  - Documented in *z/OS UNIX System Services Command Reference*

4



An Example of BPXWH2Z

- 4 filesystems
  - USSZFS.SHARE.F1.HFS
  - USSZFS.SHARE.F2.HFS
  - USSZFS.SHARE.F3.HFS
  - USSZFS.SHARE.FS.HFS
- USSZFS.SHARE.F1.HFS is mounted on /share and is a 14G Multi-Volume file system spanning 2 mod 9s
- USSZFS.SHARE.F2.HFS is mounted on /share/erahs
- USSZFS.SHARE.F3.HFS is not mounted
- USSZFS.SHARE.FS.HFS is mounted on /share/nomig and we will not be migrating it.

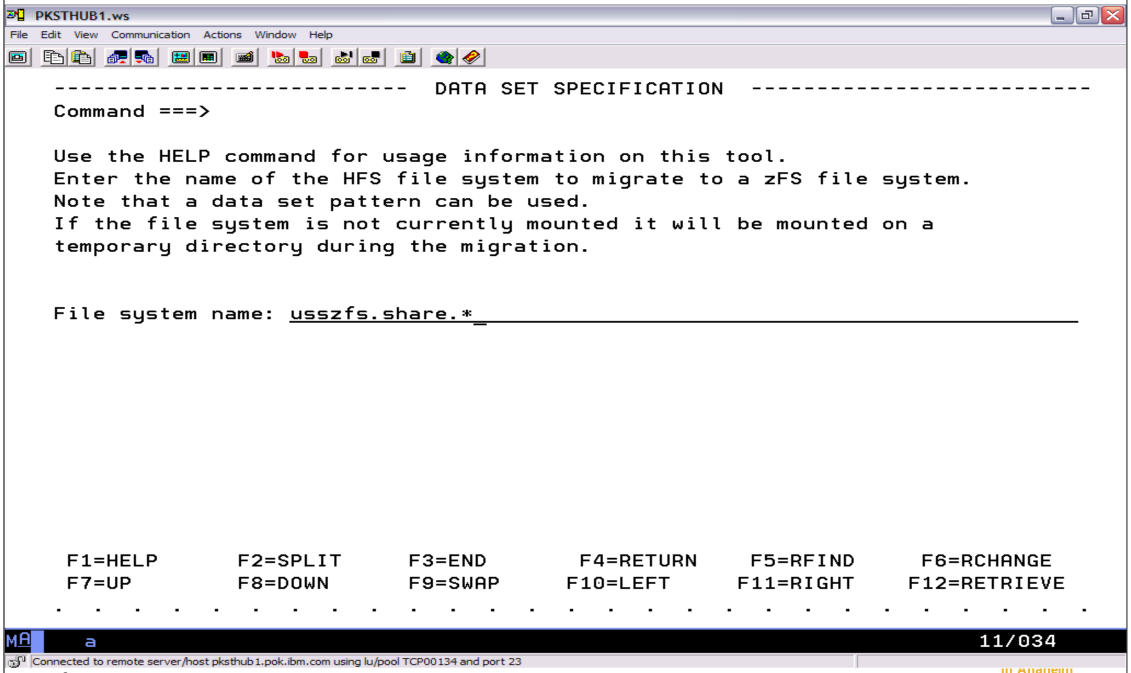
5

SHARE in Anaheim 2011

I used this example because it shows a migration of filesystem in many different configurations. This includes filesystems mounted below other filesystems that are being migrated as well as the migration of filesystems that at not mounted at all. This is done in an effort to show the flexibility of the tool.

I also used this example so that we could discuss large multi-volume filesystems.

Start the tool by issuing BPXWH2Z –cv from TSO



```
 --------------------------  DATA SET SPECIFICATION  -------------------------
 Command ===>

 Use the HELP command for usage information on this tool.
 Enter the name of the HFS file system to migrate to a zFS file system.
 Note that a data set pattern can be used.
 If the file system is not currently mounted it will be mounted on a
 temporary directory during the migration.


 File system name: usszfs.share.*_






  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
MA      a                                                            11/034
Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
```

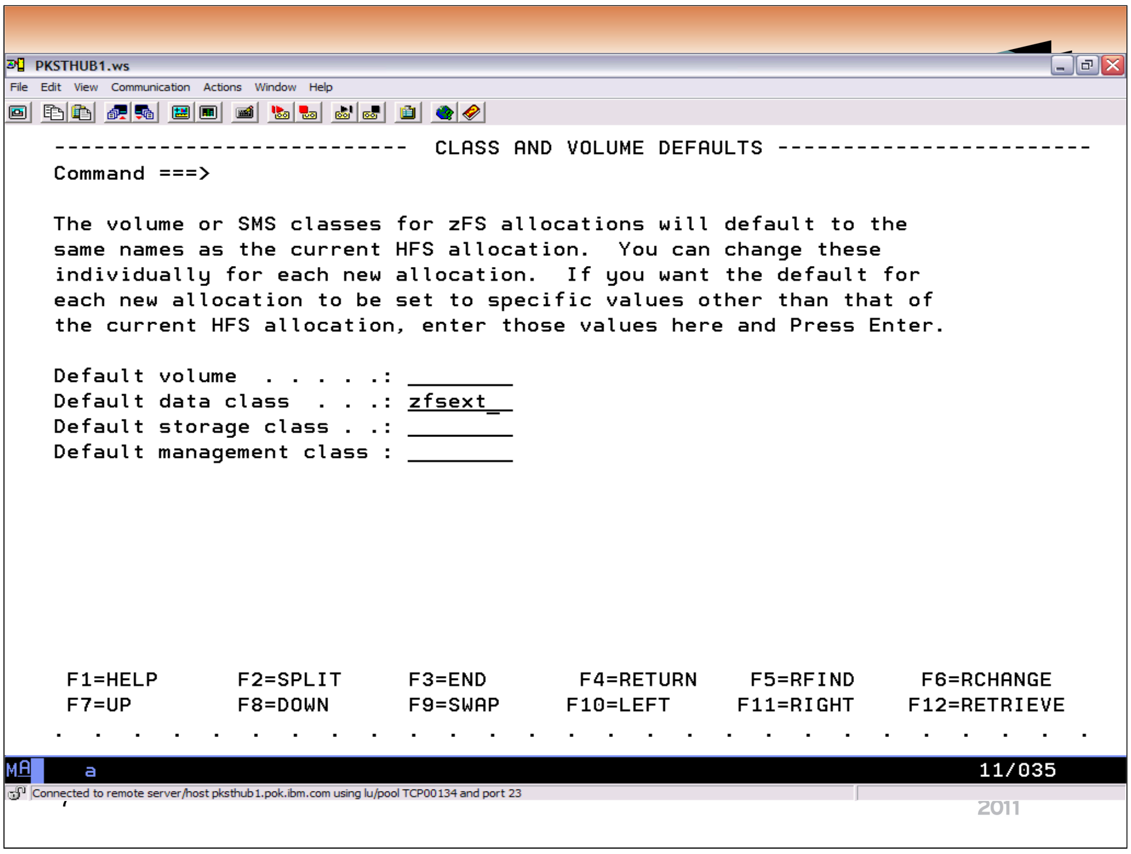
6

•Type TSO BPXWH2Z

•The –c places summary information in a file if job run in the background. If not specified it is written to the console.

•The –v is verbose mode to get more information while tool is processing.

•IMPORTANT any wild card you can use in ISPF you can use here



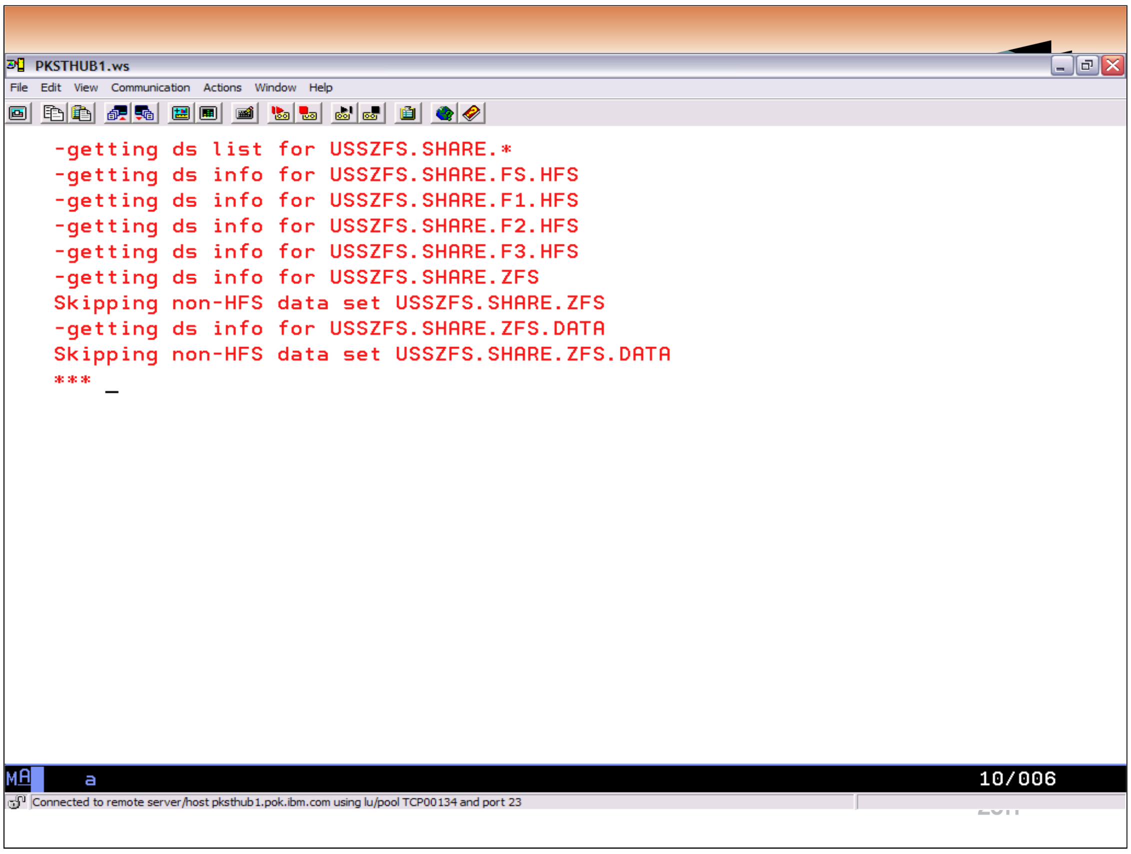
PKSTHUB1.ws

```
---------------------------  CLASS AND VOLUME DEFAULTS ------------------------
Command ===>

The volume or SMS classes for zFS allocations will default to the
same names as the current HFS allocation.  You can change these
individually for each new allocation.  If you want the default for
each new allocation to be set to specific values other than that of
the current HFS allocation, enter those values here and Press Enter.

Default volume  . . . . .: ________
Default data class  . . .: zfsext__
Default storage class . .: ________
Default management class : ________





 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23

This panel just allows you to change the default volume or SMS classes. If you want the new zFSs to have the same vol/SMS class then just press enter.

I changed my default data class because I originally defined my large (14Gig) multi-volume HFS without the extended format attributes set. I could also have done this on the individual filesystem. We will see this later in the example.

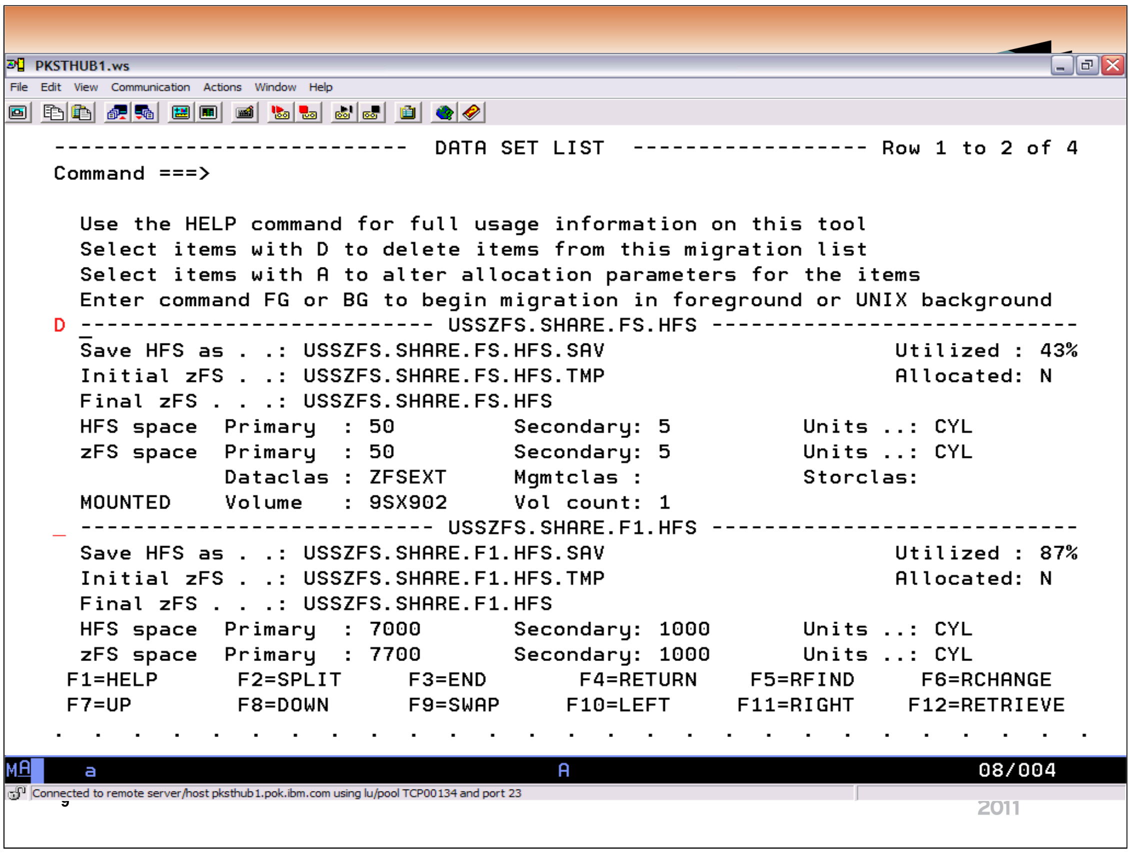


```
-getting ds list for USSZFS.SHARE.*
-getting ds info for USSZFS.SHARE.FS.HFS
-getting ds info for USSZFS.SHARE.F1.HFS
-getting ds info for USSZFS.SHARE.F2.HFS
-getting ds info for USSZFS.SHARE.F3.HFS
-getting ds info for USSZFS.SHARE.ZFS
Skipping non-HFS data set USSZFS.SHARE.ZFS
-getting ds info for USSZFS.SHARE.ZFS.DATA
Skipping non-HFS data set USSZFS.SHARE.ZFS.DATA
*** _
```

```
MA     a                                              10/006
```

Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23

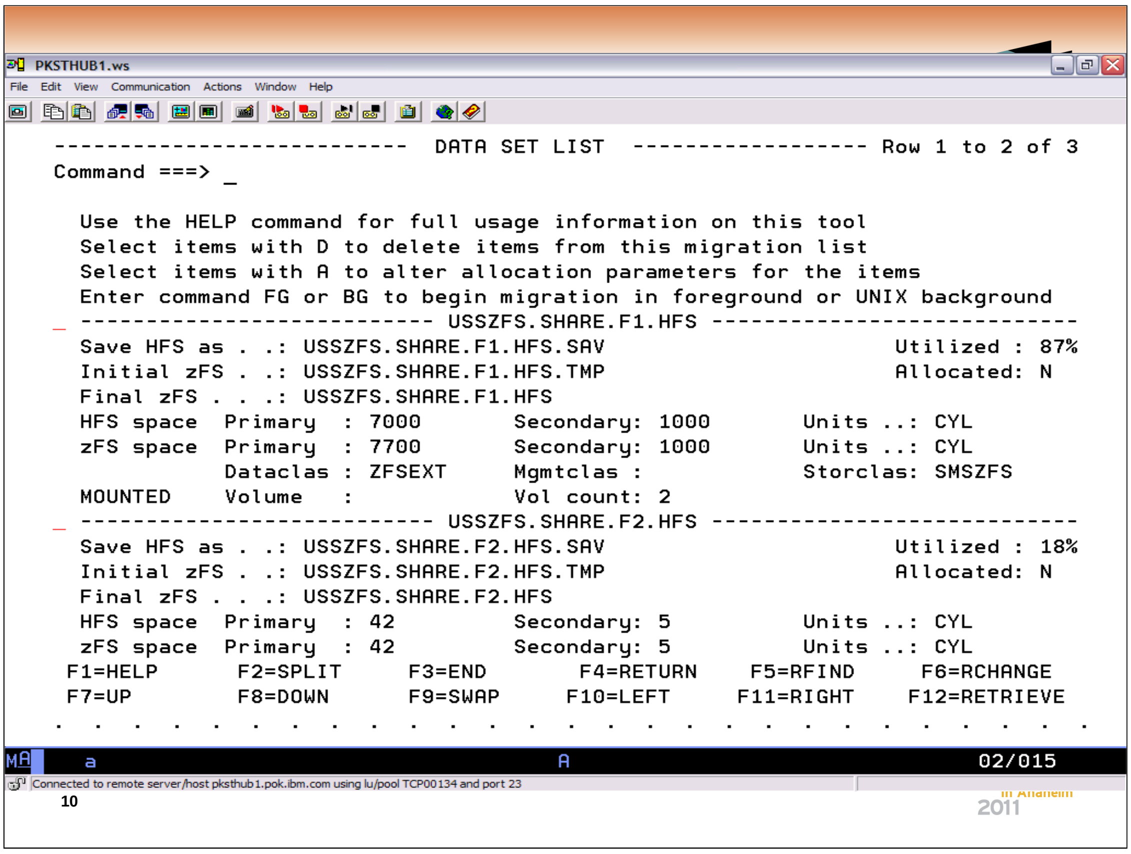
This just shows that the tool found the 4 filesystems that make up my example and the zFS that we defined earlier using ISHELL. As expected it will skip the zFS.



```
 --------------------------  DATA SET LIST  ------------------ Row 1 to 2 of 4
 Command ===>

   Use the HELP command for full usage information on this tool
   Select items with D to delete items from this migration list
   Select items with A to alter allocation parameters for the items
   Enter command FG or BG to begin migration in foreground or UNIX background
 D _-------------------------- USSZFS.SHARE.FS.HFS ---------------------------
   Save HFS as . .: USSZFS.SHARE.FS.HFS.SAV                   Utilized : 43%
   Initial zFS . .: USSZFS.SHARE.FS.HFS.TMP                   Allocated: N
   Final zFS . . .: USSZFS.SHARE.FS.HFS
   HFS space  Primary  : 50        Secondary: 5         Units ..: CYL
   zFS space  Primary  : 50        Secondary: 5         Units ..: CYL
              Dataclas : ZFSEXT    Mgmtclas :           Storclas:
   MOUNTED    Volume   : 9SX902    Vol count: 1
 _ -------------------------- USSZFS.SHARE.F1.HFS ---------------------------
   Save HFS as . .: USSZFS.SHARE.F1.HFS.SAV                   Utilized : 87%
   Initial zFS . .: USSZFS.SHARE.F1.HFS.TMP                   Allocated: N
   Final zFS . . .: USSZFS.SHARE.F1.HFS
   HFS space  Primary  : 7000      Secondary: 1000      Units ..: CYL
   zFS space  Primary  : 7700      Secondary: 1000      Units ..: CYL
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
MA     a                                A                             08/004
Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
```

This just shows that it did find the 3 filesystems that I am working with and one additional one that I do not intend to migrate at this time.

•By placing a D on the red line to the left of the FS name you can selectively delete filesystems that you choose not to perform a migration on at this time.



```
 ---------------------------  DATA SET LIST  ------------------ Row 1 to 2 of 3
 Command ===> _

   Use the HELP command for full usage information on this tool
   Select items with D to delete items from this migration list
   Select items with A to alter allocation parameters for the items
   Enter command FG or BG to begin migration in foreground or UNIX background
 _ --------------------------- USSZFS.SHARE.F1.HFS ----------------------------
   Save HFS as . .: USSZFS.SHARE.F1.HFS.SAV                      Utilized : 87%
   Initial zFS . .: USSZFS.SHARE.F1.HFS.TMP                      Allocated: N
   Final zFS . . .: USSZFS.SHARE.F1.HFS
   HFS space  Primary  : 7000       Secondary: 1000       Units ..: CYL
   zFS space  Primary  : 7700       Secondary: 1000       Units ..: CYL
              Dataclas : ZFSEXT     Mgmtclas :            Storclas: SMSZFS
   MOUNTED    Volume   :            Vol count: 2
 _ --------------------------- USSZFS.SHARE.F2.HFS ----------------------------
   Save HFS as . .: USSZFS.SHARE.F2.HFS.SAV                      Utilized : 18%
   Initial zFS . .: USSZFS.SHARE.F2.HFS.TMP                      Allocated: N
   Final zFS . . .: USSZFS.SHARE.F2.HFS
   HFS space  Primary  : 42         Secondary: 5          Units ..: CYL
   zFS space  Primary  : 42         Secondary: 5          Units ..: CYL
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
MA    a                                   A                              02/015
Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
```

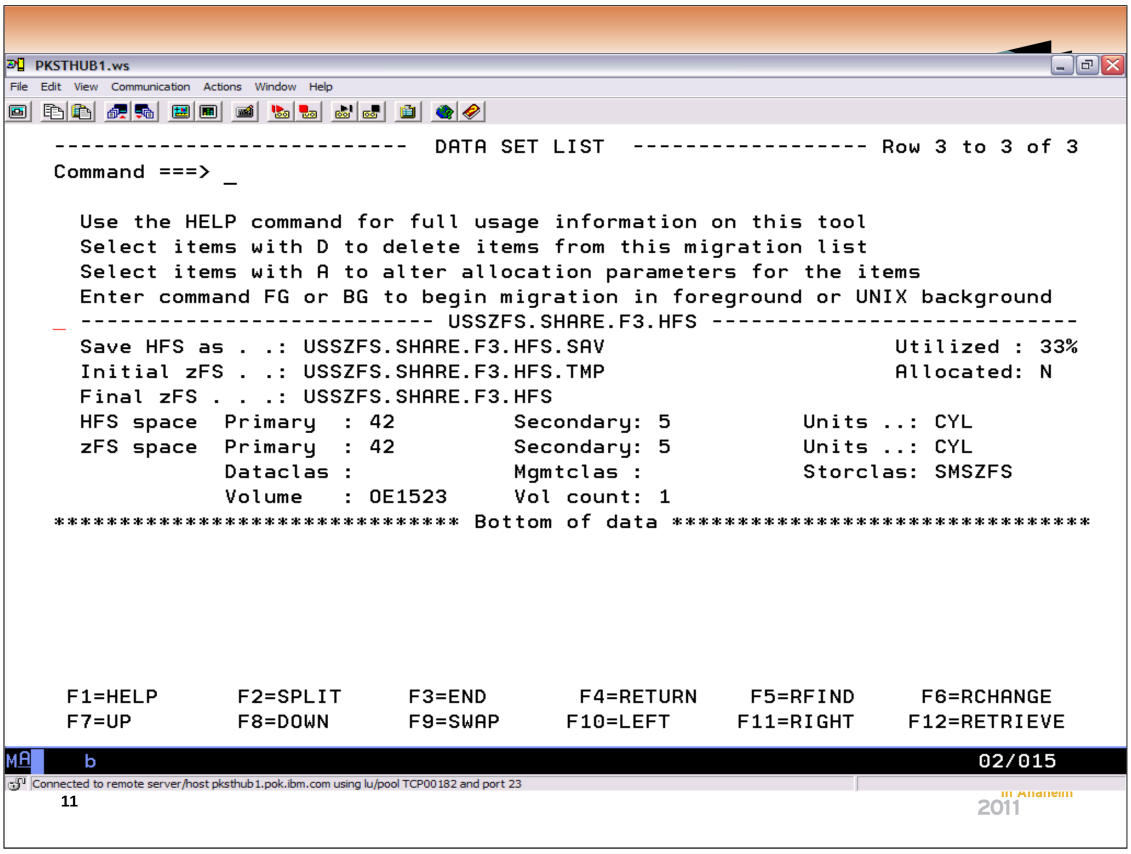
10

•On this panel you see each of the filesystems that met the wildcard search criteria.

•The panel indicated the name of the filesystem being migrated, the temporary name that will be used for the zFS while the migration is in progress, and the name that the HFS will be given so that the zFS can be named with the original HFS name.

•With the APAR **OA18196** you now have the ability to change the final zFS name rather then take the default of the previous HFS name.

•By selecting A on the line next to the FS name you can individualize the allocation parameters

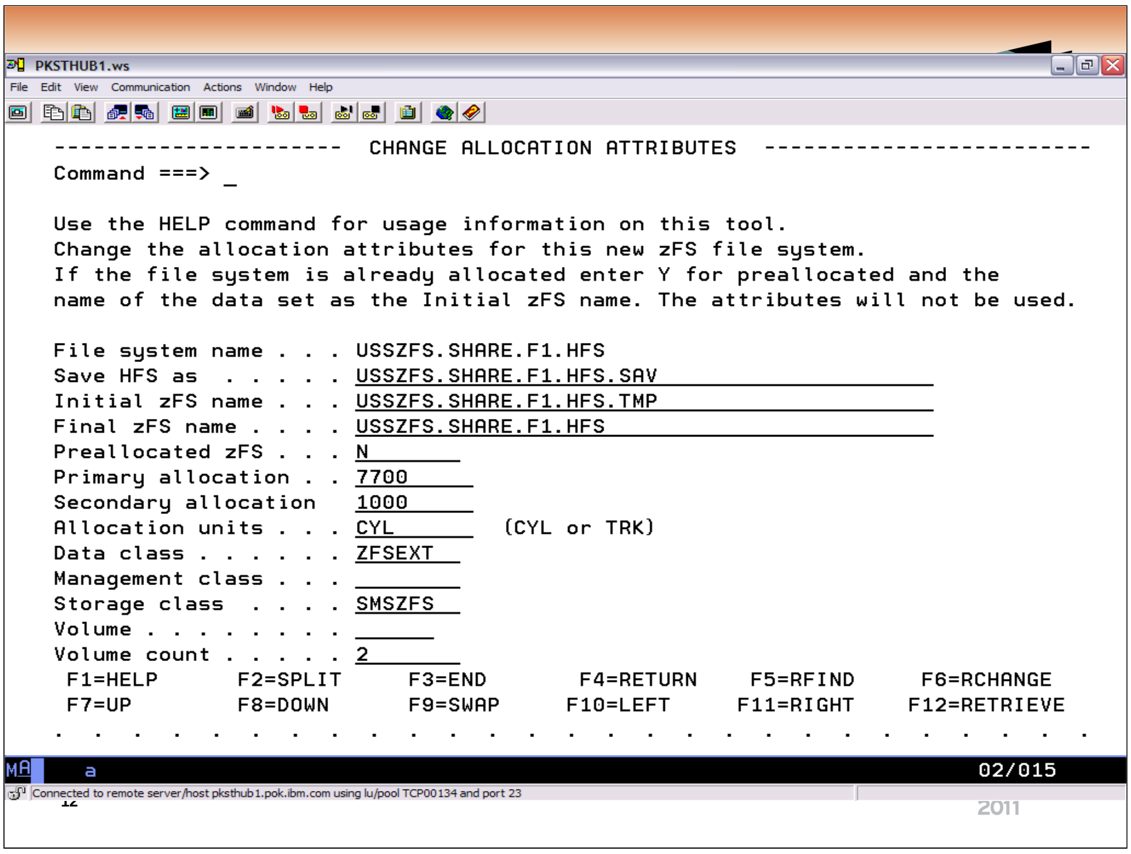


```
 ---------------------------  DATA SET LIST  ------------------ Row 3 to 3 of 3
 Command ===> _

   Use the HELP command for full usage information on this tool
   Select items with D to delete items from this migration list
   Select items with A to alter allocation parameters for the items
   Enter command FG or BG to begin migration in foreground or UNIX background
 _ -------------------------- USSZFS.SHARE.F3.HFS ----------------------------
   Save HFS as . .: USSZFS.SHARE.F3.HFS.SAV                      Utilized : 33%
   Initial zFS . .: USSZFS.SHARE.F3.HFS.TMP                      Allocated: N
   Final zFS . . .: USSZFS.SHARE.F3.HFS
   HFS space  Primary  : 42         Secondary: 5         Units ..: CYL
   zFS space  Primary  : 42         Secondary: 5         Units ..: CYL
              Dataclas :            Mgmtclas :           Storclas: SMSZFS
              Volume   : OE1523     Vol count: 1
 ******************************* Bottom of data ********************************




  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
MA     b                                                                 02/015
```

Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00182 and port 23
11
2011

•This is just a continuation of the panel and I am showing it so you can see that the filesystem that is unmounted is still part of the migration process.

•From here you just type BG to get the migration started.

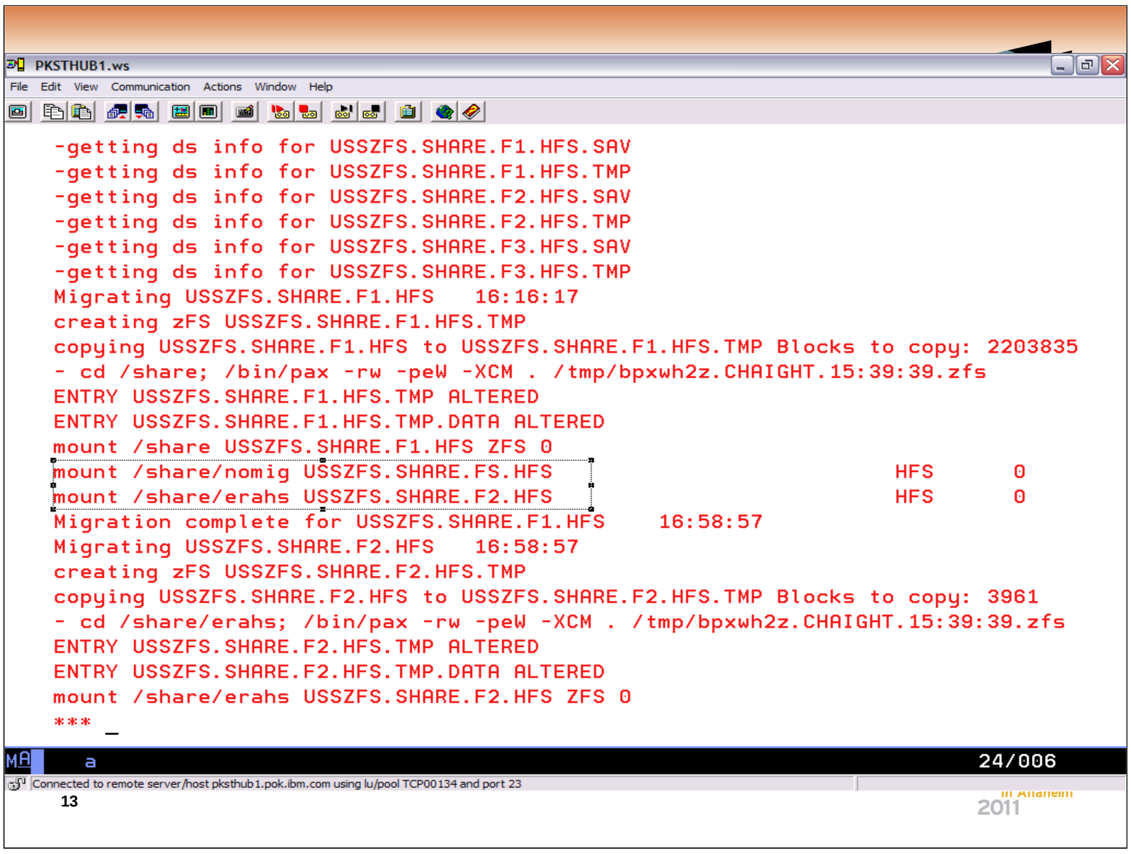


```
---------------------  CHANGE ALLOCATION ATTRIBUTES  ------------------------
Command ===> _

Use the HELP command for usage information on this tool.
Change the allocation attributes for this new zFS file system.
If the file system is already allocated enter Y for preallocated and the
name of the data set as the Initial zFS name. The attributes will not be used.

File system name . . . USSZFS.SHARE.F1.HFS
Save HFS as  . . . . . USSZFS.SHARE.F1.HFS.SAV
Initial zFS name . . . USSZFS.SHARE.F1.HFS.TMP
Final zFS name . . . . USSZFS.SHARE.F1.HFS
Preallocated zFS . . . N
Primary allocation . . 7700
Secondary allocation   1000
Allocation units . . . CYL        (CYL or TRK)
Data class . . . . . . ZFSEXT
Management class . . .
Storage class  . . . . SMSZFS
Volume . . . . . . . .
Volume count . . . . . 2
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
MA    a                                                              02/015
Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
```

•This is the panel that you are given if you choose to make changes to any of the migration attributes.

•This is also where you would specify that you have preallocated a VSAM LDS. This is a necessary step if your filesystem spans volumes. The tool can only handle filesystem that fit on a single volume unless you preallocate.

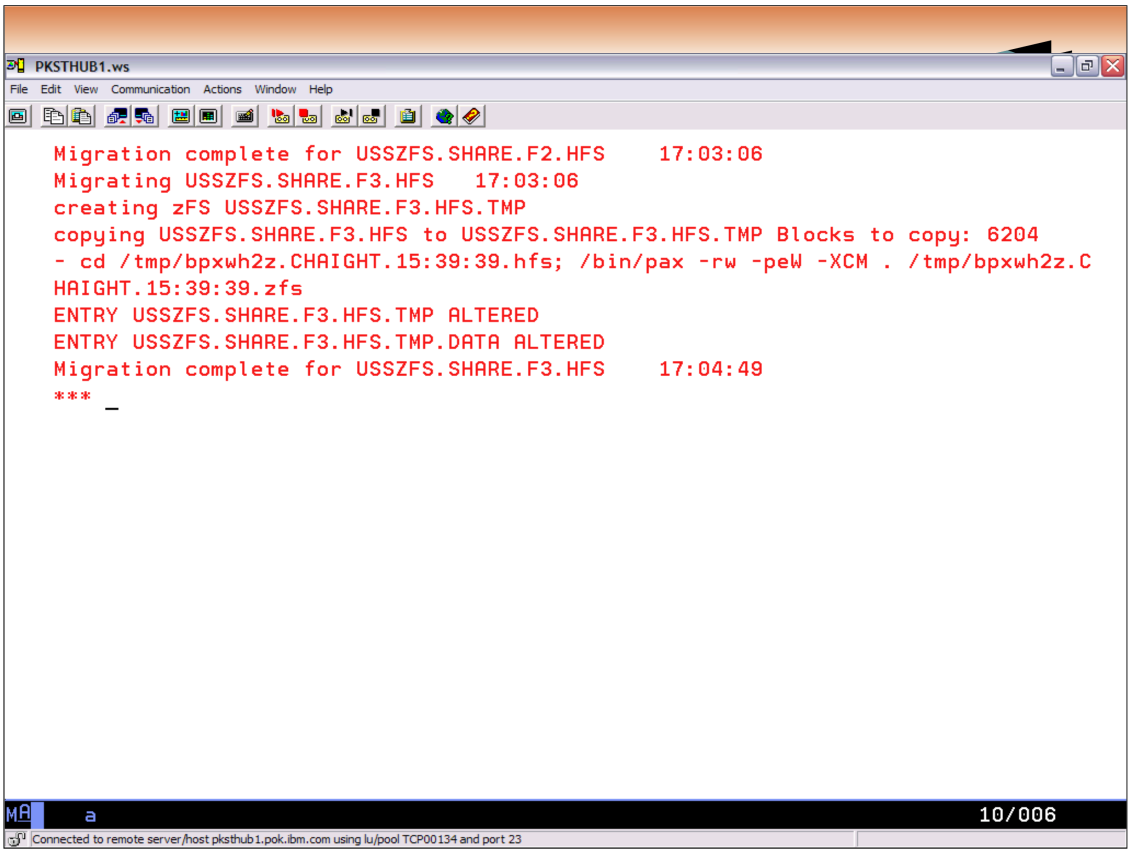
•APAR **OA18196** removes the restriction on the tool that required you to preallocate the zFS when it is multi-volume. This is also where you would add the data class that defines extended format attributes if the filesystem is larger then 4G and they were not previously defined or defaulted on the earlier panel.



```
-getting ds info for USSZFS.SHARE.F1.HFS.SAV
-getting ds info for USSZFS.SHARE.F1.HFS.TMP
-getting ds info for USSZFS.SHARE.F2.HFS.SAV
-getting ds info for USSZFS.SHARE.F2.HFS.TMP
-getting ds info for USSZFS.SHARE.F3.HFS.SAV
-getting ds info for USSZFS.SHARE.F3.HFS.TMP
Migrating USSZFS.SHARE.F1.HFS   16:16:17
creating zFS USSZFS.SHARE.F1.HFS.TMP
copying USSZFS.SHARE.F1.HFS to USSZFS.SHARE.F1.HFS.TMP Blocks to copy: 2203835
- cd /share; /bin/pax -rw -peW -XCM . /tmp/bpxwh2z.CHAIGHT.15:39:39.zfs
ENTRY USSZFS.SHARE.F1.HFS.TMP ALTERED
ENTRY USSZFS.SHARE.F1.HFS.TMP.DATA ALTERED
mount /share USSZFS.SHARE.F1.HFS ZFS 0
mount /share/nomig USSZFS.SHARE.FS.HFS                          HFS      0
mount /share/erahs USSZFS.SHARE.F2.HFS                          HFS      0
Migration complete for USSZFS.SHARE.F1.HFS    16:58:57
Migrating USSZFS.SHARE.F2.HFS   16:58:57
creating zFS USSZFS.SHARE.F2.HFS.TMP
copying USSZFS.SHARE.F2.HFS to USSZFS.SHARE.F2.HFS.TMP Blocks to copy: 3961
- cd /share/erahs; /bin/pax -rw -peW -XCM . /tmp/bpxwh2z.CHAIGHT.15:39:39.zfs
ENTRY USSZFS.SHARE.F2.HFS.TMP ALTERED
ENTRY USSZFS.SHARE.F2.HFS.TMP.DATA ALTERED
mount /share/erahs USSZFS.SHARE.F2.HFS ZFS 0
*** _
MA     a                                                         24/006
```

Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
13
2011

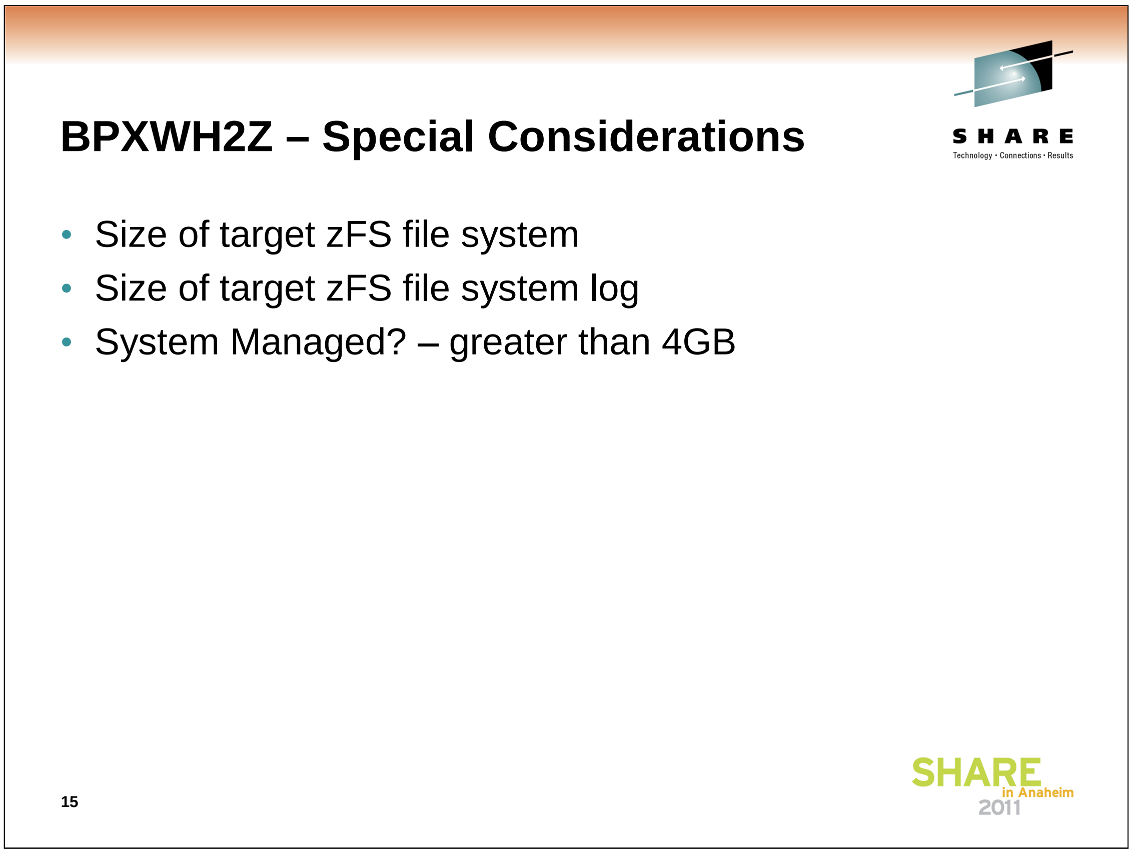
•An interesting thing to note is that in the case that the filesystem that was being migrated had a mount below it, the filesystem was unmounted, the migration occurred, the new zfs was mounted and the HFS was mounted back on the mountpoint. What this means is you can actually do the ROOT all the way down at one time if you wanted to.



```
Migration complete for USSZFS.SHARE.F2.HFS    17:03:06
Migrating USSZFS.SHARE.F3.HFS   17:03:06
creating zFS USSZFS.SHARE.F3.HFS.TMP
copying USSZFS.SHARE.F3.HFS to USSZFS.SHARE.F3.HFS.TMP Blocks to copy: 6204
- cd /tmp/bpxwh2z.CHAIGHT.15:39:39.hfs; /bin/pax -rw -peW -XCM . /tmp/bpxwh2z.C
HAIGHT.15:39:39.zfs
ENTRY USSZFS.SHARE.F3.HFS.TMP ALTERED
ENTRY USSZFS.SHARE.F3.HFS.TMP.DATA ALTERED
Migration complete for USSZFS.SHARE.F3.HFS    17:04:49
*** _
```

```
MA    a                                                          10/006
Connected to remote server/host pksthub1.pok.ibm.com using lu/pool TCP00134 and port 23
```

This just shows that even the filesystem that was not mounted goes through migration.



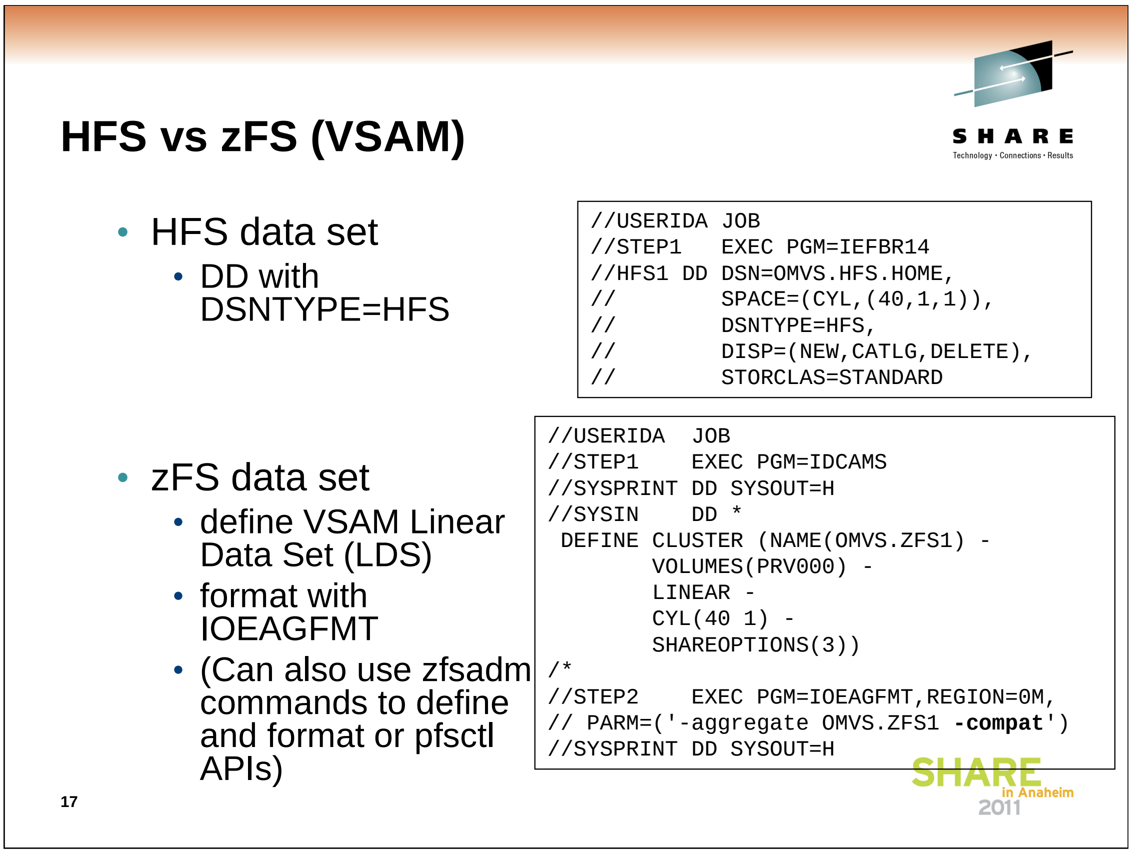
BPXWH2Z – Special Considerations

- Size of target zFS file system
- Size of target zFS file system log
- System Managed? – greater than 4GB

15

Things that are different

- HFS vs zFS (VSAM)
- zFS is a logging file system
- Growing a file system



HFS vs zFS (VSAM)

- 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)

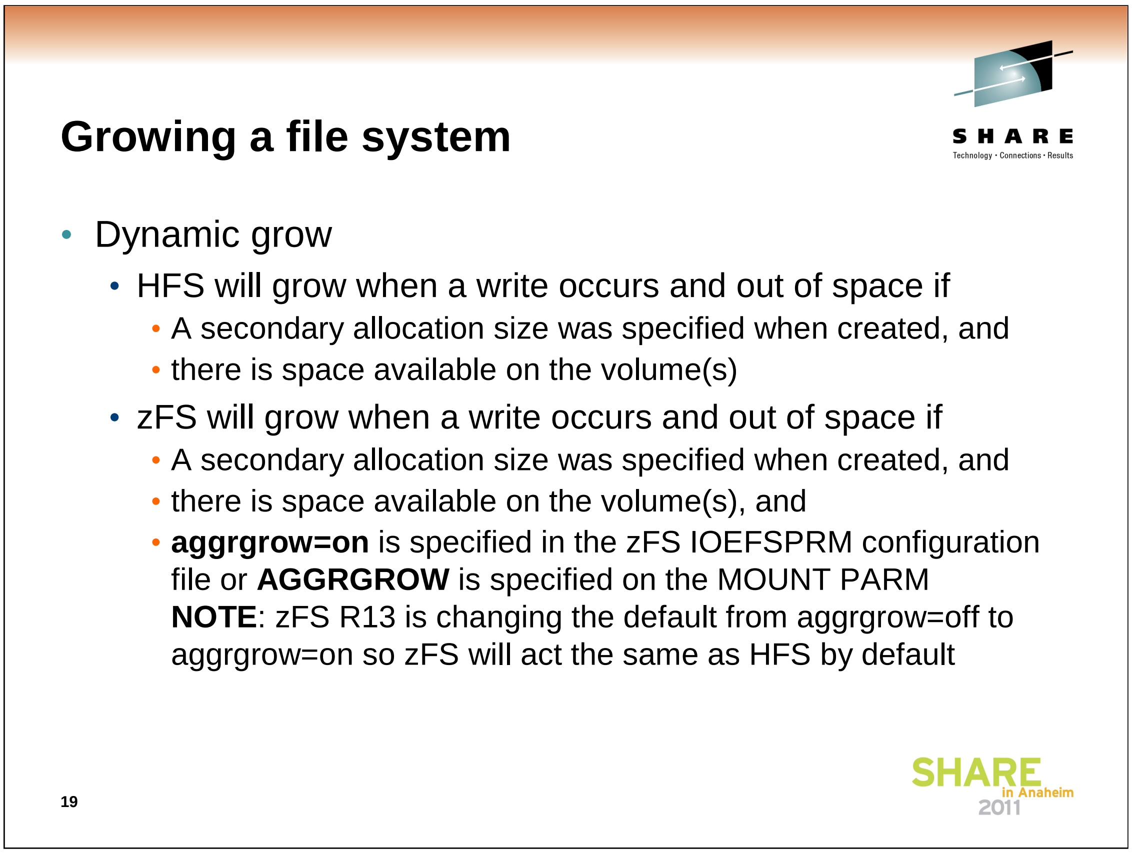
```
//USERIDA  JOB
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=H
//SYSIN    DD *
 DEFINE CLUSTER (NAME(OMVS.ZFS1) -
        VOLUMES(PRV000) -
        LINEAR -
        CYL(40 1) -
        SHAREOPTIONS(3))
/*
//STEP2    EXEC PGM=IOEAGFMT,REGION=0M,
// PARM=('-aggregate OMVS.ZFS1 -compat')
//SYSPRINT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

17

SHARE in Anaheim 2011

zFS is a logging file system

- zFS is a logging file system and has improved recovery from system failures
  - Metadata updates are transactional and are logged to maintain the structural integrity of the file system
- zFS has a utility called IOEAGSLV (Salvager) that can verify and optionally attempt to repair a zFS file system in the unlikely event that it becomes corrupted



Growing a file system

- Dynamic grow
  - HFS will grow when a write occurs and out of space if
    - A secondary allocation size was specified when created, and
    - there is space available on the volume(s)
  - zFS will grow when a write occurs and out of space if
    - A secondary allocation size was specified when created, and
    - there is space available on the volume(s), and
    - **aggrgrow=on** is specified in the zFS IOEFSPRM configuration file or **AGGRGROW** is specified on the MOUNT PARM
      NOTE: zFS R13 is changing the default from aggrgrow=off to aggrgrow=on so zFS will act the same as HFS by default

19

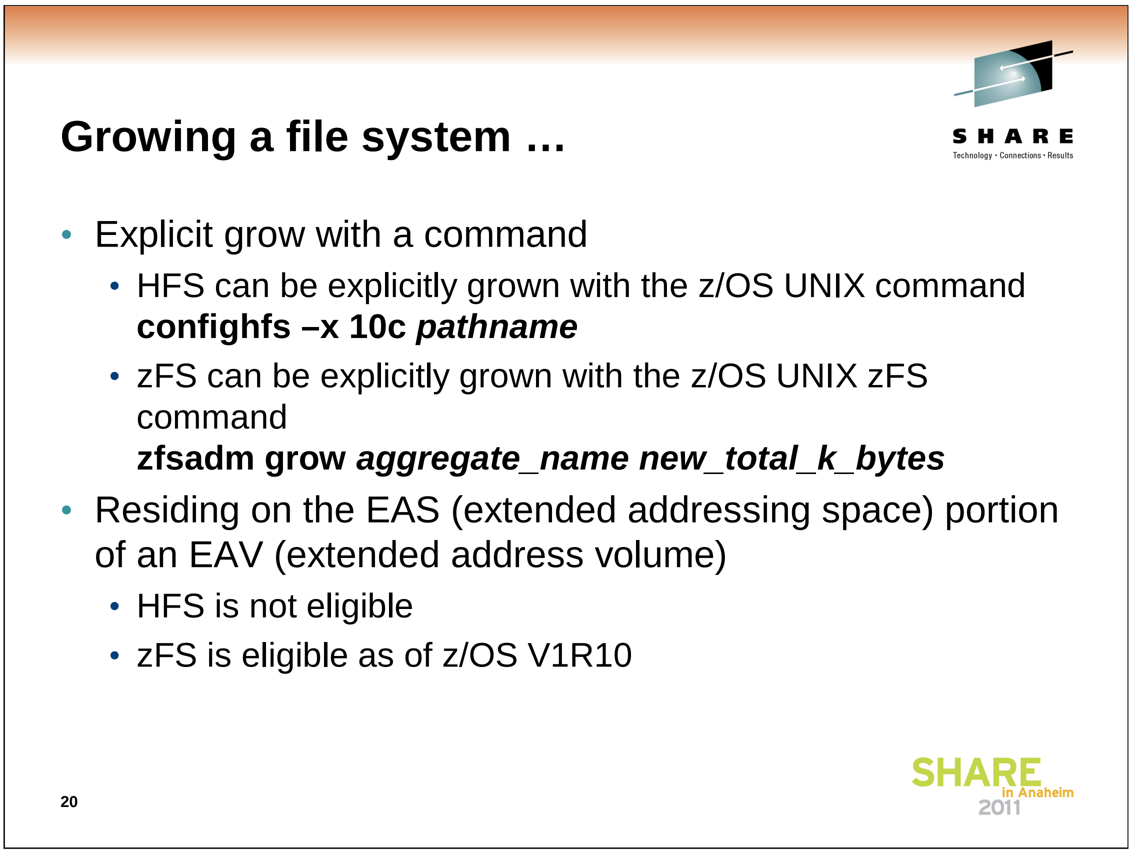
SHARE in Anaheim 2011

From TSO/E

```
MOUNT FILESYSTEM('OMVS.MNT.FS1.ZFS') TYPE(ZFS)
MODE(RDWR) MOUNTPOINT('/zfsmnt1') PARM(‘AGGRGROW’)
```

From the z/OS UNIX shell

```
/usr/sbin/mount -t ZFS -f OMVS.MNT.FS1.ZFS -o ‘AGGRGROW' /zfsmnt1
```



Growing a file system …

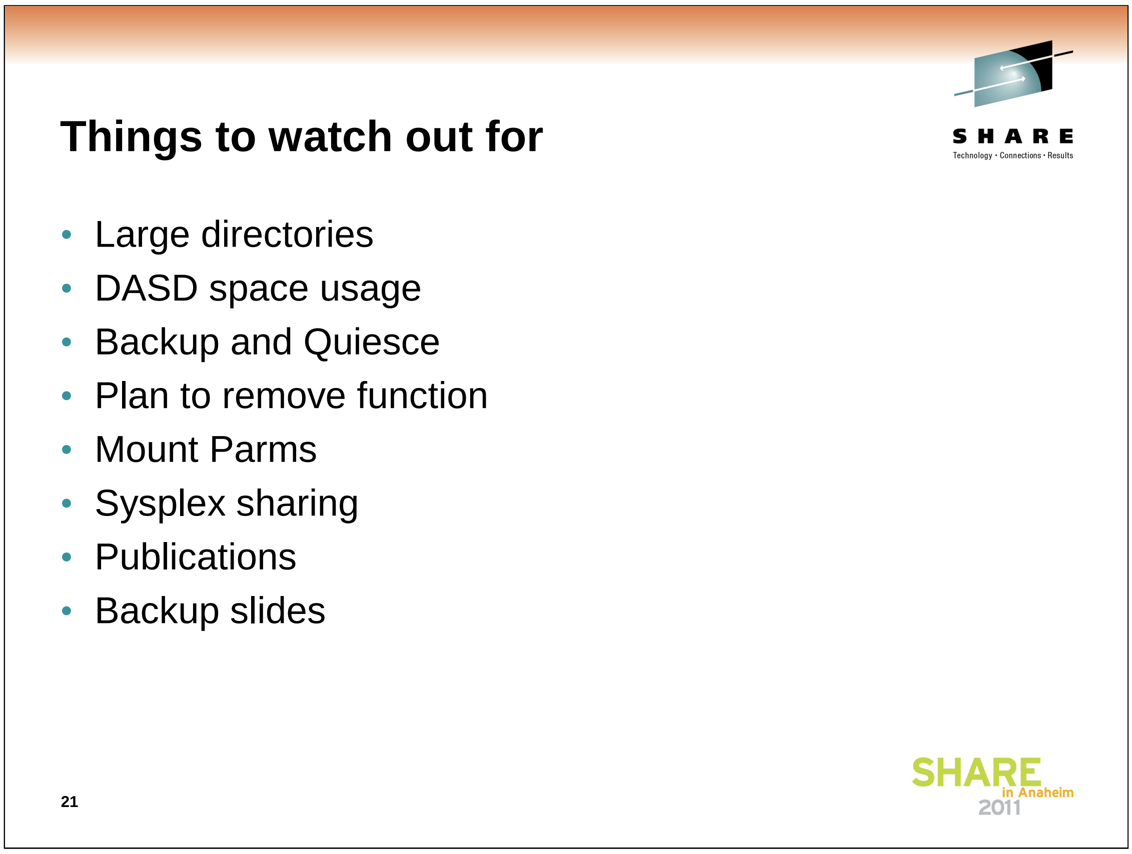
- Explicit grow with a command
  - HFS can be explicitly grown with the z/OS UNIX command **confighfs –x 10c *pathname***
  - zFS can be explicitly grown with the z/OS UNIX zFS command **zfsadm grow *aggregate_name new_total_k_bytes***
- Residing on the EAS (extended addressing space) portion of an EAV (extended address volume)
  - HFS is not eligible
  - zFS is eligible as of z/OS V1R10

20

SHARE in Anaheim 2011

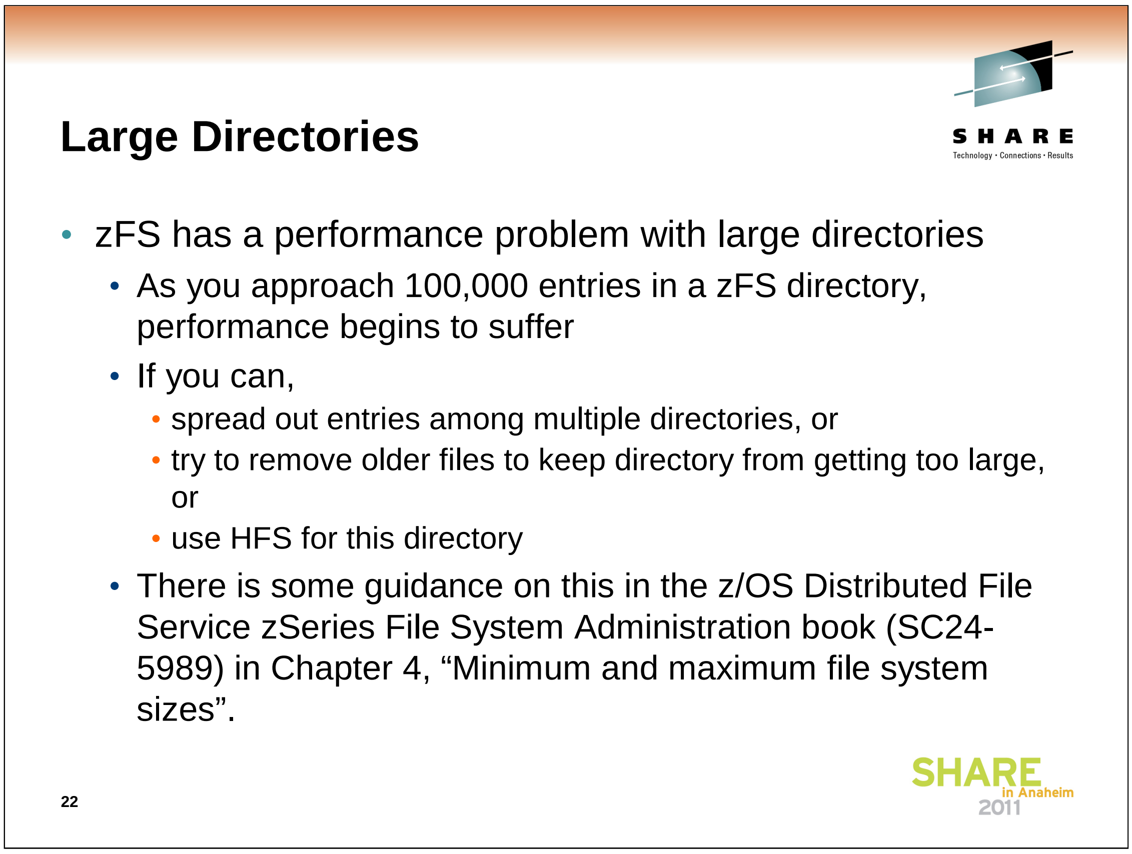
zfsadm aggrinfo displays the current size of the aggregate in K-bytes

df –k also displays the current size of the aggregate in K-bytes



Things to watch out for

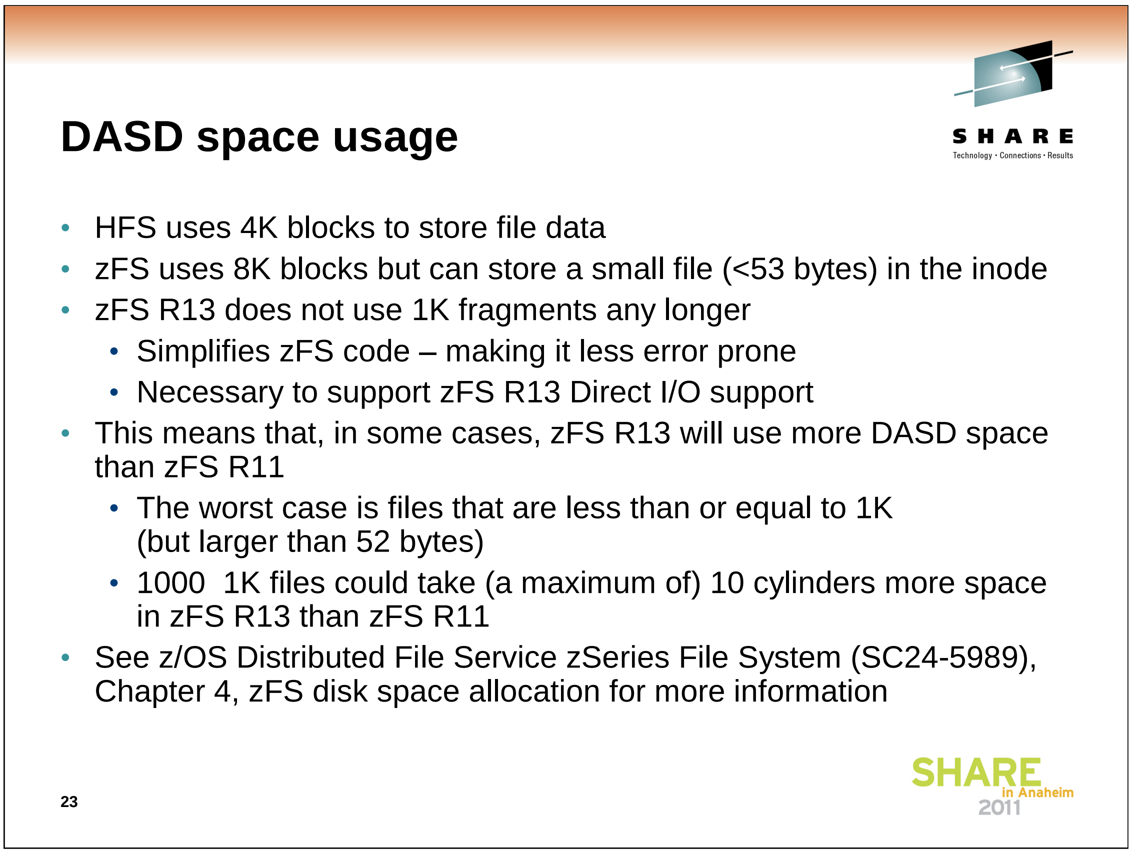
- Large directories
- DASD space usage
- Backup and Quiesce
- Plan to remove function
- Mount Parms
- Sysplex sharing
- Publications
- Backup slides



Large Directories

- zFS has a performance problem with large directories
  - As you approach 100,000 entries in a zFS directory, performance begins to suffer
  - If you can,
    - spread out entries among multiple directories, or
    - try to remove older files to keep directory from getting too large, or
    - use HFS for this directory
  - There is some guidance on this in the z/OS Distributed File Service zSeries File System Administration book (SC24-5989) in Chapter 4, “Minimum and maximum file system sizes”.

22



DASD space usage

- HFS uses 4K blocks to store file data
- zFS uses 8K blocks but can store a small file (<53 bytes) in the inode
- zFS R13 does not use 1K fragments any longer
  - Simplifies zFS code – making it less error prone
  - Necessary to support zFS R13 Direct I/O support
- This means that, in some cases, zFS R13 will use more DASD space than zFS R11
  - The worst case is files that are less than or equal to 1K (but larger than 52 bytes)
  - 1000 1K files could take (a maximum of) 10 cylinders more space in zFS R13 than zFS R11
- See z/OS Distributed File Service zSeries File System (SC24-5989), Chapter 4, zFS disk space allocation for more information

23

Calculations

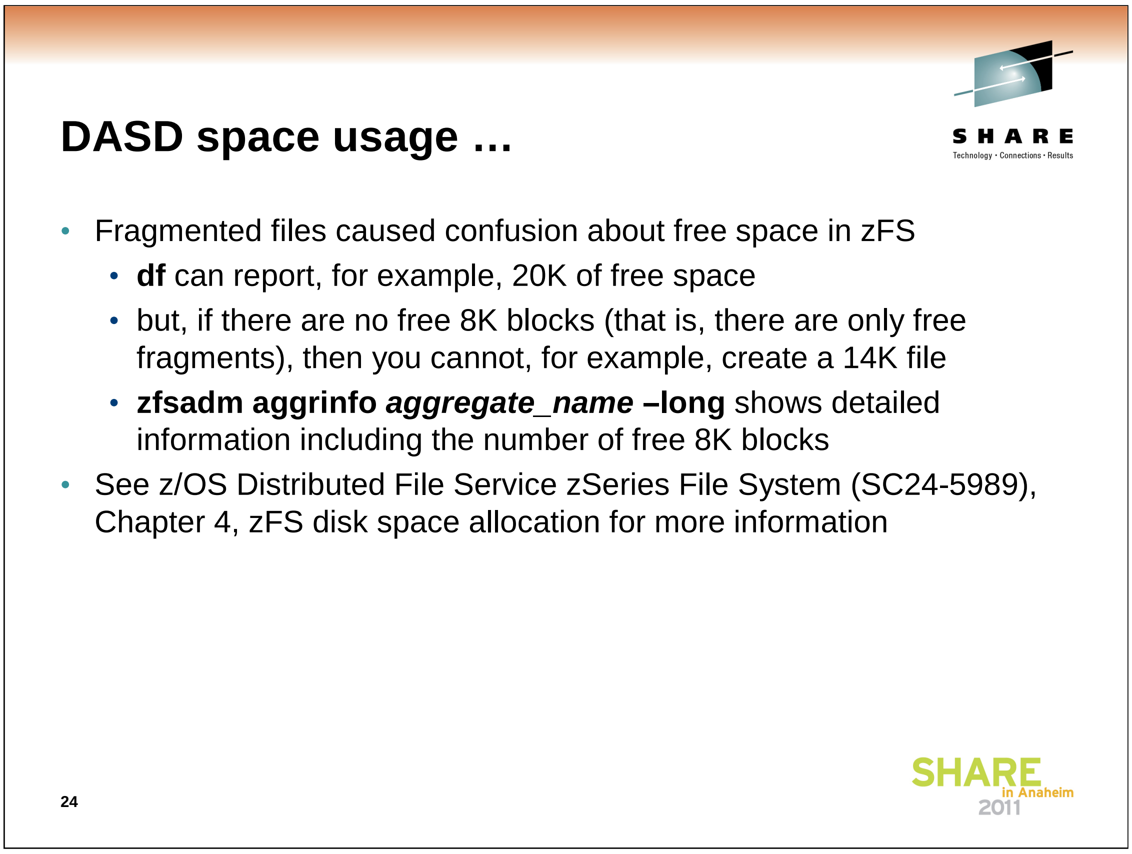
1000 small files uses 1000K in R11 (assuming they are perfectly packed into 8K blocks)

1000 small files uses 8000K in R13

8000K - 1000K = 7000K extra data

7000K / 720K per cylinder = 9.7222 cyl

Note that this calculation is showing the **difference** between storing 1000 1K files in zFS R13 and storing 1000 1K files in zFS R11. The total amount of space to store 1000 1K files in zFS is more than this due to metatdata information, directory information and other fixed storage required by zFS for the log file, the bitmap, the file system table, etc. But this other space required is generally the same between zFS R13 and zFS R11. So, if you already had 1000 1K files stored in zFS R11, this calculation shows you the maximum additional storage you would need to create those files using zFS R13.



DASD space usage …

- Fragmented files caused confusion about free space in zFS
  - **df** can report, for example, 20K of free space
  - but, if there are no free 8K blocks (that is, there are only free fragments), then you cannot, for example, create a 14K file
  - **zfsadm aggrinfo *aggregate_name* –long** shows detailed information including the number of free 8K blocks
- See z/OS Distributed File Service zSeries File System (SC24-5989), Chapter 4, zFS disk space allocation for more information

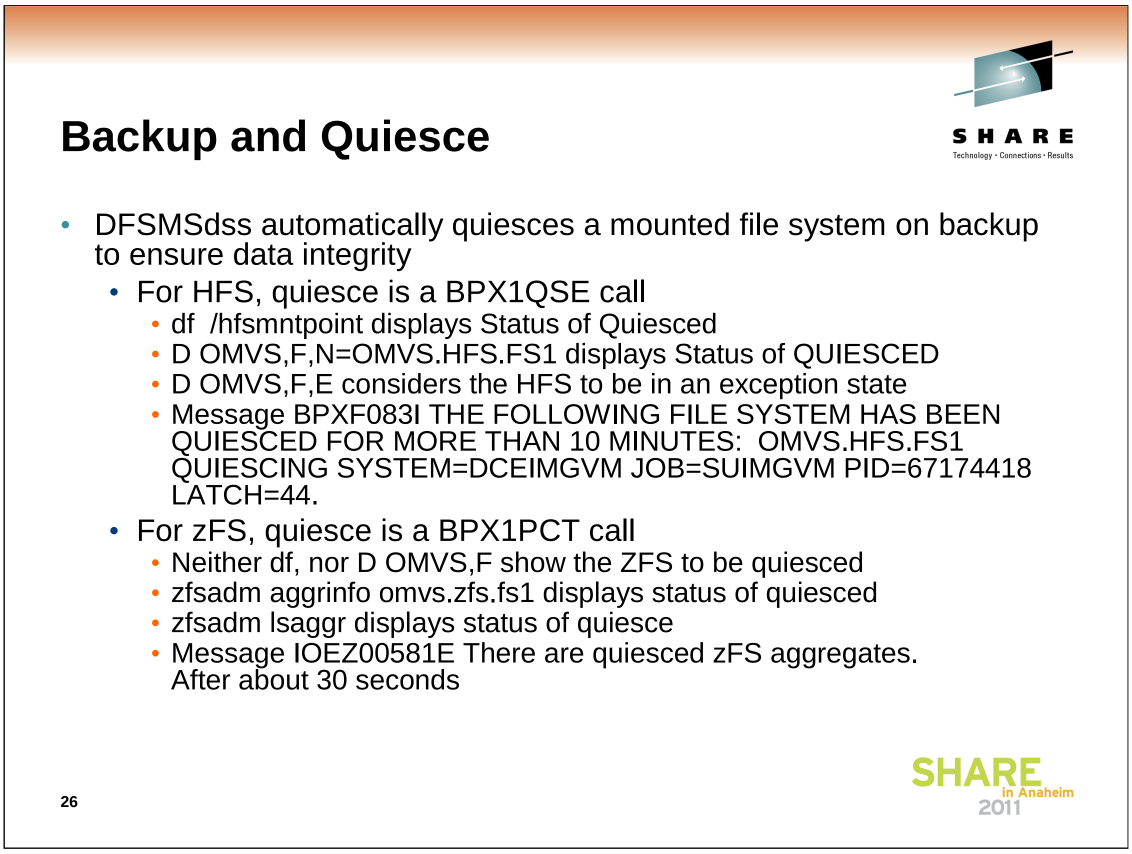
24

```
# zfsadm aggrinfo PLEX.JMS.AGGR004.LDS0004 -long
PLEX.JMS.AGGR004.LDS0004 (R/W COMP): 500 K free out of total
12960
version 1.4
auditfid C3C6C3F0 F0F0051E 0000

         55 free 8k blocks;        60 free 1K fragments
        112 K log file;          24 K filesystem table
          8 K bitmap file
```

Space in a file system …

- An HFS that is multi-volume must be SMS managed (and cataloged)
- A zFS that is > 4GB must be extended addressability in the data class definition and therefore SMS managed (A VSAM LDS is always cataloged)

SHARE in Anaheim 2011



Backup and Quiesce

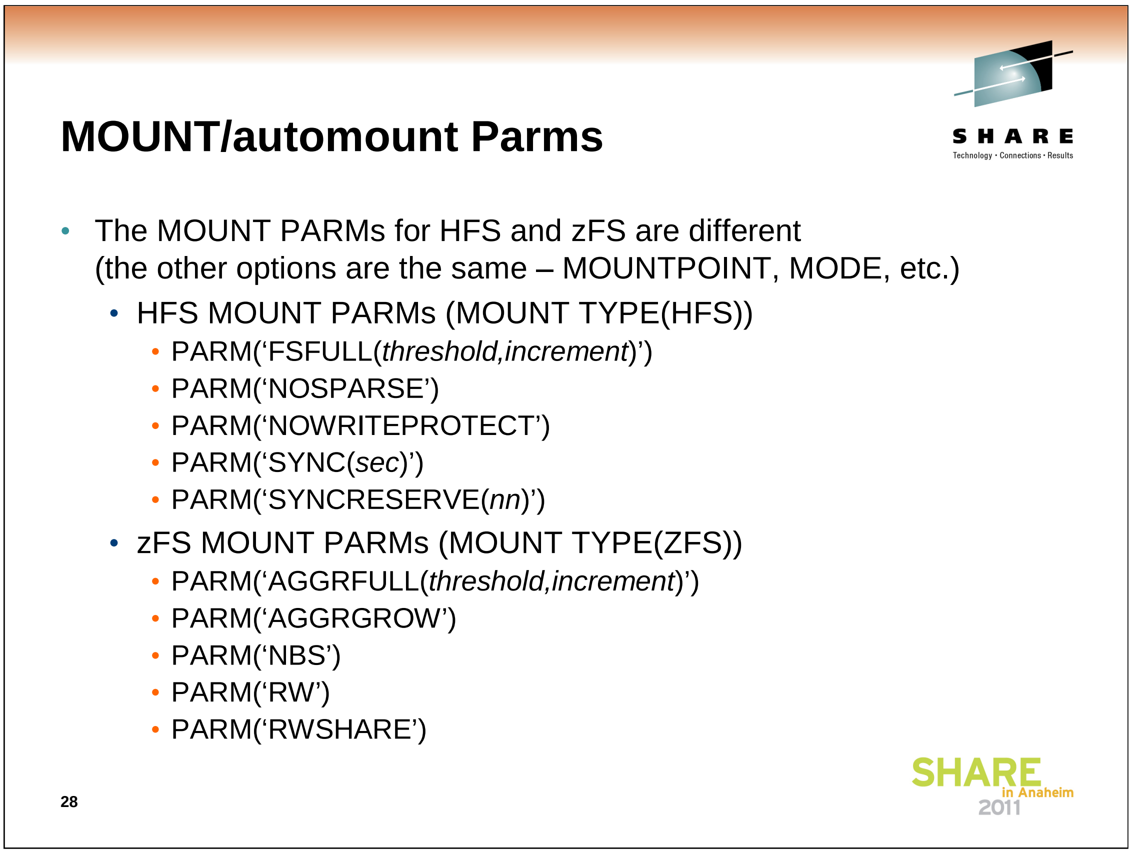
- DFSMSdss automatically quiesces a mounted file system on backup to ensure data integrity
  - For HFS, quiesce is a BPX1QSE call
    - df /hfsmntpoint displays Status of Quiesced
    - D OMVS,F,N=OMVS.HFS.FS1 displays Status of QUIESCED
    - D OMVS,F,E considers the HFS to be in an exception state
    - Message BPXF083I THE FOLLOWING FILE SYSTEM HAS BEEN QUIESCED FOR MORE THAN 10 MINUTES: OMVS.HFS.FS1 QUIESCING SYSTEM=DCEIMGVM JOB=SUIMGVM PID=67174418 LATCH=44.
  - For zFS, quiesce is a BPX1PCT call
    - Neither df, nor D OMVS,F show the ZFS to be quiesced
    - zfsadm aggrinfo omvs.zfs.fs1 displays status of quiesced
    - zfsadm lsaggr displays status of quiesce
    - Message IOEZ00581E There are quiesced zFS aggregates. After about 30 seconds

26

If another sysplex member joins, an HFS (must be R/O) will not be mounted on the joining system until the HFS is unquiesced. See z/OS UNIX System Services Programming: Assembler Callable Services Reference, Chapter 2, quiesce, Characteristics and Restrictions. A ZFS will be mounted as soon as the system joins.

Plan to remove function

- In February 2011, IBM announced
  - z/OS V1.13 is planned to be the last release to support multi-file system zSeries File System (zFS) aggregates, including zFS clones. Support for the zfsadm clone command and mount support for zFS file system data sets containing a cloned (.bak) file system will be removed. IBM recommends that you use copy functions such as pax and DFSMSdss to back up z/OS UNIX file systems to separate file systems. Support for zFS compatibility mode aggregates will remain.



MOUNT/automount Parms

- The MOUNT PARMs for HFS and zFS are different
  (the other options are the same – MOUNTPOINT, MODE, etc.)
  - HFS MOUNT PARMs (MOUNT TYPE(HFS))
    - PARM('FSFULL(*threshold,increment*)')
    - PARM('NOSPARSE')
    - PARM('NOWRITEPROTECT')
    - PARM('SYNC(*sec*)')
    - PARM('SYNCRESERVE(*nn*)')
  - zFS MOUNT PARMs (MOUNT TYPE(ZFS))
    - PARM('AGGRFULL(*threshold,increment*)')
    - PARM('AGGRGROW')
    - PARM('NBS')
    - PARM('RW')
    - PARM('RWSHARE')

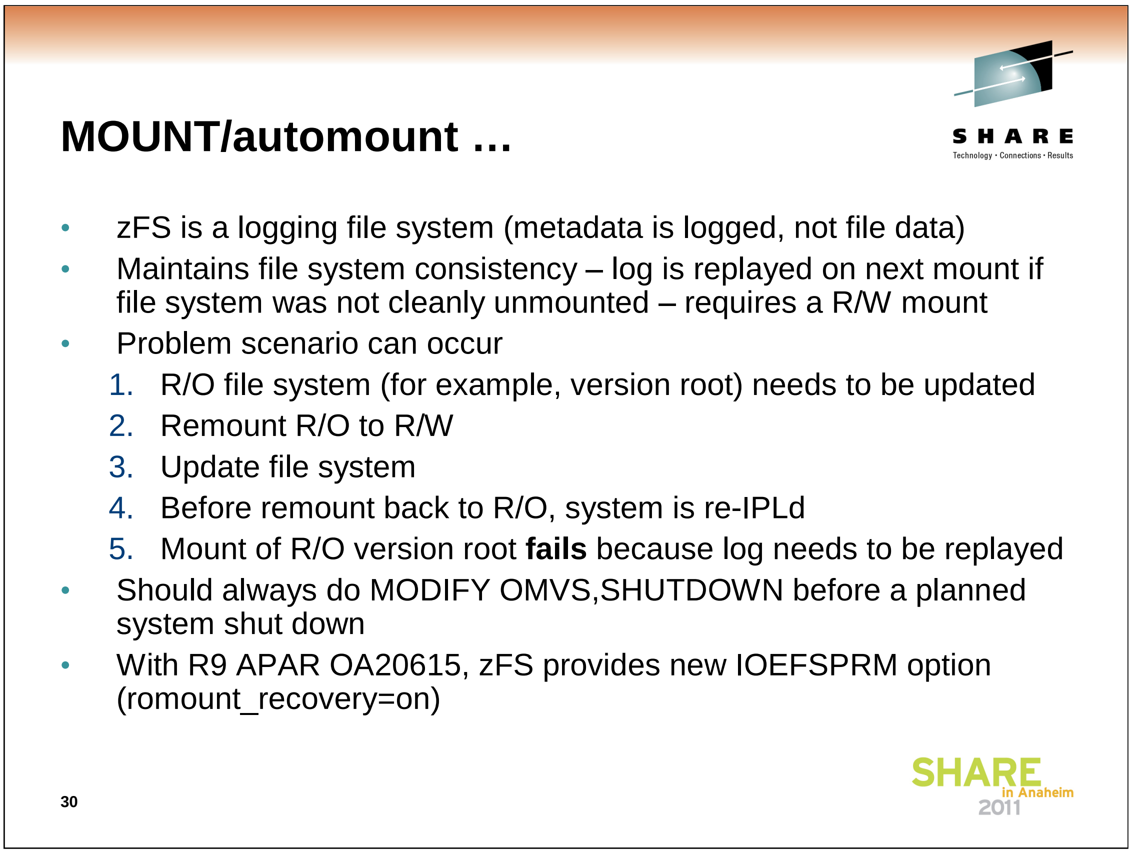
28

HFS MOUNT PARMs described in z/OS MVS Initialization and Tuning Reference (SA22-7592)

zFS MOUNT PARMs described in z/OS Distributed File Service zFS Administration (SC24-5989)

MOUNT/automount Parms…

- Generic file system TYPE on MOUNT
  - If you specify TYPE(HFS) and the data set is not HFS or is not found, it is treated as ZFS (and you get a zFS reason code)
  - If you specify TYPE(ZFS) and the data set is HFS, it is treated as HFS
  - In each of these cases where the TYPE did not match the actual data set type, **THE MOUNT PARMs ARE DISCARDED**
    (we don't want the mount to fail due to invalid PARMs)
  - Once you have fully migrated a file system from HFS to zFS, you should specify TYPE(ZFS) so that MOUNT PARMs are effective

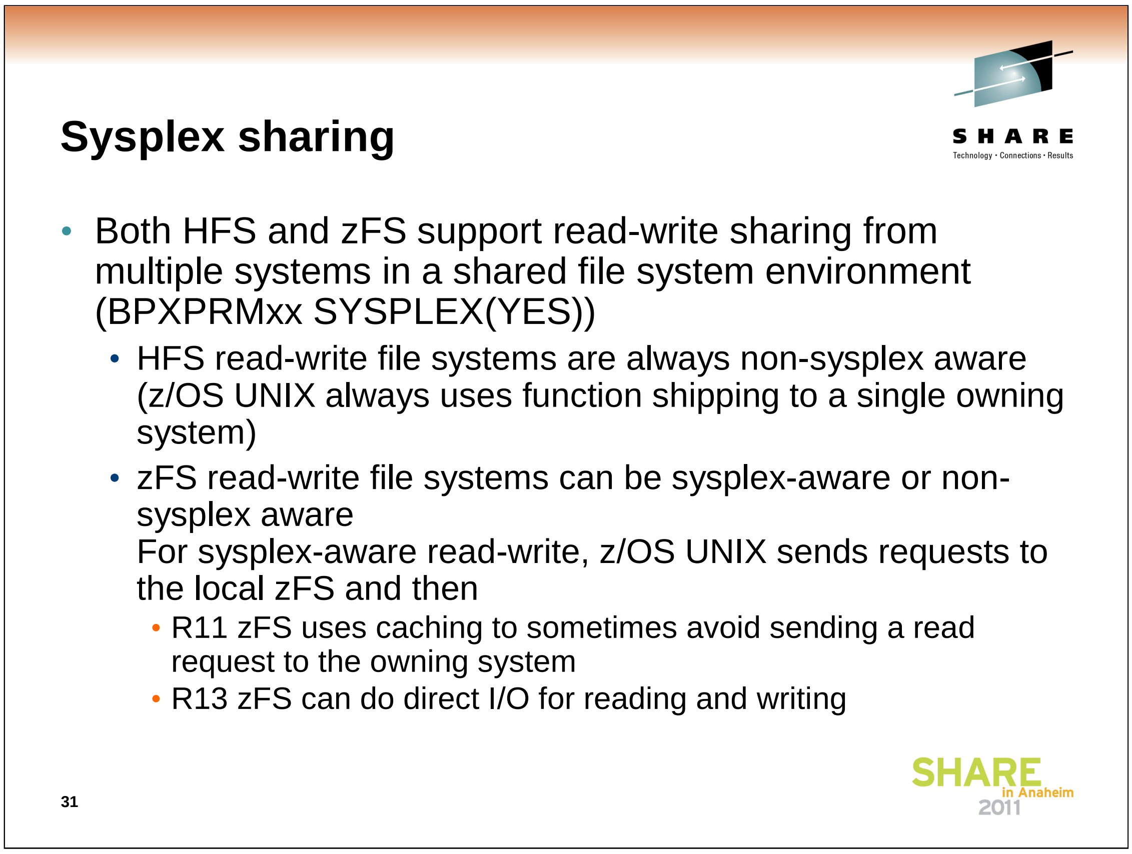


MOUNT/automount …

- zFS is a logging file system (metadata is logged, not file data)
- Maintains file system consistency – log is replayed on next mount if file system was not cleanly unmounted – requires a R/W mount
- Problem scenario can occur
  1. R/O file system (for example, version root) needs to be updated
  2. Remount R/O to R/W
  3. Update file system
  4. Before remount back to R/O, system is re-IPLd
  5. Mount of R/O version root **fails** because log needs to be replayed
- Should always do MODIFY OMVS,SHUTDOWN before a planned system shut down
- With R9 APAR OA20615, zFS provides new IOEFSPRM option (romount_recovery=on)

30

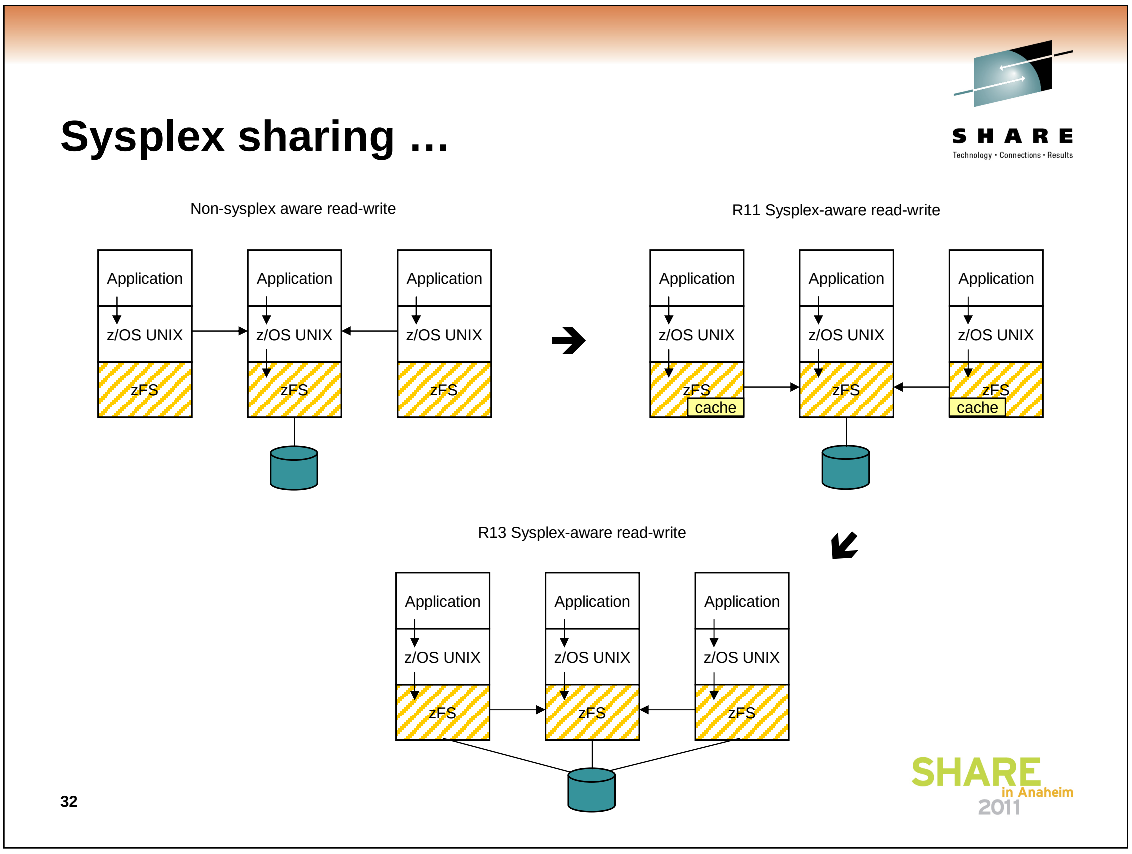
The romount_recovery=on IOEFSPRM configuration option says that if the log needs to be replayed and it is a R/O mount, then zFS will temporarily mount the file system R/W, replay the log and then unmount and mount the file system R/O. The default for romount_recovery is off. You can also dynamically set this option with **zfsadm config –romount_recovery on**.



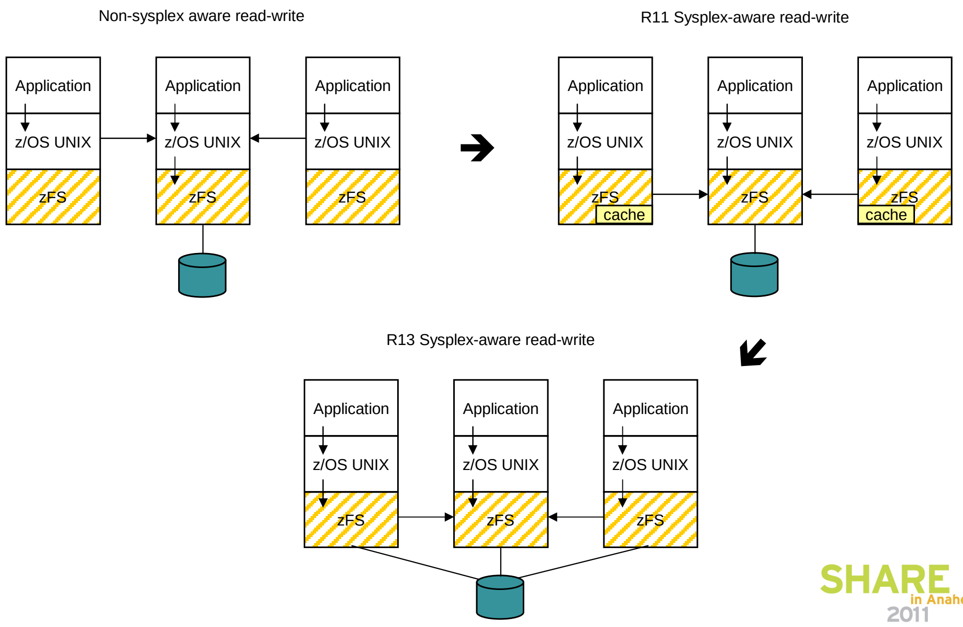
Sysplex sharing

- Both HFS and zFS support read-write sharing from multiple systems in a shared file system environment (BPXPRMxx SYSPLEX(YES))
  - HFS read-write file systems are always non-sysplex aware (z/OS UNIX always uses function shipping to a single owning system)
  - zFS read-write file systems can be sysplex-aware or non-sysplex aware
    For sysplex-aware read-write, z/OS UNIX sends requests to the local zFS and then
    - R11 zFS uses caching to sometimes avoid sending a read request to the owning system
    - R13 zFS can do direct I/O for reading and writing

31



Sysplex sharing …

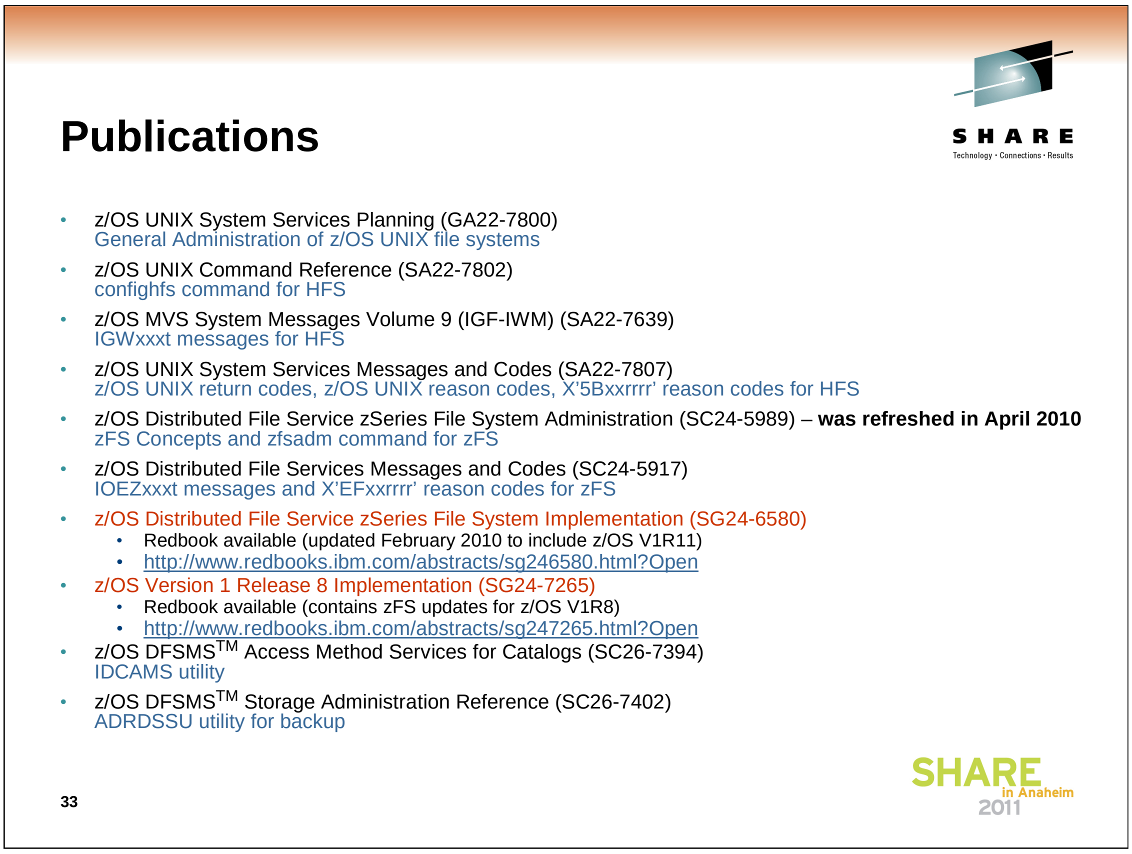


Non-sysplex aware read-write
Application
z/OS UNIX
zFS
R11 Sysplex-aware read-write
cache
R13 Sysplex-aware read-write

32

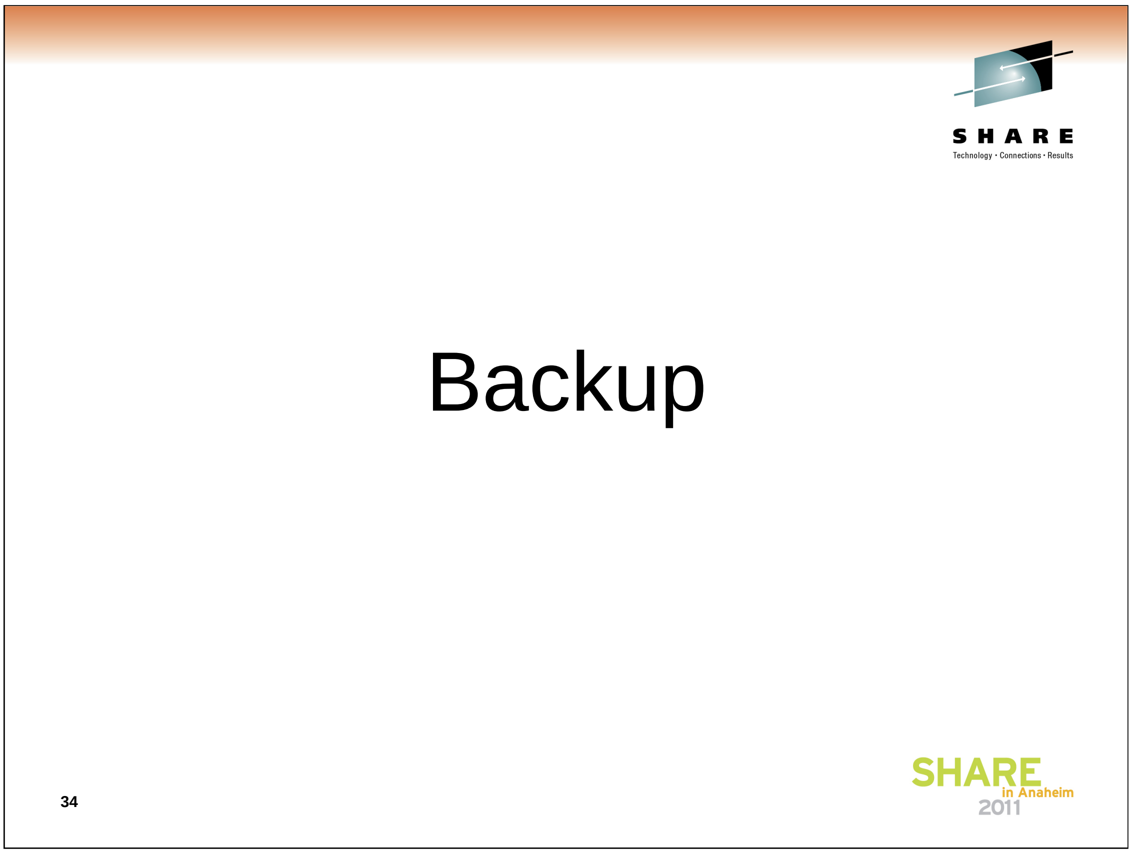
As of z/OS V1R11, zFS running in a shared file system environment supports sysplex-aware read-write file systems. Sysplex-aware read-write file systems can improve performance when the file system is accessed from multiple systems or when file systems require manual movement to optimize access performance. The preferred method is to specify IOEFSPRM sysplex=filesys. After all your systems are sysplex=filesys, then choose which zFS read-write file systems you want to be sysplex-aware and specify the RWSHARE MOUNT PARM. (sysplex=filesys requires R11 zFS APAR OA29619). See SHARE Session 2272 from Seattle 2010 for a full presentation on this zFS capability.

As of z/OS V1R13, zFS support for read-write sysplex-aware file systems is enhanced to directly access zFS user data from all R13 systems. Metadata updates are still sent to the zFS owning system.



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) – **was refreshed in April 2010**
  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 2010 to include z/OS V1R11)
  - 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^{TM} Access Method Services for Catalogs (SC26-7394)
  IDCAMS utility
- z/OS DFSMS^{TM} Storage Administration Reference (SC26-7402)
  ADRDSSU utility for backup



Backup

34

File access

- File access commands and APIs for zFS are the same as HFS except for reason codes on failures
  - z/OS UNIX reason codes – X’0000rrrr’ to X’20FFrrrr’
    Documented in z/OS UNIX System Services Messages and Codes (SA22-7807) – these are common to HFS and zFS
  - HFS specific reason codes – X’5Bxxrrrr’
    Documented in z/OS UNIX System Services Messages and Codes (SA22-7807)
  - zFS specific reason codes – X’EFxxrrrr’
    Documented in z/OS Distributed File Service Messages and Codes (SC24-5917)
- The **bpxmtext** shell command can be used to display the meaning of zFS reason codes (as of z/OS V1R8) and z/OS UNIX reason codes

Directory access

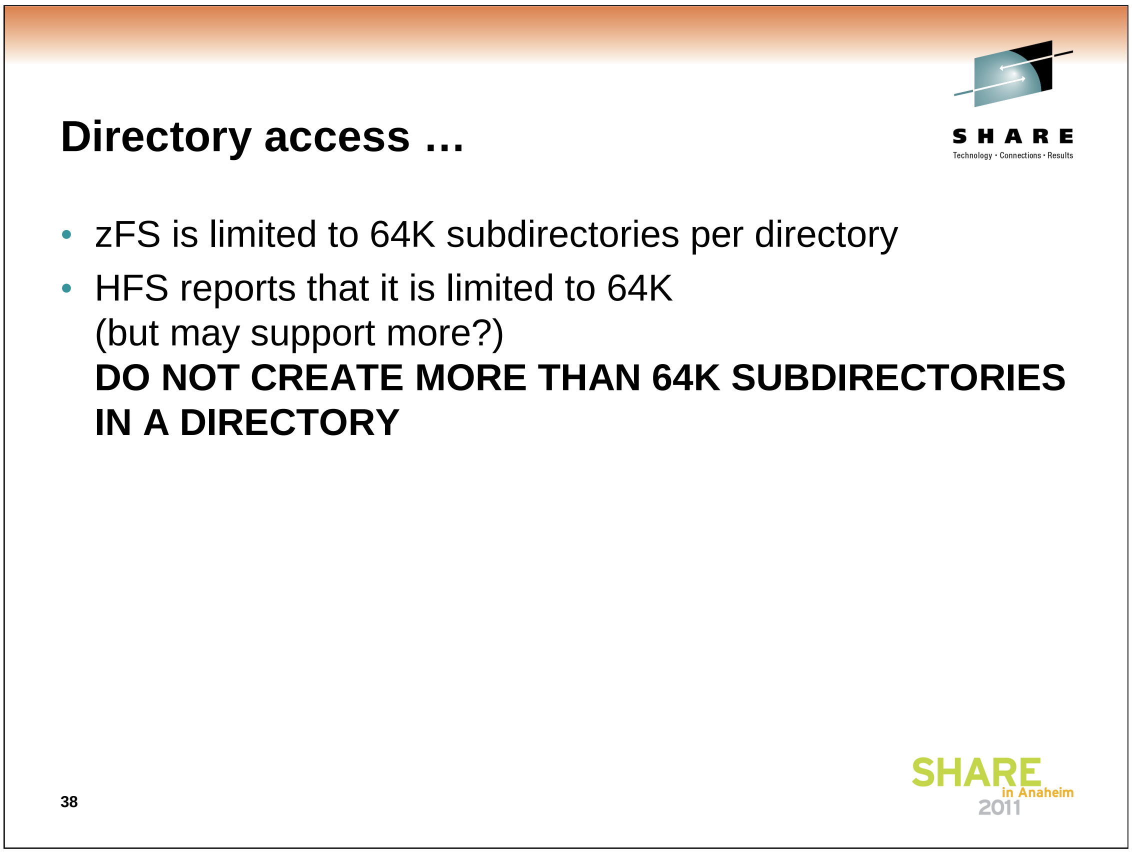
- Directory access commands and APIs for zFS are the same as HFS except for a few non-obvious situations
  - HFS returns names in a directory in (some) alphabetical order (using opendir, readdir, closedir APIs)
    DO NOT BECOME DEPENDENT ON THIS ORDER
    - This is not POSIX behavior
    - It is not controlled by localization envars (LC_COLLATE)
    - **ls** returns sorted names but that is because –C is the default
    - The order that HFS returns names is not the same as **ls**
      (ls -C returns uppercase characters first; HFS returns uppercase characters last.)
    - zFS returns names unsorted – as every other UNIX file system does

Directory access …

- zFS directories can be read as a file; HFS directories return 0 bytes (using open, read, close APIs)
  - You can see this by using the **strings** command against a directory

SHARE in Anaheim 2011

```
# cd /zfsmnt2
# df -v .
Mounted on     Filesystem                Avail/Total    Files      Status
/zfsmnt2       (PLEX.JMS.AGGR004.LDS0004) 1000/25920     4294967269 Available
ZFS, Read/Write, Device:27, ACLS=Y
AGGRGROW
File System Owner : DCEIMGVM    Automove=Y      Client=N
Filetag : T=off   codeset=0
Aggregate Name : PLEX.JMS.AGGR004.LDS0004
# ls -a
.          A          abc        acldir     file2      go.o       test1
test4.txt  test6.dat  test8.txt
..         ab         abcd       file1      file4      linkname   test3
test5.txt  test7.txt  testdir
# strings -n 1 -t d /zfsmnt2
       4 .
      13 ..
      23 test1
      36 test3
      49 test4.txt
      66 test5.txt
      83 test6.dat
     100 test7.txt
     117 test8.txt
     134 testdir
     149 linkname
     165 A
     174 ab
     184 abc
     195 abcd
     207 acldir
     221 go.o
     233 file1
     246 file2
     259 file4
```



Directory access …

- zFS is limited to 64K subdirectories per directory
- HFS reports that it is limited to 64K
  (but may support more?)
  DO NOT CREATE MORE THAN 64K SUBDIRECTORIES IN A DIRECTORY

38

SHARE in Anaheim 2011

```
# df -v /zfsmnt3
Mounted on     Filesystem                Avail/Total    Files      Status
/zfsmnt3       (PLEX.JMS.AGGR005.LDS0005) 2522566/2604960
4294967274 Available
ZFS, Read/Write, Device:28, ACLS=Y
NBS,NONBS
File System Owner : DCEIMGVM    Automove=Y      Client=N
Filetag : T=off   codeset=0
Aggregate Name : PLEX.JMS.AGGR005.LDS0005
# getconf LINK_MAX /zfsmnt3
65535
#
# df -v /
Mounted on     Filesystem                Avail/Total    Files      Status
/            (PLEX.CFCIMGVM.ROOT)      88/1440        4294967225
Available
HFS, Read/Write, Device:1, ACLS=Y
File System Owner : DCEIMGVM    Automove=Y      Client=N
Filetag : T=off   codeset=0
# getconf LINK_MAX /
65536
#
```