HFS to zFS Migration Support

Vivian Morabito

HFS to zFS Migration support

- In V2R3 IBM provides support to migrate HFS filesystems to zFS filesystems while they are still mounted. This resolves migration issues for customers who cannot unmount filesystems from their production systems.

- With this support IBM has resolved the last major roadblock for zFS migration

- The release after z/OS V2R3 is planned to be the last release of the operating system to support the HFS

HFS to zFS Migration support

- **bpxwmigf** command can be used to migrate a **mounted** HFS filesystems to zFS

- existing HFS to zFS migration tool BPXWH2Z, which allows migration of an HFS not in-use (i.e., not mounted HFS) is unchanged
  - This presentation will not discuss this existing tool

bpxwmigf command: HFS to zFS migration

- Available from:
  - z/OS Unix shell (/usr/sbin/bpxwmigf)
  - TSO/E
  - SYSREXX (console)

HFS to zFS migration

Migration occurs in two phases:

1. Data is mirrored and the active mirror is maintained until swap
   – Applications continue to run and are unaffected during this phase

2. The zFS file system is swapped in replacing the HFS
   – Access to the file system is briefly quiesced during the swap

These phases can be scheduled independent of one another if desired.

HFS to zFS migration

- All attributes except access time are replicated
- File FIDs and the devno are preserved
- System calls that result in updates to the HFS are replicated on the zFS under the same serialization when that update impacts something already mirrored
- Errors reading the HFS or writing zFS during migration cancel the migration
- Unmounting or moving ownership of the HFS cancels migration

Before starting the Migration

- All systems in the OMVS sharing group must be z/OS V2R3
- Review mount statements or policies for the file system being migrated to assure they are up to date and correct.
  - Do NOT update any mount statement to use the new zFS at this time.
- HFS file system must be mounted
- Migration must be initiated from the HFS owning system

Before starting the Migration…

- zFS filesystem must be allocated and formatted (must be a version 1.5)
- Ensure zFS filesystem is properly sized and empty
- **Do not** mount the zFS file system

What size do I make my target zFS?

- From *Distributed File Service zFS Administration:*

- "The number of storage blocks that are needed to store a zFS file system might not be exactly the same as the amount needed for HFS. For example, starting with z/OS V1R13, zFS uses 8 K blocks to contain small files; however, HFS uses 4 K blocks. In this case, some HFS file systems might need additional storage (possibly twice as much) when they are migrated to zFS."

- For more information about migrating data from HFS to zFS, see *z/OS Migration*.

bpxwmigf command: HFS to zFS migration

- Must be a superuser to use the bpxwmigf command to initiate, modify, or cancel a migration.
  - uid-0
  - READ authority to SUPERUSER.FILESYS.PFSCTL
- Regular user can query the status of the migration.

bpxwmigf command: HFS to zFS migration

bpxwmigf -source *sourcefs* [**-target** *targetfs*] [**-mode** rw|ro|asis]
[**-parm** *fsparms*|-noparm] [-**swap**|-**noswap**] [**-srename** *newname*|-nosrename]
[**-trename** *newtname*|-notrename] [**-wait** *minutes*] [**-status** *minutes*]
[**-priority** *priority*]

bpxwmigf -query [*sourcefs*] [-filter active|success|failed]

bpxwmigf -cancel *sourcefs*

bpxwmigf –source *sourcefs* **-swap**

bpxwmigf parameters

- **-source** <source HFS>
  - names the mounted file system to migrate or migration to modify

- **-target** <target zFS>
  - names the pre-allocated V1.5 zFS file system to replace the HFS

bpxwmigf parameters…

- **-mode** <rw|ro|asis>
  - sets the mode for the zFS after the swap
  - asis uses the hfs mode

- **-parm** <zFS parms> | -**noparm**
  - sets the zFS file system parms after the swap
  - Any specified parms are validated

- **-swap** | -**noswap**
  - initiate or do not initiate swap when mirroring is complete

bpxwmigf parameters…

- **-srename** <new HFS name> | **-nosrename**
- **Controls whether or not source HFS gets renamed after swap occurs**

- **-trename** [<new zFS name>]| **-notrename**
- **Controls whether or not targeted zFS gets renamed after swap occurs**

bpxwmigf parameters…

- **-wait** <minutes>
  - The migration command does not return until the specified number of minutes has elapsed or when the migration completes
- **-status** <minutes>
  - print status at specified interval while waiting
- **-priority** <1-9>
  - Controls the speed the files are copied from the source to target
  - A higher number slows the migration and decreases performance impact to applications using the file system during migration.
  - Default 2

bpxwmigf parameters

- Most parameters are optional
- All may be specified at migration initiation
  - Specify both –source & -target and any additional optional parameters
- All may be specified or modified any time before or when the swap occurs
  - Specify just the –source and any desired parameters

Usage & Invocation: **Initiate migrations**

- Migration is initiated when both source and target are specified

- **-source** <source HFS>
  - names the mounted file system to migrate or migration to modify
- **-target** <target zFS>
  - names the pre-allocated zFS file system to replace the HFS

- Additional parameters for the migration may be assigned at initiation
- *(parms can be added or modified by a subsequent bpxwmigf call before swap)*

Usage & Invocation: **Query migration status**

- **bpxwmigf -query** [*sourcefs*] [-filter active|success|failed]

- To query the status of a specific migration, specify the name of the source HFS
- To query the status of all migrations do not specify any source HFS names(s)
-
- Optional filtering is available when querying the status of all migrations.

- Note: Only one active migration is allowed at a time

Usage & Invocation: **swap**

- **bpxwmigf –source** *sourcefs* **–swap**

- Initiates swapping after mirroring complete.
- The zFS file system is swapped in, replacing the HFS
- Access to the file system is briefly quiesced during the swap

Usage & Invocation: **Cancel a migration**

- **bpxwmigf -cancel** *sourcefs*

- Cancels a migration in progress

-

- A cancelled migration may not be resumed. A new, empty zFS target will need to be re-created* and a new migration must be initiated.

- **either create a brand new zFS or format –overwrite the previously created zFS*

Example 1: Initiate migration

- **Specify BOTH –source and –target**
  - POSIX.HFS.EXAMPLE is initially mounted at /ict/MtPt
  - POSIX.ZFS.EXAMPLE (target zFS) has been created & is empty
  - -swap not specified ➔ mirroring continues until –swap is explicitly initiated

```
# bpxwmigf -source POSIX.HFS.EXAMPLE -target POSIX.ZFS.EXAMPLE
POSIX.HFS.EXAMPLE
  status.......: queued at 01:36:45 08/08/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 2


BPXWMG019I end of output
```

Example 1…

- Mirror phase complete, but no swap done since –swap was not specified
- Status indicates mirroring completed… migration will not be completed until new zFS is swapped in (mirroring continues until swap or cancel)

```
# bpxwmigf -query  POSIX.HFS.EXAMPLE
POSIX.HFS.EXAMPLE
  status.......: mirror complete at 01:38:20 08/08/2017 GMT
  started......: 01:36:45 08/08/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no


BPXWMG019I end of output
```

Example 1: SWAP

- Swap initiated independently, after mirroring complete:

```
# bpxwmigf -source POSIX.HFS.EXAMPLE -swap
POSIX.HFS.EXAMPLE
  status.......: swap initiated at 01:40:01 08/08/2017 GMT
  started......: 01:36:45 08/08/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:


BPXWMG019I end of output
```

Example 1: migration complete

- POSIX.ZFS.EXAMPLE is now mounted at /ict/MtPt
- Status shows completed migration

```
# bpxwmigf -query  POSIX.HFS.EXAMPLE
POSIX.HFS.EXAMPLE
  status.......: completed at 01:40:02 08/08/2017 GMT
  started......: 01.36.45 08/08/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:


BPXWMG019I end of output
#
```

Example 2: Initiate migration with auto-swap

- zFS automatically swapped in after mirroring complete.
- Initially POSIX.HFS.EXAMPLE2 mounted at /ict/MtPt2

```
# bpxwmigf -source POSIX.HFS.EXAMPLE2 -target POSIX.ZFS.EXAMPLE2 -swap
POSIX.HFS.EXAMPLE2
  status.......: mirroring 3% complete
  started......: 00:17:03 08/04/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE2
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: yes
  priority.....: 2


BPXWMG019I end of output
# df /ict/MtPt2
Mounted on      Filesystem                Avail/Total    Files       Status
/ict/MtPt2      (POSIX.ZFS.EXAMPLE2)      34110/72000    4294963981 Available
#
```

Filesystems are kept in-synch until swapping (or cancel)

- Mirroring completed on POSIX.HFS.EXAMPLE
  - *-swap was not specified when migration initiated*
- Updates subsequently made to POSIX.HFS.EXAMPLE…

```
# df /ict/MtPt
Mounted on     Filesystem                Avail/Total    Files      Status
/ict/MtPt      (POSIX.HFS.EXAMPLE)       114904/144000  4294963983 Available
# ls /ict/MtPt
dir_7   dir_71  dir_73  dir_75  dir_77  dir_79
dir_70  dir_72  dir_74  dir_76  dir_78
# echo 'Create a new file on source' > /ict/MtPt/A_New_File
# ls /ict/MtPt
A_New_File  dir_70      dir_72      dir_74      dir_76      dir_78
dir_7       dir_71      dir_73      dir_75      dir_77      dir_79
```

Filesystems are kept in-synch until swapping (or cancel)…

- …Are also updated in the target zFS continually until swap occurs.

```
# bpxwmigf -source POSIX.HFS.EXAMPLE -swap
```

```
# df /ict/MtPt
Mounted on      Filesystem                Avail/Total    Files      Status
/ict/MtPt       (POSIX.ZFS.EXAMPLE)       105358/144000  4294963980 Available
# ls /ict/MtPt
A_New_File  dir_70      dir_72      dir_74      dir_76      dir_78
dir_7       dir_71      dir_73      dir_75      dir_77      dir_79
#
```

-mode <rw|ro|asis>

- Default behavior is to mount the zFS filesystem with the same mode as the HFS
- zFS mount mode may be specified at initiation

bpxwmigf –source SRC.HFS -target TARGET.ZFS -mode rw

- zFS mount mode may be modified before swap occurs

bpxwmigf –source SRC.HFS -mode rw

Renaming filesystems

- Default behavior is no rename for either source or target
- Either or both filesystem names may be changed

Renaming filesystems

- **-srename** <new HFS name> | **-nosrename**
- **Controls whether or not source HFS gets renamed after swap occurs**
  - **-srename :** after swap rename the HFS to <new HFS name>
  - **-nosrename** : source hfs will not be renamed after swap

- **-trename** [<new zFS name>]| **-notrename**
- **Controls whether or not targeted zFS gets renamed after swap occurs**

- **-trename *new_zfs_name:*** new zFS is renamed to *new_zfs_name*
- **-trename**: The target zFS file system is renamed to the source HFS name. Optional, but if specified, -srename <new HFS name> must also be used to rename the original HFS.
- **-notrename:** zFS filesystem will not be renamed after swap

Rename the source HFS

- Default behavior is no rename
- at initiation:

  bpxwmigf –source SRC.HFS -target TGET.ZFS –srename SRC.HFS.OLD

- before swap occurs:

  bpxwmigf –source SRC.HFS –srename SRC.HFS.OLD

- When swap occurs:

  bpxwmigf –source SRC.HFS –srename SRC.HFS.OLD -swap

Rename the target zFS to the source HFS name

- You must change the source name

Example with renames set at migration initiation:

bpxwmigf –source MY.FILESYS –target MY.ZFS.MIGR –trename
-srename MY.OLD.FILESYS

Example with renames set at swap:

bpxwmigf –source MY.FILESYS –trename -srename MY.OLD.FILESYS -swap

Rename just the target zFS

bpxwmigf –source BUSY.HFS -target MY.MIGR.ZFS –trename BUSY.ZFS

Specify zFS mount parm

- **-parm** <zFS parms> | -**noparm**
  - sets the zFS file system parms after the swap
  - Any specified parms are validated

  - Supported zFS mount parms are:
    - aggrfull(threshold,increment)
    - aggrgrow | noaggrgrow
    - rwshare | norwshare

zfs mount parm: aggrfull

- Specifies the threshold and increment for reporting aggregate utilization messages to the operator.

- May be set to:

  threshold,increment or

  OFF

zfs mount parm: aggrgrow

- Specifies whether aggregates can be dynamically extended when they become full
- may be set ON or OFF
- Aggregate must have a secondary allocation specified (when created) and there must be space on the volumes.

zfs mount parm: RWSHARE and NORWSHARE

- Both are read / write mounts
- RWSHARE is “sysplex-aware”
  - Filesystem is locally mounted on every member of the sysplex
  - File requests are handled locally by zFS
- NORWSHARE is “sysplex-Unaware”
  - File operations must be “function-shipped” to OMVS owning system

zFS mount parms example

bpxwmigf –source MY.FILESYS –target MY.ZFS.MIGR –parm RWSHARE

(RWSHARE mounts must be read/write!)

bpxwmigf -priority

- **priority** <1-9>
  - Controls the speed the files are copied from the source to target
  - A higher number slows the migration and decreases performance impact to applications using the file system during migration.
  - Default is 2

For example, to slow an in-progress migration down:

bpxwmigf –source SRC.HFS1 –trename TGT.ZFS1 –priority 9

bpxwmigf: example -wait *minutes* –status *minutes*

- **-wait minutes:**
- The migration command does not end until the specified number of minutes has elapsed or when the migration completes

- **-status minutes:**
- Prints status every at interval specified by minutes

- The example on the following screens demonstrate the bpxwmigf returning after a wait time of 5 minutes, printing status every 1 minute.

- The mirroring continues after the bpxwmigf command returns, and synchronization continues until swap

bpxwmigf: example -wait *minutes* –status *minutes*

```
# date; bpxwmigf -source POSIX.HFS.EXAMPLE3 -target POSIX.ZFS.EXAMPLE3      -pri
ority 9 -wait 5 -status 1; date
Mon Jul 24 21:26:07 EDT 2017
POSIX.HFS.EXAMPLE3
  status.......: mirroring 1% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


POSIX.HFS.EXAMPLE3
  status.......: mirroring 9% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


POSIX.HFS.EXAMPLE3
  status.......: mirroring 17% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
 ===>
                                                                      MORE...
```

bpxwmigf: example -wait *minutes* –status *minutes…*

```
  mount parms..:
  auto-swap....: no
  priority.....: 9


POSIX.HFS.EXAMPLE3
  status.......: mirroring 24% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


POSIX.HFS.EXAMPLE3
  status.......: mirroring 31% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


POSIX.HFS.EXAMPLE3
  status.......: mirroring 36% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
 ===>
                                                                MORE...
```

bpxwmigf: example -wait *minutes* –status *minutes…*

```
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


BPXWMG019I end of output
Mon Jul 24 21:31:07 EDT 2017
#
#
# bpxwmigf -query  POSIX.HFS.EXAMPLE3
POSIX.HFS.EXAMPLE3
  status.......: mirroring 40% complete
  started......: 01:26:07 07/25/2017 GMT
  user.........: SUIMGKA
  target name..: POSIX.ZFS.EXAMPLE3
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no
  priority.....: 9


BPXWMG019I end of output
#
```

bpxwmigf parameters

- Multiple parameters may be specified on a single bpxwmigf call

- For example:

```
bpxwmigf -source POSIX.HFS.EXAMPLE4  -target POSIX.ZFS.TOMIG
        -parm  RWSHARE -priority 6 -mode rw
```

bpxwmigf parameters

- Parameters may be modified any time before swap occurs.
- bpxwmigf query will always show the current settings

Before:

```
# bpxwmigf -query  POSIX.HFS.EXAMPLE5
POSIX.HFS.EXAMPLE5
  status.......: mirror complete at 15:11:11 08/04/2017 GMT
  started......: 15:11:06 08/04/2017 GMT
  user.........: SUIMGKB
  target name..: POSIX.ZFS.MIG5
  source rename: no
  rename target: no
  mount mode...: same
  mount parms..:
  auto-swap....: no

BPXWMG019I end of output
```

bpxwmigf -source POSIX.HFS.EXAMPLE5 -mode rw -parm RWSHARE -srename POSIX.HFS.OLD5

After:

```
# bpxwmigf -query  POSIX.HFS.EXAMPLE5
POSIX.HFS.EXAMPLE5
  status.......: mirror complete at 15:11:11 08/04/2017 GMT
  started......: 15:11:06 08/04/2017 GMT
  user.........: SUIMGKB
  target name..: POSIX.ZFS.MIG5
  source rename: POSIX.HFS.OLD5
  rename target: no
  mount mode...: rw
  mount parms..: RWSHARE
  auto-swap....: no

BPXWMG019I end of output
#
```

Querying

bpxwmigf -query HFS.NAME

- Queries the status of a **specific** migration (for example HFS.NAME)
- indicates percent complete, based on the ratio of byte sizes of the target/source
- Shows parameters in effect
- Show start time of migration

Possible migration statuses

- Mirroring
- Mirroring complete
- Swap initiated
- Swap deferred
- Queued
- Cancelled
- Cancelling
- Completed
- Failed

Query

- **bpxwmigf -query** [-filter active|success|failed]

- When no HFS name is specified, queries the status of all migrations.
-
- Optional filtering is available when querying the status of all migrations.

Querying migration status

- HFS (specified by –source) must still be mounted to report status.
  - If it has been unmounted the query will fail since it can no longer find the migration to report.
  - the syslog still will contain information

SYSLOG example

- BPXU006I FILESYSTEM MIGRATION WAS STARTED
- SOURCE POSIX.HFS.EXAMPLE
- TARGET POSIX.ZFS.EXAMPLE
- BPXU007I FILESYSTEM MIRROR COMPLETE AND WAITING FOR SWAP.
- .
- .
- .
- IOEZ00044I Aggregate POSIX.ZFS.EXAMPLE attached successfully.
- BPXU008I FILESYSTEM MIGRATION IS COMPLETE
- OLD POSIX.HFS.EXAMPLE
- NEW POSIX.ZFS.EXAMPLE

Cancelling a migration

- May be explicitly cancelled using bpxwmigf

  bpxwmigf –cancel *sourcefs*

- Shutdown of OMVS on the HFS owning LPAR will also cancel the migration

- Changing owner of the HFS source filesystem ownership cancels the migration

Cancelling a migration…

- If bpxwmigf is canceled or aborted (by a change in ownership etc) the migration will always be cancelled.

- Ensure that the mount statements or mount policies will not mount the new zFS file system after a system restart if the migration is canceled.

  – zFS filesystem target will not remain synchronized with HFS source after a cancel.

  – Therefore, do not use the target zFS filesystem after a cancel… rather re-initiate a migration

bpxwmigf from the operators console

- All bpxwmigf commands may be entered from the console using SYSREXX

- Note: all functionality of bpxwmigf is available from the console via SYSREXX, but this interface is probably most useful for cancelling or querying migration status.

bpxwmigf from the operators console….

```
- f axr,bpxwmigf -query MY.HFS
- IRR8131I NO PROFILE WAS FOUND IN THE STARTED CLASS FOR
-         AXR04 WITH JOBNAME AXR04. RACF WILL USE ICHRIN03.
  $HASP100 AXR04    ON STCINRDR
- $HASP373 AXR04    STARTED
0  MY.HFS
     status.......: mirror complete at 21:17:37 08/02/2017 GMT
     started......: 21:17:37 08/02/2017 GMT
     user.........: IBMUSER
     target name..: MY.ZFS
     source rename: no
     rename target: no
     mount mode...: same
     mount parms..:
     auto-swap....: no
```

After the Migration is successfully completed:

- Update mount statements and policies to mount the newly migrated zFS

- mount statements / mount policies should not be updated until successful completion, to avoid the possibility that zFS target from a cancelled migration (either due to explicit cancel, HFS ownership change, or shutdown) is mounted upon a subsequent remount

HFS to zFS Migration Performance

- “Official” IBM Performance Measures
- Your mileage may vary
- Measurements were taken on a IBM z12 (model 2827) with 4 CPUs

HFS to zFS Migration Performance

#Dirs * #Files * Size	1 * 10* 100MB	1 * 20 * 100MB	100 * 1000 * 50
Mirror Elapsed Time (sec)	14	28	76
Mirror CPU (sec) (OMVS/zFS)	0.72 / 0.10	1.45 / 0.20	15.21 / 0.38
Mirror I/Os (OMVS/zFS)	0 / 244740	0 / 488880	0 / 16709
Swap CPU (sec) (OMVS/zFS)	0.00 / 0.02	0.00 / 0.02	0.06 / 0.12
Swap I/Os (OMVS/zFS)	0 / 58	0 / 622	0 / 93

HFS to zFS Migration Performance

#Dirs * #Files * Size	200 * 1000 * 50	500 * 1000 * 50
Mirror Elapsed Time (sec)	110	395
Mirror CPU (sec) (OMVS/zFS)	27.28 / 0.91	81.73 / 2.38
Mirror I/Os (OMVS/zFS)	0 / 39452	0 / 104296
Swap CPU (sec) (OMVS/zFS)	0.13 / 0.15	3.50 / 0.15
Swap I/Os (OMVS/zFS)	0 / 2618	0 / 981

Apply APAR OA56827

- **Description:**
- If the source HFS file system had any active child file systems mounted at the time of migration, the entry in the OMVS couple data set (BPXMCDS) for each child file system is corrupted after the migration.

- There is no external indication of this corruption at the time of the migration. The corruption becomes evident when any LPAR at any z/OS release level is IPLed (new system or re-IPL of existing system) in the shared file system environment following the BPXWMIGF migration. The catchup mounts for the affected child file systems will fail on the system being IPLed.

APAR OA56827 …

- **Recommended Actions:**

- Do NOT use the BPXWMIGF utility to migrate any HFS file system with active child file systems mounted under it until the fix for APAR OA56827 is installed on all systems in the shared file system environment.

- Without the fix for APAR OA56827 installed, the BPXWMIGF utility can still be used for migration of HFS file systems that do NOT have active child file systems mounted.

- If you have already used the BPXWMIGF utility to migrate HFS file systems with active child file system mounts, it is recommended that you avoid IPLing any systems in the shared file system environment until the corruption in the OMVS couple data set (BPXMCDS) has been recovered or until the fix has been installed on all systems. Please contact z/OS UNIX support for assistance.

HFS to zFS migration – Summary

- Only one active migration at a time
- Migration must be initiated from the HFS owning system

HFS to zFS migration – Summary (cont'd)

- Mirror and swap phases can be scheduled independently
  - *If you want to control when the swap occurs, issue swap independently*
- After migration, ensure mount statements or policies will mount the zFS filesystem
- BPXWMIGF is available via APAR OA53128
- Apply APAR OA56827 before migrating a filesystem with active child file systems mounted

HFS to zFS migration

Questions?

Backup and recover of individual z/OS UNIX files

- z/OS V2R3 now provides backup support **on a file basis**

- V2R3 APARs required for the HSM backup support on a file basis.
  - OA52836 – DFSMSdss
  - OA52703 – DFSMShsm
  - OA56145 – ZFS
  - OA54218 – USS
  - OA55165 – RACF

- Doc is here: http://publibz.boulder.ibm.com/zoslib/pdf/OA52703.pdf

OA56145: zFS File Snapshot API

- Creates a point-in-time copy of a file in a zFS filesystem
- Allows subsequent read request from that snapshot along with concurrent reads and writes to the actual file on-disk.
- When a snapshot is created, backup programs an also request information about the file, which will help determine if the file was changed since the last backup
- API is a w_ioctl (BPX1IOC) call which specifies a file **descriptor**
- Requires USS APAR OA54218

OA56145: zFS File Snapshot API

- New output fields in FSINFO / FILEINFO displays has been updated to indicate if there is a current backup underway in the filesystem and the backup progress on an individual file

- File snapshot cannot be used:
  - while the containing filesystem is encrypting, decrypting, compressing, decompressing, or shrinking.
  - if the containing filesystem is a Version 1.4
  - on a file that is stored in compressed format

OA56145: zFS File Snapshot API …

- **Security Considerations:**
  - The user must have lookup authority (x) to the directory and READ authority (r) to the file
  - The caller must be an authorized program

- **Migration / Coexistance:**
  - No migration concerns
  - Writes to files undergoing backup from systems without APAR OA56145 will fail