Taming the Beast – Best Practices for zFS with CICS

Phil_Wakelin@uk.ibm.com

CICS Strategy & Design, IBM Hursley UK
Disclaimer

IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM’s sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including 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 results similar to those stated here.
Why zFS?
The Problem...
Objective
A place for everything
Types of CICS zFS files

Install
- CICS stuff provided by IBM
- CICS Programs and Samples
- May change during APARs
Types of CICS zFS files

Install
- CICS stuff provided by IBM
- CICS Programs and Samples
- May change during APARs

Critical
- Things that define your applications and environments
- Programs, Properties and Config files etc
- Should be considered critical and be recoverable
Types of CICS zFS files

Install
- CICS stuff provided by IBM
- CICS Programs and Samples
- May change during APARs

Critical
- Things that define your applications and environments
- Programs, Properties and Config files etc
- Should be considered critical and be recoverable

Temp
- Things that applications create or write to at runtime
- Non critical, messages and logs files etc
Where should they go?

Install
- Default: /usr/lpp/cicsts52
- SIT Parm: USSHOME

Critical
- Suggest using /var/cicsts/...
- May have sub directories per CICSPlex or region
- Use directories to group common assets eg all web files

Temp
- Suggest using a separate place eg /cicslogs/<APPLID>...
- Typically have one per region
- May need to clean out periodically
Setup for Install Dir

Create release specific directory structure

- `/usr/lpp/cicsts` is created ONCE for all releases (read only)
- Depending on SMP/E target zone structure
- You may create variants of `/cicsts52`

```
/pathprefix
  /usr1  /usr2  /usr3  /usrn
    /lpp
    /cicsts
      /cicsts41  /cicsts42  /cicsts51  /cicstsnn
```
Setup for Install Dir

- Directories under /cicsts52
  - docs
  - IBM
  - JVMPROFILES
  - lib
  - pipelines
  - samples
  - schemas
  - ...
Setup for Install Dir

- SITParms
  - USSHOME
    - The name and path of the root directory for CICS files
    - Default: /usr/lpp/cicsts/cicsts52
Setup for Critical Dir

- Create data set for usage as `/var/cicsts`
- If sharing zFS across a sysplex
  - Mount data set onto root filing system as `/cicsts` as a r/w filing system
  - On each LPAR create symbolic link to link `/var/cicsts` to `/cicsts` (`/var` is always a symlink to `<LPAR>/var`)
    - `ln -s /cicsts /var/cicsts`
- If not sharing zFS
  - Mount data set onto `/var` as `/var/cicsts`
Setup for Critical Dir

- **SIT Parms**
  - **USSCONFIG**
    - The name and path of the directory for CICS config files
    - Default: /var/cicsts/dfhconfig
  - **JVMPROFILEDIR**
    - The directory name for the JVM profiles
    - Suggest: /var/cicsts/jvmprofiles
    - Sample profiles DFHJVMAX and DFHOSGI must be copied to here (and edited) if you want to use them
Setup for Temp Dirs

- Give a z/OS UNIX User ID to each CICS region user ID
- Set up a Temp Dir on zFS for each of your CICS regions (eg /cicslogs/<APPLID>)
  - Each Temp Dir should be a separate filing system each backed by their own dataset
  - Give each region's User ID r/w permissions to their directory
- Choose a z/OS UNIX Group ID (GID) for the RACF group, and assign the it to the RACF Group
- Make sure that each CICS region user ID connects to the RACF group that you choose
Everything
in its place
CICS and zFS usage survey: types of resources

Q2. What CICS application resources do you use today in HFS/zFS?

- 35% Web services
- 35% Java
- 10% Static Web content
- 10% Bundles
- 10% N/A

CICS TS 5.1 introduces Application and Platform resources which are packaged as bundles on zFS…
DOCTEMPLATEs, URIMAPs and TCPIPSERVICES

- **DOCTEMPLATE** attribute
  - **HFSFILE**
    - zFS file containing the template (when it resides in zFS)

- **URIMAP** attribute
  - **HFSFILE**
    - zFS file that forms the body of a static response to an HTTP request from a Web Client

- **TCPIPSERVICE** attribute
  - **CIPHERS**
    - zFS file that contains the SSL cipher suite specification
DOCTEMPLATEs, URIMAPs and TCPIPSERVICES

- **DOCTEMPLATE** attribute
  - **HFSFILE**
  - zFS file containing the template (when it resides in zFS)

- **URIMAP** attribute
  - **HFSFILE**
  - zFS file that forms the body of a static response to an HTTP request from a Web Client

- **TCPIPSERVICE** attribute
  - **CIPHERS**
  - zFS file that contains the SSL cipher suite specification
  - Relative to USSCONFIG

- Put all of these in the Critical Dir!
WEBSERVICEs

- **WEBSERVICE** attributes
  - **WSBIND**
    - zFS file name of web service binding file
  - **WSDLFILE**
    - zFS file name of the WSDL file
  - **ARCHIVEFILE**
    - zFS file name of the zip file that contains the WDSL
WEBSERVICEs

- **WEBSERVICE** attributes
  - **WSBIND**
    - zFS file name of web service binding file
  - **WSDLFILE**
    - zFS file name of the WSDL file
  - **ARCHIVEFILE**
    - zFS file name of the zip file that contains the WDSL

- Put all of these in the Critical Dir!
PIPELINEs

- **PIPELINE** attributes
  - **CONFIGFILE**
    - ZFS file that defines processing nodes
  - **SHELF***
    - Directory for CICS to store installed artifacts (may be shared by multiple CICS Regions)
  - **WSDIR**
    - Directory for WSDL and WSBIND files that are to be installed into CICS
PIPELINEs

- PIPELINE attributes
  - CONFIGFILE
    - ZFS file that defines processing nodes
  - SHELF*
    - Directory for CICS to store installed artifacts (may be shared by multiple CICS Regions)
  - WSDIR
    - Directory for WSDL and WSBIND files that are to be installed into CICS

- Put all of these in the Critical Dir!

*Recovery of shelf directory should be considered critical resource for WARM restart if using a CSD defined WEBSERVICE (as opposed to a pipeline scan)
ATOMSERVICEs

- **ATOMSERVICE attributes**
  - **CONFIGFILE**
    - zFS file name that describes meta data for the feed
  - **BINDFILE**
    - ZFS file that describing XML transformation for the feed
ATOMSERVICEs

- **ATOMSERVICE** attributes
  - **CONFIGFILE**
    - zFS file name that describes meta data for the feed
  - **BINDFILE**
    - ZFS file that describing XML transformation for the feed

- Put all these in the Critical Dir!
BUNDLEs

- **BUNDLE** attribute
  - **BUNDLEDIR** – zFS path to bundle directory
BUNDLEs

- **BUNDLE** attribute
  - **BUNDLEDIR** – zFS path to bundle directory

- Critical Dir again!
Java Resources

- **PROGRAM** attribute
  - **JVMCLASS**
    - Name of service defined in OSGI bundle, in a CICS Bundle
    - Contained in BUNDLEDIR on the BUNDLE

- **JVMSERVER** attribute
  - **JVMPROFILE**
    - zFS file name of the JVMPROFILE
    - Relative to the SIT parm JVMPROFILEDIR
      - If the JVMSERVER is installed as a part of a bundle, JVMPROFILE is relative to BUNDLEDIR on the BUNDLE
Java Resources

- **PROGRAM** attribute
  - **JVMCLASS**
    - Name of service defined in OSGI bundle, in a CICS Bundle
    - Contained in BUNDLEDIR on the BUNDLE
  - **JVMSERVER** attribute
    - **JVMPROFILE**
      - zFS file name of the JVMPROFILE
      - Relative to the SIT parm JVMPROFILEDIR
        - If the JVMSERVER is installed as a part of a bundle, JVMPROFILE is relative to BUNDLEDIR on the BUNDLE
- Critical Dir...
You should copy sample JVMPROFILEs to the directories under /var/cicsts if you want to use them.
JVMPROFILEs

- JVMPROFILE properties
  - $JAVA_HOME – zFS path to jre install
  - $WORK_DIR – zFS path to logs and dumps

- JVMPROFILE (Liberty) properties
  - $WLP_INSTALL_DIR – install location wlp
  - $WLP_USER_DIR – zFS path to server.xml
  - $WLP_OUTPUT_DIR – Liberty logs and installedApps
JVMPROFILEs

- **JVMPROFILE** properties
  - `$JAVA_HOME` – zFS path to jre install
  - `$WORK_DIR` – zFS path to logs and dumps

- **JVMPROFILE (Liberty)** properties
  - `$WLP_INSTALL_DIR` – install location wlp
  - `$WLP_USER_DIR` – zFS path to server.xml
  - `$WLP_OUTPUT_DIR` – Liberty logs and installedApps
JVMPROFILE suggestions

- **JVMPROFILE properties**
  - `$JAVA_HOME` – zFS path to jre install
  - `$WORK_DIR` – Use CICS Temp Dir: /cicslogs/&APPLID;

- **JVMPROFILE (Liberty) properties**
  - `$WLP_INSTALL_DIR` – Use USSHOME: &USSHOME;/wlp
  - `$WLP_USER_DIR` – Use JVMPROFILEDIR: /&CONFIGROOT;
  - `$WLP_OUTPUT_DIR` – Use /cicslogs/&APPLID;/&JVMSERVER;
JTA Logs in server.xml

- If you use JTA in Liberty, CICS needs r/w access to the JTA transaction logs.
- But they are critical for WARM restarts of CICS so should be recoverable
- Where do we put them?...
JTA Logs in server.xml

- If you use JTA in Liberty, CICS needs r/w access to the JTA transaction logs.
- But they are critical for WARM restarts of CICS so should be recoverable
- Where do we put them?...
- Their own recoverable filing system under /cicsjta/...

```
```
CICS Platforms

- With the new CICS Platforms we've made it easy...
CICS Platforms

- With the new CICS Platforms we've made it easy...
CICS Platforms

- Explorer is aware of the platform home directory and takes care of the rest for you!
## Summary Table

<table>
<thead>
<tr>
<th>Resources</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOCTEMPLATE</td>
<td>HFSFILE</td>
</tr>
<tr>
<td>URIMAP</td>
<td>HFSFILE</td>
</tr>
<tr>
<td>TCPIPSERVICE</td>
<td>CIPHERS</td>
</tr>
<tr>
<td>WEBSERVICE</td>
<td>WSBIND</td>
</tr>
<tr>
<td></td>
<td>WSDLFILE</td>
</tr>
<tr>
<td></td>
<td>ARCHIVEFILE</td>
</tr>
<tr>
<td>PIPELINE</td>
<td>CONFIGFILE</td>
</tr>
<tr>
<td></td>
<td>SHELF</td>
</tr>
<tr>
<td></td>
<td>WSDIR</td>
</tr>
<tr>
<td>ATOMSERVICE</td>
<td>CONFIGFILE</td>
</tr>
<tr>
<td></td>
<td>BINDFILE</td>
</tr>
<tr>
<td>PROGRAM (Java)</td>
<td>JVMCLASS</td>
</tr>
<tr>
<td>JVMSERVER</td>
<td>JVMPROFILE</td>
</tr>
<tr>
<td>JVMPROFILE</td>
<td>$JAVA_HOME</td>
</tr>
<tr>
<td></td>
<td>$WORK_DIR</td>
</tr>
<tr>
<td></td>
<td>$WLP_INSTALL_DIR</td>
</tr>
<tr>
<td></td>
<td>$WLP_USER_DIR</td>
</tr>
<tr>
<td></td>
<td>$WLP_OUTPUT_DIR</td>
</tr>
<tr>
<td>server.xml</td>
<td>transactionLogDirectory</td>
</tr>
<tr>
<td>BUNDLE</td>
<td>BUNDLEDIR</td>
</tr>
<tr>
<td>CICS Platforms</td>
<td>Platform Home</td>
</tr>
</tbody>
</table>
Secure access
Who needs access to what?

<table>
<thead>
<tr>
<th>Directory</th>
<th>CICS Region</th>
<th>Admin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install Dir</td>
<td>read</td>
<td>read</td>
</tr>
<tr>
<td>Critical Dir</td>
<td>read</td>
<td>read/write</td>
</tr>
<tr>
<td>Temp Dir</td>
<td>read/write</td>
<td>read/write</td>
</tr>
</tbody>
</table>

• Both user types would also have execute permissions on directories (not required for files)
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS

Here’s an example of an entry you might see if you listed contents of a zFS directory

```
  drwxr-x---  2 SYSADMIN  CICS  8192 May 10 14:52 MyBundle/
```

CICS zFS files may require access via 3 classes of user:

- CICS regions - region userid
- CICS system administrators (humans)
- Code management systems (tools)
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS

Is a directory (called MyBundle)

```
drwxr-x--- 2 SYSADMIN CICS 8192 May 10 14:52 MyBundle/
```
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS

- Owned by user `sysadmin` and by group `CICS`
- Is a directory (called MyBundle)

```
drwxr-x---  2 SYSADMIN  CICS     8192 May 10 14:52 MyBundle/
```
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS.

- User SYSADMIN has read, write, execute permissions
- Owned by user sysadmin and by group CICS
- Is a directory (called MyBundle)

```
drwxr-x---  2 SYSADMIN  CICS  8192 May 10 14:52 MyBundle/
```
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS.

User SYSADMIN has read, write, execute permissions.

Any user in group CICS has read and execute permissions.

Owned by user sysadmin and by group CICS.

Is a directory (called MyBundle).

drwxr-x--- 2 SYSADMIN CICS 8192 May 10 14:52 MyBundle/
File permissions in zFS

Use the UNIX permission flags for Owner, Group and All to control access to your CICS resources on zFS.

- **User SYSADMIN** has read, write, execute permissions.
- Any user in group CICS has read and execute permissions.
- Owned by user sysadmin and by group CICS.
- Is a directory (called MyBundle).
- Anyone else has no permissions.

```
drwxr-x--- 2 SYSADMIN CICS 8192 May 10 14:52 MyBundle/
```
UMASK

- The file permission bits are set using the UMASK of the creating process, which signifies the bits that are not set
  - i.e. a umask of 022 causes
    - Directories to be created with 755 (rwxr-xr-x) permissions
    - Files to be created with 644 (rw-r--r--) –
      - By default x permissions are not given for files
Example Setup for Critical Dir

- Set permissions of /var/cicsts to allow access by multiple readers (CICS regions) and a common writer (administrator)
  - 1. Set the owner to have read/write/execute, this will be the userid required by zFS to export files into zFS
  - 2. Set the readers to have read/execute access
     - > chgrp –R <group> /cicsts
     - > chmod –R 750 /cicsts
  - 3. Set default file permission for the FTP daemon to give writers(owners) rw and readers(group) r
     - i.e UMASK 027
     - see SYS1.TCPPARMS(FTPDATA)
     - Used by CICS Explorer “Export Bundle Project to z/OS USS” process
Multiple writer's problem

- A user can be in many groups, but a file has only one group permission
  - Meaning that if multiple users need to access the file they must be in that group, and will all share the same permissions
  - This means 2 logical groups of users (such as system admins and CICS regions) can not use UNIX permission bits to be granted access

- ACLs provide a solution to this as they allow a more flexible model
  - Multiple groups can have file permissions
  - ACL inheritance can be controlled
  - However, they may only restrict the access permissions that are defined by the UNIX permissions bits
  - RACF CLASS FSSEC
  - Control using setfacl USS command
Performance of zFS

- Performance of shared zFS mounted r/w filesystems has been regarded as an issue (in terms of XCF signalling costs) and function shipping of I/O between LPARs
  - Often a problem on JVM startup due to loading of JARs from zFS and writing of info to OSGi cache directory (under WORK_DIR)

- Solutions:
  - V1R11 provides local read caching – removing overheads for reads
  - V1R13 provides direct I/O for read and write, removing need to function ship these commands to the owning LPAR
  - Or mounting file system locally removes need to function ship I/O
  - JVM class caching provides ability to cache Java byte codes in a shared memory area (i.e within LPAR)
    - Requires APAR PM78799 on CICS TS V4.2 to support class caching
Manage Change
zFS usage survey...

Where do you store the master copy of CICS USS files?

- HFS/zFS: 50%
- Workstation: 33%
- SCM: 17%
- N/A: 0%
Managing changes to CICS Bundles

CICS Bundle XML should be treated as source code.
Changes should be managed and shared using a source code management (SCM) repository.

The CICS Explorer should be used to make changes to the Bundles and deploy them to zFS.
Migrating CICS Bundles from Dev to Test to Production

- BUNDLEs should be treated like any other CICS resource that has a reference to an artefact that lies outside the CSD eg:
  - PROGRAMS have load modules/java classes
  - WEBSERVICEs have wsbind files

- You should migrate the CICS Bundle XML before the BUNDLE resource

- You wouldn’t migrate a new PROGRAM resource before you migrated the load module for it!
Migrating CICS bundle XML from Dev to Production

1. Build bundle project from SCM
2. Deliver changes
3. Promote

Dev SCM Stream

Test SCM Stream

Promote

Dev 1
Dev 2

CICS Explorer

Install Bundle resource

Export Bundle to zFS

Dev Region

Dev zFS

Dev Region

Dev zFS

Automation system

Install Bundle resource

Transfer bundle to zFS

Test Regions

Test zFS

Once tests pass ok, migrate (copy) bundle from test to production zFS

Production Regions

Production zFS
Migrating CICS bundle XML from Dev to Production

- **Dev SCM Stream**
  - Build bundle project from SCM
  - Deliver changes
  - Install Bundle resource
  - Export Bundle to zFS
  - Install resource
  - Export bundle to zFS

- **Test SCM Stream**
  - Promote
  - Deliver changes
  - Install Bundle resource
  - Export bundle to zFS
  - Install resource
  - Export bundle to zFS

- **CICS Build Toolkit**
  - Build bundle project from SCM
  - Automation system
  - Transfer bundle to zFS
  - Install Bundle resource

- **DFHDPLOY**
  - Install Bundle resource
  - Once tests pass ok, migrate (copy) bundle from test to production zFS

- **Dev Region**
  - Dev zFS
  - Test zFS
  - Production Regions
  - Production zFS
Migrating bundles

Option 1: Modify BUNDLEDIR to new zFS location

Option 2: Use Symlinks to point to the real bundle location
Option 3: Use zFS mounts to migrate bundles
Option 3: Use zFS mounts to migrate bundles

- **Dev CSD**
  - BUNDLE
  - CICS Bundle DIR
  - Promote

- **Test CSD**
  - BUNDLE
  - CICS Bundle DIR
  - Promote

- **Prod CSD**
  - BUNDLE
  - CICS Bundle DIR
  - Promote

- **Dev zFS**
  - CICS Bundle DIR
  - Mount
  - Unmount

- **Test zFS**
  - CICS Bundle DIR
  - Mount
  - Unmount

- **Prod zFS**
  - CICS Bundle DIR
  - Mount
  - Unmount

- **zOS Datasets**
  - CICS.BUNDLES.MYBUNDLE.V3
  - Mount
  - Unmount

  - CICS.BUNDLES.MYBUNDLE.V2
  - Mount
  - Unmount

  - CICS.BUNDLES.MYBUNDLE.V1
  - Mount
  - Unmount
Top Tip 2: Use separate EPADAPTERs on all EVENTBINDINGs

Use Separate EP Adapters in CICS TS V4.2 to ensure no changes are needed to CICS Bundles containing EVENTBINDINGs during migration
Migrating bundles - JVMSERVER names

When using JVM server you need to have the same JVMSERVERs in all regions to ensure no changes are needed when migrating CICS bundles project containing OSGi bundles or Web project.

Defined in the .osgibundle or .warbundle in CICS bundle project.

```xml
<osgibundle symbolicname="com.ibm.cics.server.examples.hello" version="1.0.0" jvmserver="DFHJVMS"/>
```
Summary

- A place for everything
  - Create separate file systems for install, critical and temp artifacts
  - Have places in those file system for each type of artifact
  - Consider requirements for sharing across LPARs

- Everything in it's place
  - Put each artifact into it's appropriate place
  - Use symlinks where necessary
  - Consider requirement for recovery and backup

- Secure access
  - Set permission bits on each file system to control access
  - Use ACLs to solve the multiple writer problem
  - Consider requirements for FTP access

- Manage change
  - Manage changes to zFS artifacts through a SCM
  - Consider references from zFS files, to minimise changes

Complete your session evaluations online at www.SHARE.org/Seattle-Eval