• z/OS UNIX System Services, which is an element of z/OS, is a UNIX operating environment.
• It is implemented within the z/OS operating system and is also known as z/OS UNIX.
• z/OS UNIX is made up of MVS architected services that behave as though the operating system is a UNIX system.
  - All applications are started as regular MVS address spaces without initially caring whether they have to acquire UNIX process characteristics.
  - The z/OS address space is given the characteristics of a UNIX process at the time of the first call to a z/OS UNIX system service.
  - The z/OS address space is then said to be *dubbed* and the system makes necessary updates to control blocks so that from now on the address space is also considered to be a UNIX *process*. Note that new address spaces or tasks created as a direct consequence of calls by existing UNIX process or thread are systematically dubbed when they are initialized
• UNIX also provides the following operating system features:
  - Multitasking with virtual memory support and address spaces isolation.
  - All users must be authenticated by a valid account and password to use the system
  - Files are owned by particular accounts.
  - The owner can decide whether others have read or write access to the files he or she owns.
  - Tooling and command language for system and application developments.
  - One peculiar feature of UNIX is the huge amount of different user commands.
  - A unified file system to represent data, programs, and any input or output ports used to transfer data as files, nested in directories, in a hierarchical tree structure.
  - Support for distributed processing.
• UNIX has three large functional blocks; The kernel, the shell, and the utilities.
  NOTE: Technically, only the kernel and the shell form the operating system, while the utilities are intended to make the operating system more immediately useful to the user.
  - The **kernel** functions are of two broad types: Autonomous and responsive. Kernel functions, such as allocation of memory and CPU, are performed without being explicitly requested by user processes, therefore *autonomous*.
  - Other functions of the kernel, such as resource allocation and process creation and management, are initiated by explicit requests from processes.
    NOTE: UNIX users are not required to have kernel knowledge in order to use the system.
  - The z/OS UNIX shell can be compared to TSO/E for z/OS.
    It is the interactive interface to z/OS UNIX. The shell feature comes with multiple commands and utilities which are in most cases the same as the ones that come with a UNIX system. The z/OS UNIX shell is based on the UNIX System V shell with some of the features from the Korn shell.
    NOTE: The z/OS UNIX shell conforms to the POSIX 1003.2 standard and to the XPG4 standard. [POSIX 1003.2 distinguishes between a command (a directive to the shell to perform a specific task) and a utility (a program callable by name from the shell). To the end user there is no difference between a command and a utility.]
• The z/OS Distributed File Service zSeries File System (zFS) is a z/OS UNIX System Services (z/OS UNIX) file system that can be used in addition to the hierarchical file system (HFS).
  - zFS file systems contain files and directories that can be accessed with z/OS UNIX application programming interfaces (APIs). These file systems can support access control lists (ACLs).
  - zFS file systems can be mounted into the z/OS UNIX hierarchy along with other local (or remote) file system types (for example, HFS, TFS, AUTOMNT, and NFS).
• zFS does not replace HFS, rather it is complementary to HFS.
  - zFS can be used for all levels of the z/OS UNIX System Services hierarchy (including the root file system).
  - Because zFS has higher performance characteristics than HFS and is the strategic file system, HFS might not be supported in future releases, which will cause you to migrate the remaining HFS file systems to zFS.
• zFS provides many features and benefits:

Performance zFS provides significant performance gains in many customer environments.
  - zFS provides additional performance improvements when running sysplex-aware in a shared file system environment.

Restart zFS reduces the exposure to loss of updates. zFS writes data blocks asynchronously and does not wait for a sync interval. zFS is a logging file system.
  - It logs metadata updates. If a system failure occurs, zFS replays the log when it comes back up to ensure that the file system is consistent.

Aggregate movement As a part of supporting read-write mounted file systems that are accessed as sysplex-aware, zFS automatically moves zFS ownership of a zFS file system or the system that has the most read-write activity.

Commands You can use the **man** command view the descriptions of zFS command manual pages. To use man pages, enter **man** followed by the command information you want to display NOTE: You must enter the **zfsadm command suite entries as one word**. Table below shows examples of the zFS man commands.

zFS man command examples

zFS command	man command
ioeagfmt	man ioeagfmt
mount	man zfsmount
zfsadm aggrinfo	man zfsadmaggrinfo
zfsadm query	man zfsadmquery

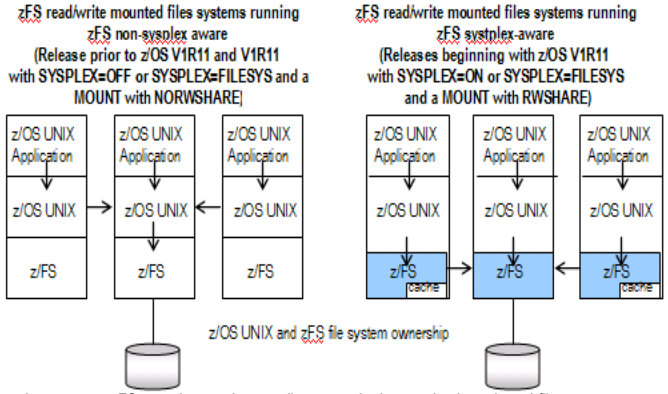
z/OS UNIX Kernel Services
- Creation and management of processes
- The file system
- The communication system
- The operating system start up processes

The TSO/E **OMVS** command is one method of accessing the z/OS shell. It provides a 3270 terminal interface to the shell. If you come from a UNIX or AIX background, you will encounter some differences when you begin to use the OMVS interface to the shell. The 3270-type terminal interface may surprise you!

z/OS and UNIX functional comparison

Function	z/OS	UNIX
Background work	Submit batch JCL	sh, cmd &
Configuration parameters	SYS1.PARMLIB	/etc
Data Management	DFSMS, HSM	tar, cpio, pax
Debug	TSO TEST	dbx
Editor	ISPF option 2	ed, sed, oedit, tshell
Initiate new task	ATTACH, LINK, XCTL	fork(), spawn()
Interactive access	TSO LOGON	telnet/rlogin to sh/tcsh
Job management	SDSF	ps, kill()
List files	ISPF option 3.4 TSO LISTCAT (LISTC)	ls
Long running work	Started task (STC)	daemon
Post IPL commands	COMMNDxx	etc/rc
Power user	RACF OPERATIONS	superuser or root
Primary configuration	IEASYSxx	BPXPRMxx
Primary data index	Master Catalog	root ("/") directory
Procedural language	CLIST, REXX, SYSTEM REXX	Shell scripts, REXX
Program Products	LNKLST (Link List)	/usr
Resident programs	LPA (Link Pack Area)	sticky bit
System logging	SYSLOG	SYSLOGD
System programs	LNKLST (Link List)	/bin
Test programs	STEPLIB	/sbin
User data	&SYSUID or &SYSPREF	/u/<username>
User identity	user/group	UID/GID

• zFS and HFS can both participate in a shared sysplex environment.
• zFS supports a shared file system capability in a multisystem sysplex environment. The term shared file system environment refers to a sysplex that has a BPXPRMxx specification of SYSPLEX(YES).
  - That is, users in a sysplex can access zFS data that is owned by another system in the sysplex



• For full sysplex support, zFS must be running on all systems in the sysplex in a shared file system environment and all zFS file systems must be compatibility mode file systems (that is, they cannot be file systems in multi-file system aggregates). zFS muti-file system aggregates are not supported by automount.
• **SMF record type 30** reports activity on a job and job step basis.
  - Even though file system activity is included in the EXCP count for the address space, the process section in the record breaks down the EXCP count into the following categories:
    > Directory reads
    > Reads and writes to regular files
    > Reads and writes to pipes
    > Reads and writes to character special files
    > Reads and writes to network sockets

Applications can reduce lookup activity by using the **chdir** command to change the working directory and specifying only the file name when opening a file.

• You can monitor the file system activity of various classes of users by post processing SMF 30 records.
  - This type of monitoring might be helpful in forecasting DASD and other system resource requirements.
  - If key jobs appear to be doing many lookups, your installation might be able to reduce this overhead.
  - To reduce overhead, reorganize the file system so that key files are closer to the root of the file system.
• SMF records contain a program name field for job steps that are initiated by fork(), spawn(), or exec().
  - For interactive commands, this feature allows performance analysts to determine what resources were required to complete a particular command.
    NOTE: If a user runs the OMVS command with the SHAREAS option or sets the environment variable BPX_SHAREAS to YES, two or more processes might be running in the same address space. In this case, SMF provides process identification only for the first process in the address space. However, resource consumption is accumulated for all processes that are running.
• **Enhanced ASCII functionality** allows z/OS UNIX to deal with files that are in both ASCII and EBCDIC format. z/OS is an EBCDIC platform. The z/OS UNIX shells and utilities are configured as EBCDIC programs — that is, they encode their characters in the EBCDIC codeset.
  - If you wanted to convert files from EBCDIC to ASCII or ASCII to EBCDIC, you needed to use **iconv**.
  - The introduction of Enhanced ASCII functionality gives you the ability to have z/OS UNIX deal with applications and their data in your choice of ASCII or EBCDIC codesets.
• z/OS UNIX still operates as an EBCDIC system, but it can automatically convert the data from ASCII to EBCDIC and back as necessary to complete commands and tasks.

BPXPRMxx PARMLIB member contains the parameters that control the z/OS UNIX System Services (z/OS UNIX) environment.

• To specify which BPXPRMxx PARMLIB member to start with, the operator can include OMVS=xx in the reply to the IPL message or can include OMVS=xx in the IEASYSxx PARMLIB member. The two alphanumeric characters, represented by xx, are appended to BPXPRM to form the name of the BPXPRMxx PARMLIB member.
• You can use multiple PARMLIB members to start OMVS. This is shown by the following reply to the IPL message:
  R 0,CLPA,SYSP=R3,LNK=(R3,R2,L),OMVS=(AA,BB,CC)
  - The PARMLIB member BPXPRMCC would be processed first, followed by and overridden by BPXPRMBB, followed by and overridden by BPXPRMAA. This means that any parameter in BPXPRMAA has precedence over the same parameter in BPXPMRBB and BPXPRMCC.
  - You can also specify multiple OMVS PARMLIB members in IEASYSxx. For example:
    OMVS=(AA,BB,CC)
  - To modify BPXPRMxx PARMLIB settings without re-IPLing, you can use the SETOMVS operator command, or you can dynamically change the BPXPRMxx PARMLIB members that are in effect by using the SET OMVS operator command.