RSM ENTERPRISE SOLUTIONS

Everything you wanted to know about mainframe security, pen testing and vulnerability scanning .. But were too afraid to ask!

Mark Wilson

markw@rsmpartners.com

Session Details: UNIX System Services (USS)

Agenda

- Introduction
- Objectives
- What is UNIX?
- What is z/OS UNIX aka USS?
- How do you do USS Security it properly?
- Summary
- References
- Questions

IBM Mainframe
Are they really secure?

RSM ENTERPRISE SOLUTIONS

INTRODUCTION

Introduction

- Mark Wilson
  - Technical Director at RSM Partners
  - I am a mainframe technician with some knowledge of Mainframe Security
  - I have been doing this for over 30 years (34 to be precise ☺)
  - This is part four of seven one hour long sessions on mainframe security
  - Full details can be seen on the New Era Website:
    - http://www.newera-info.com/MF-SEC.html

This is where Mark Lives!

My passions outside of work?

- One wife and three daughters…..enough said…..don't have anytime or money for anything else….or so they tell me ☺
- Motorbikes
  - www.wilson-mark.co.uk
- Football
  - www.wba.co.uk
- Scuba Diving
  - Way too many links to list here…..But I have been and dived here
  - http://en.wikipedia.org/wiki/Chuuk_Lagoon

OBJECTIVES

Objectives

- Slightly different session this month as this is its not about hacking
- This session is all about doing z/OS USS Security Properly
- We will start with an overview of UNIX and USS
- We will then look at the things we need to do, so that we do USS security properly

RSM ENTERPRISE SOLUTIONS

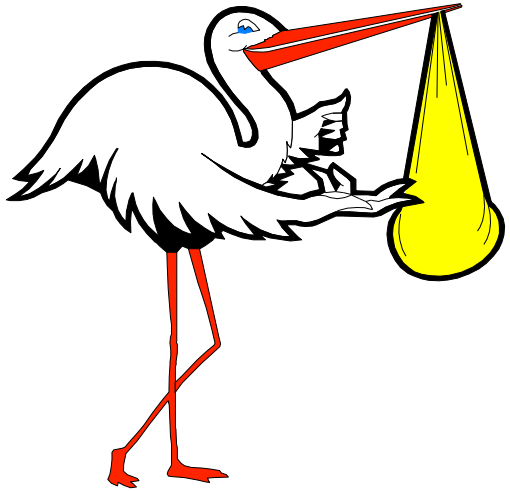
WHAT IS UNIX?

UNIX History

- UNIX is an OPEN SYSTEM
  - Following a set of Standards
  - Allowing multiple vendors
  - Allowing multiple systems
  - Providing Interoperability and Portability
  - Will run on any type of terminal
- Developed at Bell Labs in late 1960's
- Initially called MULTICS - MULTiplexed Information and Computing Service
- Implemented on PDP-7
- Command interpreter interface (shell)

Birth Of UNIX

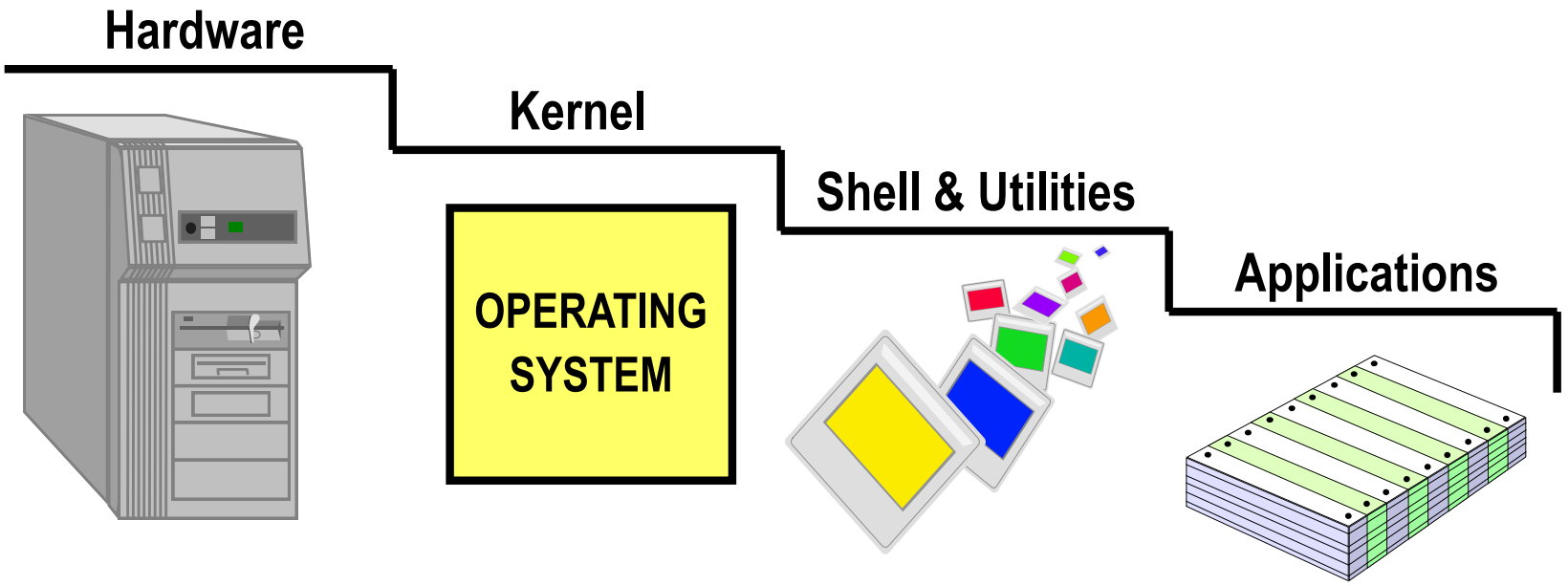
- AT&T began work on its own idea of the system:
  - Ken Thompson and Dennis Ritchie researchers
  - Scaled down version referred to as UNICS (UNiplexed Information and Computing System
  - Brian Kernighan suggested the name UNIX for new system



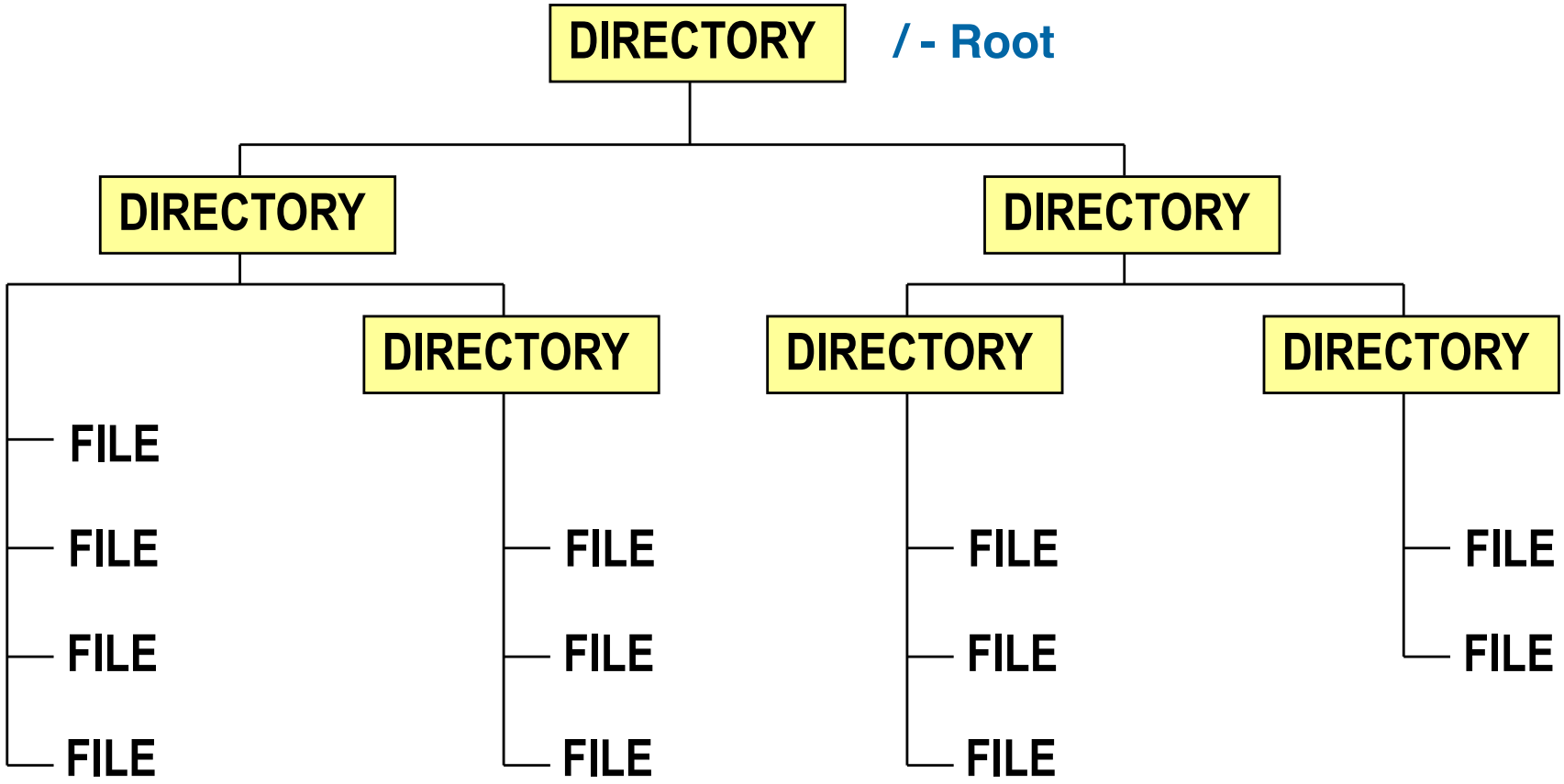
The Original Programmers

Dennis Ritchie and Ken Thompson working at the PDP-11.

UNIX Components



Hierarchical File System (HFS)



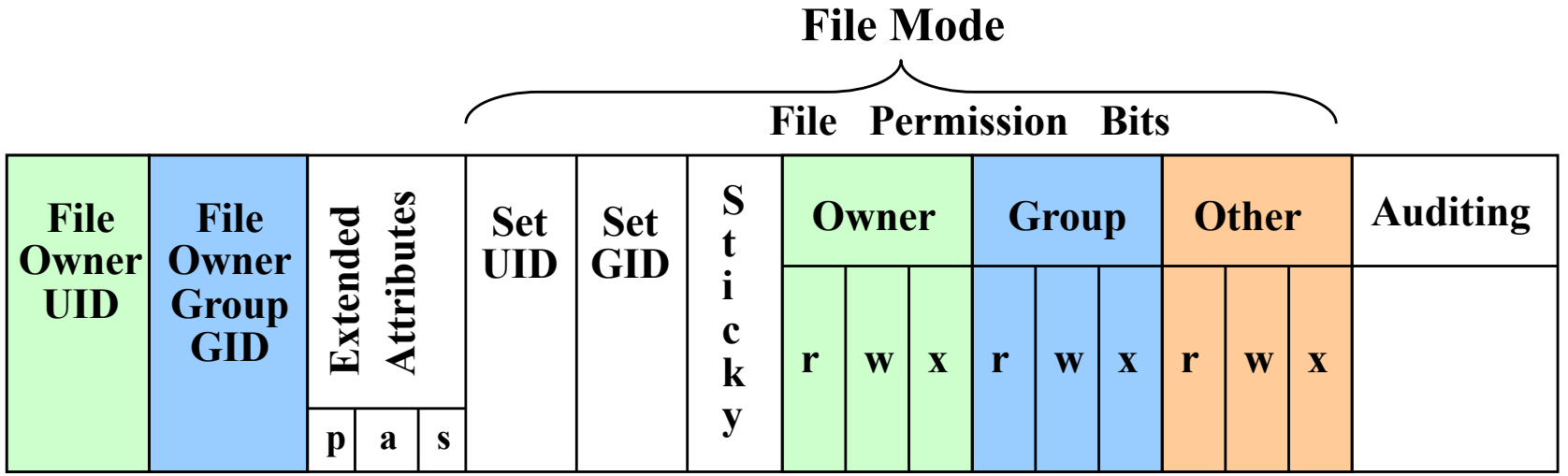
Hierarchical File System (HFS)

- Search through directories to find files
- Files are named by specifying directories in path plus file name /dir1/dir3/MyData
- Users usually specify the current working directory to eliminate having to specify complete path
- Users must have EXECUTE authority to the path in order to access a file

UNIX Security

- UID identifies user, its just a number
- GID identifies group to which user belongs, its just a number
- Access rights determined by
  - UID - if user is owner of file
  - GID - if user is in group that owns file
  - Other - if neither UID nor GID match

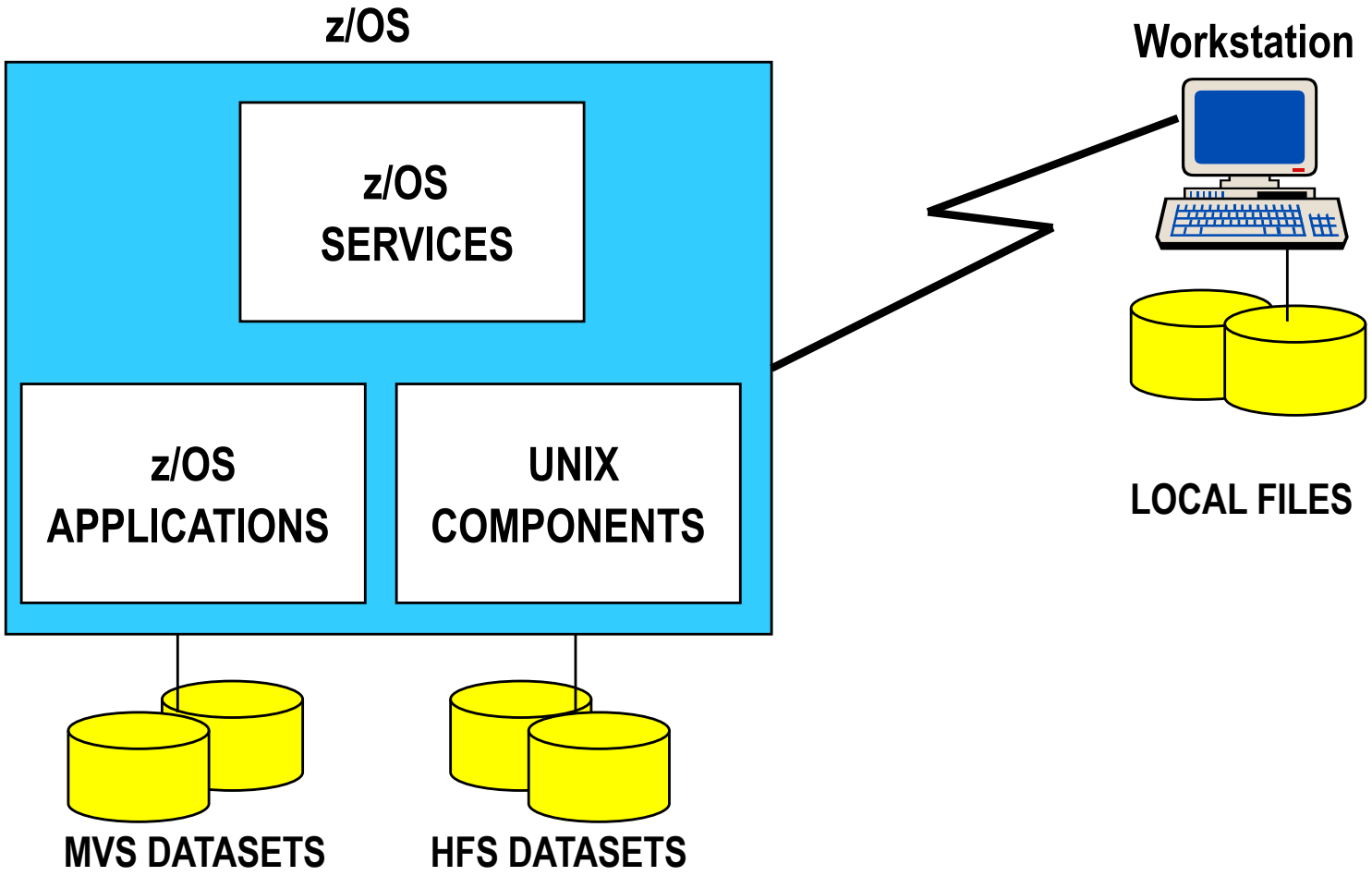
UNIX File Security Packet



RSM ENTERPRISE SOLUTIONS

WHAT IS Z/OS UNIX AKA USS

Open Systems on z servers

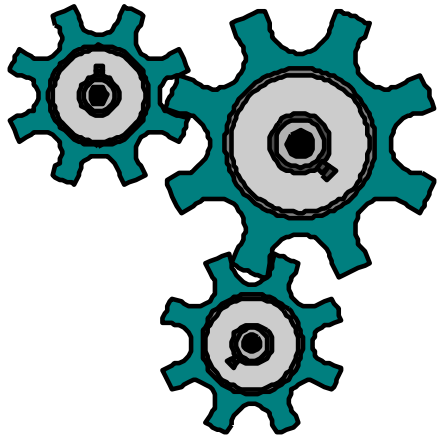


What is z/OS UNIX?

- Up to OS/390 2.5 known as OpenEdition MVS
- With OS/390 2.6 known as OS/390 UNIX System Services
- Base element of z/OS
- Has the look and feel of UNIX because it is UNIX
- Enables two open systems interfaces
  - An application program interface (API)
  - An interactive z/OS shell interface

What is z/OS UNIX?

- API's Programs can run in almost any environment:
  - Batch
  - TSO via OMVS
  - As Started Tasks

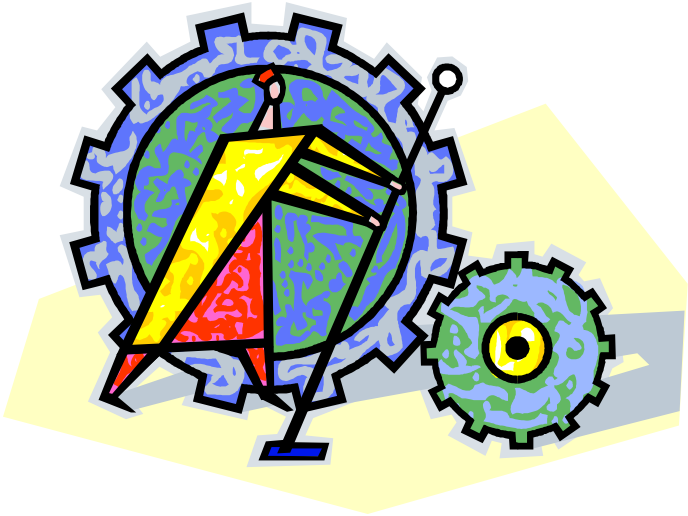


- Programs can request:
  - Only MVS services
  - Only z/OS UNIX
  - Both MVS and z/OS UNIX

- SHELL Interface is an execution environment
  - Programs run by shell users
  - Shell commands and scripts run by shell users
  - Shell commands and scripts run as batch jobs

What is z/OS UNIX Used For?

- Makes application development easier
  - Standard (open) programming interface
  - Interoperability in networks
  - Portable programs
  - Portable data
- Required by some products
  - TCP/IP
  - FTP
  - LDAP
  - Plus many others!!



Components of z/OS UNIX

- KERNEL - Low-level system code
- SHELL - A command processor
- FILE SYSTEM - Hierarchical File System (HFS)
  - Directories
  - Files
- DAEMONS - Processes that run in background i.e. Started Tasks
- COMMUNICATION SERVICES - Methods of access
  - TSO/E
  - VTAM
  - TCP/IP

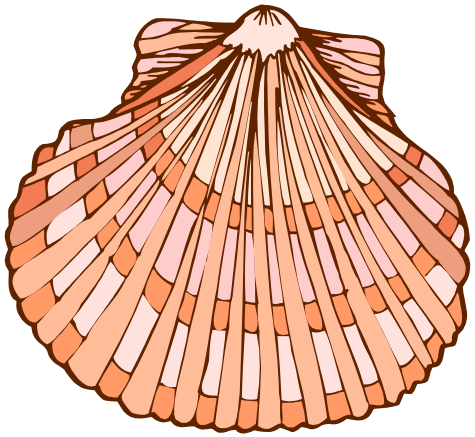
UNIX Kernel

- Manages memory
- Schedules work
- Controls machine data
- Executes shell instructions
- Enforces security

UNIX Shell

- UNIX Shell
  - interprets commands
  - runs program requested
  - coordinates activity between you and operating system

- Common shells in use:
  - bourne SHell - sh
  - C SHell - csh
  - TC SHell - tcsh
  - Korn SHell - ksh
  - Bourne Again SHell - bash

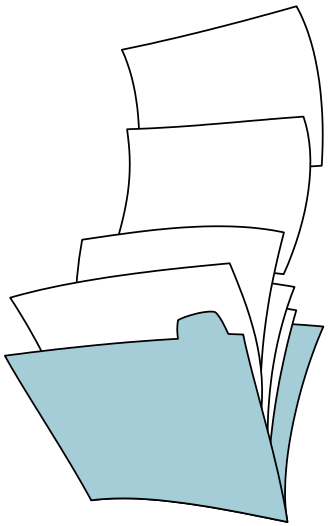


Hierarchical File System (HFS)

- File system is contained in z/OS dataset(s)
- Data sets can be SMS managed or non-SMS managed
- Additional file systems can be mounted
- Only a superuser can MOUNT or UNMOUNT a file system
- Security is handled by the SAF security interface on z/OS
- External security system needed for security of the file system

Hierarchical File System in z/OS

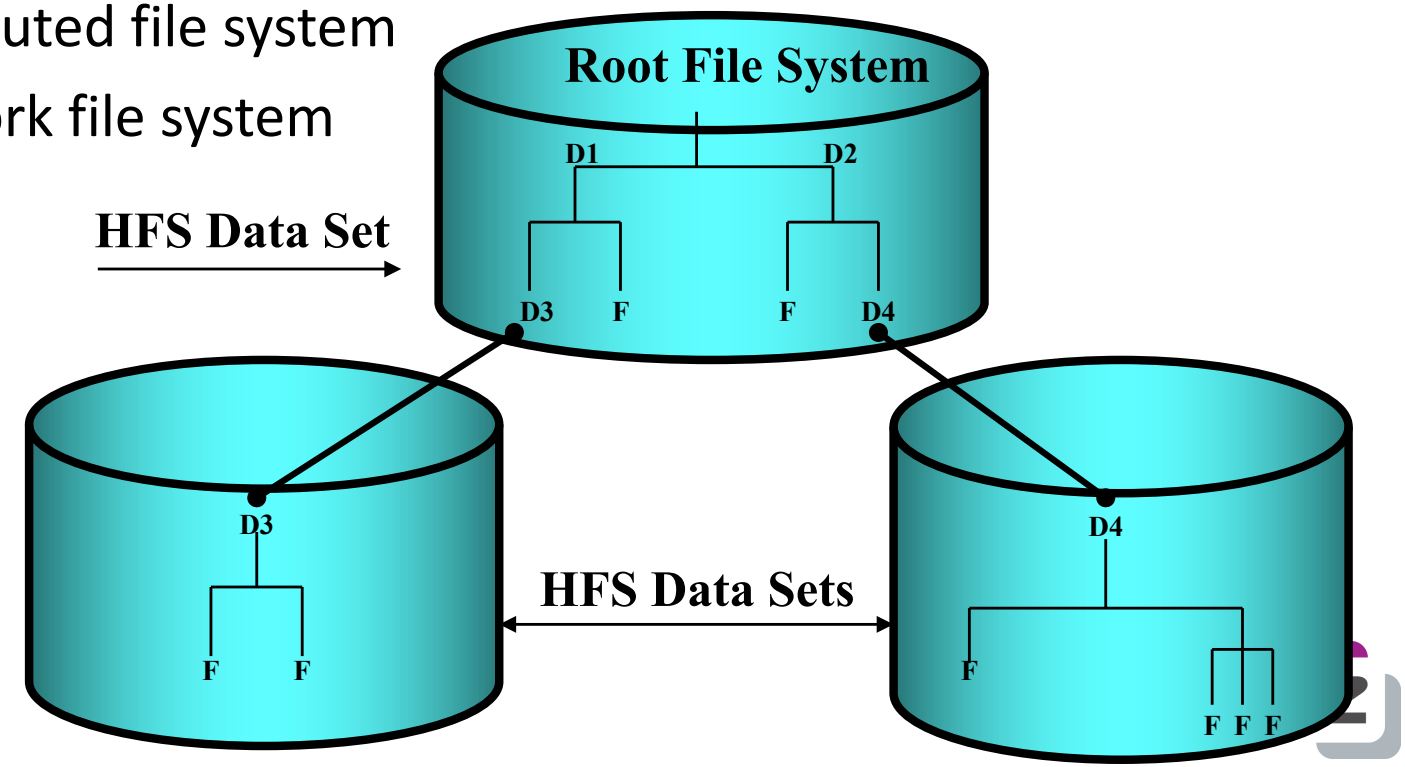
- Location for all data accessed by z/OS UNIX
- Can copy files between z/OS and HFS via TSO/ISPF panels



- Primary issue is long path-oriented names (like personal computer DOS names) and UNIX's mixed-case file names

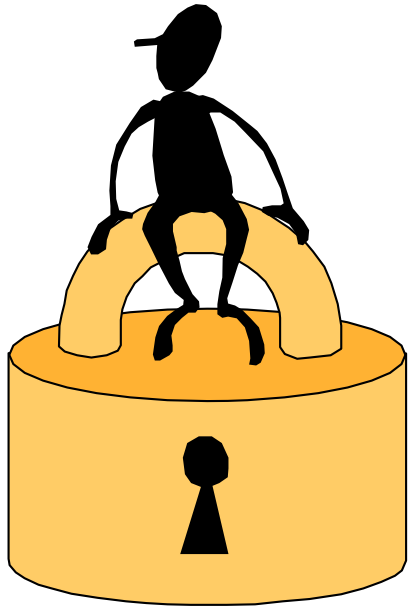
File System Mount Points

- Additional file systems can be mounted
  - HFS - Hierarchical file system
  - zFS - zSeries file system
  - TFS - temporary (or toy) file system
  - DFS - distributed file system
  - NFS - network file system

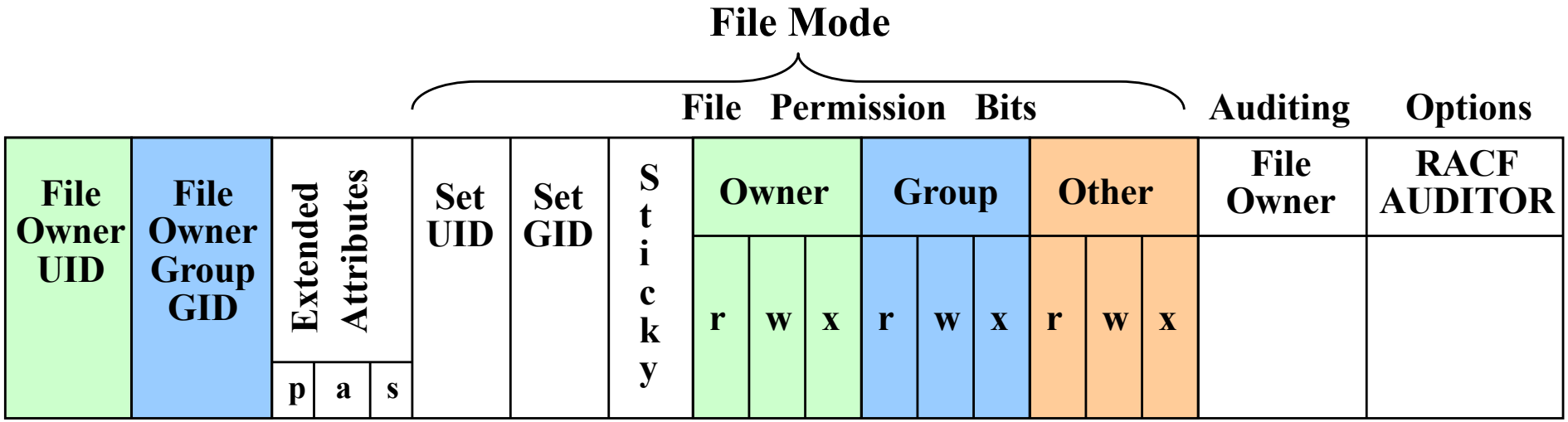


z/OS UNIX Security Functions

- User Validation
  - UID, GID
- File Access Checking
  - File Security Packet (FSP) containing File Permission Bits and now ACLS
- Auditing
  - FSP, File Audit Bits
  - RACF Systemwide Options
  - UNIXPRIV and FACILITY Classes
- Security Administration
  - RACF and UNIX Commands



z/OS UNIX File Security Packet



RSM ENTERPRISE SOLUTIONS

HOW DO YOU DO USS SECURITY IT PROPERLY?

How do you do it properly?

- You need a plan
- You need an owner for USS security
  - And it should NOT BE the z/OS system programmers
  - USS security should be owned by the security groups within your organisation
    - Security Engineering
    - Security Administration
    - Risk & Compliance
    - Audit

SOME RULES OF THE ROAD

UID Assignment

- You need a plan for:
  - Normal/Real Users….could use employee/staff number if unique
  - Started Tasks
  - Break Glass Users (Emergency Users)
  - Superusers
    - UID(0), Access to BPX.SUPERUSER, Trusted and Privileged
  - You may have to remediate the current system to align to the plan as stuff was just assigned previously

GID Assignment

- You need a plan for:
  - Default Groups
  - Access Control Groups
  - Owning Groups
  - All of the other types of groups you may have!

- Even if the plan simply states that a GID should not be defined

BPX.DEFAULT.USER

- Think enough has been said about this
- Just get off it ASAP
- z/OS 1.13 is the last release to support this profile
- There is lots of reference material out there for how this can be done

HFS & ZFS Security

- Created as VSAM Linear Datasets
- Use strict access control using RACF, ACF2 or TSS
- In a RACF environment suggest a fully qualified generic dataset profile
- No normal users need any access to these VSAM datasets
  - Only the creator and the userid used for backing them up
- Never make the HLQ the same as a real RACF Userid

Some File and Directory Ownership

- You need a plan, just like we do for ownership of z/OS data
- Data needs to be classified so that it can be secured and audited correctly
- Ownership of USS files and Directories needs to bee factored into your wider Joiner, Mover, Leaver (JML) process
- You need a regular monitoring report that shows unowned files and directories (BPXBATCH with –nouser and –nogroup is a good start)
- Consider creating some none personal userids and specific groups to own files and directories

Access Control Lists

- There are some good things about ACLs…..
  - Default file and directory security supported
  - Gets around the one UID & GID limit of the FSP
- But there are some bad things too…..
  - Not POSIX compliant
  - Security information is not transported to NON z/OS systems as they don’t understand them
  - Performance can be an issue
  - And many others……
- So if you need to use them do it after careful review and debate and their use should be an exception not the NORM!

Key Files & Directories

- There are key z/OS Unix files and directories that must be secured and the security policy should state the most important ones
- It Should dictate the settings for the file and directory FSP
- Key directories:
  - / (root) Holds all mount points
  - /bin Core programs, many APF or Program Controlled
  - /dev Holds many files used during IPL and Shell login
  - /etc Many key config files held here
  - /tmp All users need write access
  - /var Many services need write access

Key Files & Directories

- Key files:
  - /etc/rc
    - run commands executes at IPL time with UID(0) privilege
  - /etc/init.options
    - kernel control file
  - /etc/profile
    - default user profile settings/script
  - /etc/steplib
    - list of steplib datasets - location depends on BPXPRMxx
  - any cron files
    - Assume cron is not used so files should be secured

Z/OS UNIX Configuration Files

- Many UNIX services require configuration files
- Typically these are stored in the USS file system
- But some of the services can support PDS datasets
- If you can use a PDS then you should
- Consider a separate PDS/PARMLIB for USS
- Irrespective of the ESM (RACF, ACF/2 and TSS)
  - We have greater understanding
  - We have greater flexibility
    - Multiple users/groups can have access at different levels

BPXPRMxx Security Considerations

- You need strict change control around these parameters
- Even, consider having them stored in a specific PARMLIB dataset, that the z/OS sysprogs cannot update with their normal userids
- BPXPRMxx Keywords
  - NOSECURITY
    - No checking of multi system access from multiple LPARs
    - If coded, you may end up with corrupt data in a NON GRS configuration
  - NOSETUID
    - Means no extended security functions. Resulting in SETUID(0) or SETGID(0) bits not being honoured, eg Program Control or APF
  - NOWRITEPROTECT
    - Results in unprotected files, as in, no directory or file level security

UNIXPRIV: General

- Allows delegation of specific Superuser privileges

- What is/can a Superuser (root - Unix system administrator) do?
  - Full access to all Unix directories and files (like OPERATIONS)
  - Change directory/file owners and permissions (like SPECIAL)
  - Perform privileged Unix functions
  - Use privileges related to most unprotected FACILITY BPX resources and to some protected ones without permission
  - If BPX.DAEMON is not defined, can assume other user's identities

UNIXPRIV: General

- If BPX.DAEMON is not defined, can assume other user's identities

- Superuser authority assigned by:
  - OMVS(UID(0))
  - FACILITY - BPX.SUPERUSER
  - PRIVILEGED / TRUSTED for Started Tasks

UNIXPRIX: SECADM Related

- SUPERUSER.FILESYS.CHANGEPERMS
  - chmod and setfacl any permit
- SUPERUSER.FILESYS.CHOWN
  - Chown any file or directory
  - READ access is required
  - Also requires search (x) access to traverse directories
- Limit access to CHNAGEPERMS and CHOWN to security administrators and use instead of BPX.SUPERUSER

UNIXPRIX: SECADM Related

- SHARED.IDS
  - Prevents assignment of existing UID or GID values
- CHOWN.UNRESTRICTED
  - Existence of profile acts as switch to activate - must be Discrete
  - Allow any user to 'chown' their files & directories to any other user

UNIXPRIV: SECADM

- FILE.GROUPOWNER.SETGID
  - Change method of GROUP inheritance for new files and directories
    - Standard Unix behaviour - GROUP taken from parent Directory in which new subdirectory or file is created
    - New optional behaviour - GROUP taken from 'effective' gid in User Security Packet (USP) of the creating process
  - Existence of profile acts as switch to activate - must be Discrete
  - Behaviour depends on set-gid bit for the parent directory
    - If bit OFF (default) - GROUP taken from USP
    - If bit ON - GROUP taken from Directory as before
    - Must use 'chmod' command to turn on set-gid bit for directory in order for it to revert to original behaviour
    - 'ls' display shows 's' ('x' on) or 'S' ('x' off) in x-bit place for GROUP

- Currently running processes do not recognize the change

UNIXPRIV: Maintenance Related

- SUPERUSER.FILESYS.MOUNT
  - 'mount' and 'chmount' HFS files
  - READ access required with NOSETUID only
  - UPDATE access required with SETUID or NOSETUID

- SUPERUSER.FILESYS.QUIESCE
  - 'quiesce' and 'unquiesce' HFS files
  - READ access required with NOSETUID only
  - UPDATE access required with SETUID or NOSETUID

- Limit access to Support staff responsible for maintaining UNIX

UNIXPRIV: Service Related

- SUPERUSER.FILESYS.PFSCTL
  - Physical File System services
- SUPERUSER.FILESYS.VREGISTER
  - Register as VFS server (e.g. NFS)
- SUPERUSER.IPC.RMID
  - Release IPC resources ('ipcrm’)
- SUPERUSER.PROCESS.GETPSENT
  - Get process status info

UNIXPRIV: Service Related

- SUPERUSER.PROCESS.KILL
  - Issue kill to processes
- SUPERUSER.PROCESS.PTRACE
  - Use ptrace through dbx debugger
- SUPERUSER.SETPRIORITY
  - Increase own priority
- All Require READ access to use
- Typically, limit access to UNIX processes or support users performing debugging

UNIXPRIV: Access Related

- SUPERUSER.FILESYS
  - Grants access to all Unix files and directories at specified permit level, even if denied access by permission bits and ACLs (unless ACLOVERRIDE is defined)
    - READ access grants Read access to all files and search all directories
    - UPDATE access grants Write access to all files
    - CONTROL access grants Write access to all directories
- SUPERUSER.FILESYS.ACLOVERRIDE
  - Causes ACL permissions to overrule access SUPERUSER.FILESYS would otherwise grant
  - Permitting access grants access like SUPERUSER.FILESYS
- RESTRICTED.FILESYS.ACCESS
  - Prohibits RESTRICTED users from gaining access via OTHER permission bits
  - Permitting READ access bypasses the restriction

Other Stuff

- We only have an hour in this session…. So…..
- Finish the research and setup of the UNIXPRIV Class
- Other things you need to have a plan for:
  - Profiles in the:
    - APPL Class (OMVSAPPL)
    - FACILITY Class (BPX. Prefixed profiles)
  - Security for Servers and Daemons
  - FSP Attributes
    - Extended Attribute Bits (p, a & s), The sticky bit & Audit attributes
  - Joiner Mover Leaver Process
  - Reporting, Monitoring & Real Time Alerting of USS Events
  - Auditing…When was the last time your USS security implementation was audited properly??

Summary

- USS Security is here to stay
- You cannot run a z/OS system today without needing USS
- Therefore, we need to get USS security under control
- Many installations do not and still leave it in the hands of the z/OS systems programmer
- You need to get your USS security posture as strong as you z/OS one….and this is even more important if you are storing Production Applications and Data in USS

References

- Unixpriv and other information provided by Bob Hansell, RSH Consulting
  - http://www.rshconsulting.com/racfres.htm#RSHpres
- z/OS Security Server (RACF) Security Administrator's Guide
  SA22-7683
- z/OS Security Server (RACF) Security Auditors Guide
  SA22-7684
- z/OS UNIX System Services Planning - GA22-7800
- z/OS UNIX System Services User's Guide - SA22-7801
- z/OS UNIX System Services Command Reference
  SA22-7802
- z/OS UNIX System Services Home Page
  http://www-1.ibm.com/servers/eserver/zseries/zos/unix/
- HFS Unload Utililty – irrhfsu (Download from RACF home page)
- mvs-oe listserv - http://www2.marist.edu/htbin/wlvindex?mvs-oe
- z/OS SYS1.SAMPLIB member BPXISEC1

Questions

Contact Details

Mark Wilson
RSM
markw@rsm-es.com
Mobile +44 (0) 7768 617006
www.rsm-es.com