Auditing UNIX System Services (in OS/390)

Mark S. Hahn, mhahn@us.ibm.com
Consultant, IBM Corporation
ISACA Toronto, Canada
March 22, 2001

Trademarks

The following are trademarks or registered trademarks of the International Business Machines Corporation:

OS/390

RACF

SecureWay

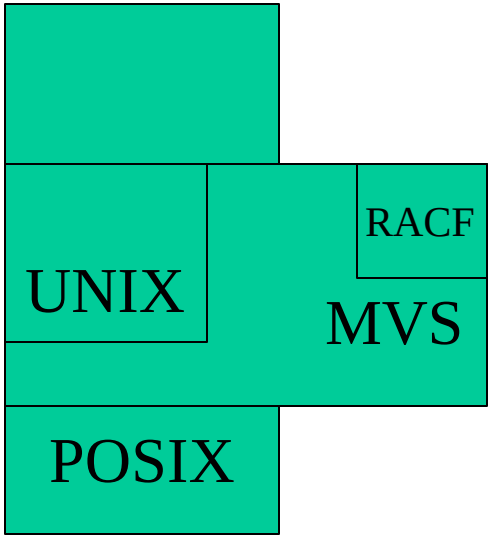
z/OS

Topics

- Introduction to UNIX Environment
  - Definitions
  - Unix <=> OS/390 UNIX
- Users
- Unix Access Controls and Audit Features
- Unix System Services Controls and Features
- RACF Controls
- Summary

What is UNIX System Services?

- POSIX - Portable Operating System Interface
- Unix under OS/390 and z/OS
- Introduced in MVS/ESA 5.1
- Controlled by *parmlib*(BPXPRMxx)
- MVS/ESA 5.2.2 supports XPG4 base of X/Open - “This release is a major step toward IBM’s intention to have MVS/ESA branded as a UNIX system.”



Definitions

- The term *OS/390 UNIX System Services* and its abbreviation *OS/390 UNIX* are new names for what was previously known as OpenEdition in earlier levels of OS/390 and MVS
- OMVS – precursor to Unix System Services; originally OpenEdition/MVS
- uid(n) – UNIX user number (user id)
- Userid – RACF 8-character userid
- Data set – MVS DASD unit of data
- File – UNIX unit of data
- Catalog – How to find a data set
- Directory – How to find files

More Definitions

- Signon controls:
  - RACF enforces password expiration, rules, REVOKE for invalid, no files (/etc/password or /etc/shadow)
- List of groups
  - Up to 300 supplementary group connections checked

RACF vs UNIX Differences

- MVS has (record-oriented) data sets
- Data set names are all capitals
- Data sets are discrete entities, cataloged by HLQ
- File authorizations carried in RACF profiles, maintained in RACF database, and permit lists.
- Catalog access controlled only at HLQ level.

- Unix has byte-oriented files.
- File names are mixed case -- and are case sensitive.
- File collections exist.
- File authorizations are carried within the directory in FSP (File Security Packet)
- Every level of catalog access has separate authorizations possible.

Environment

Defining the Environment

- UNIX System Services is integral part of OS/390 and z/OS.
- Starts at IPL
- Definitions in BPXPRMxx member(s)
- 3 levels
  - Minimal
  - Unix level
  - OS/390 level

Minimal Level

- No Hierarchical File System (HFS) – only TFS (Temporary File System)
- Nothing stored long-term

UNIX Level

- HFS in use
- Long Term storage (DASD)
- Stored executable program files
- Basic UNIX security

OS/390 Level

- HFS in use
- Long-term storage (DASD)
- BPX.DAEMON profile in FACILITY class defined
  - Invokes program controls
  - Requires PROGRAM profiles for APF programs

Role of *parmlib*

- BPXPRMxx defines UNIX System Services environment
- File system
- System-wide user limits and controls
- IEASYSxx: OMVS=(xx)
- SET OMVS=(xx)
- IBM suggests splitting into two

File System BPXPRM0F

```
FILESYSTYPE TYPE(HFS)            /* Type of file system to start  */
            ENTRYPOINT(GFUAINIT) /* Entry Point of load module    */
            PARM(' ')            /* Null PARM for physical file
                                    system                        */
          /* ASNAME(adrspc01) */  /* Name of address space for
                                    physical file system - not
                                    used for HFS                  */

 ROOT     FILESYSTEM('SYS4.&SYSNAME..OMVS.ROOT.HFS')
            TYPE(HFS)              /* Type of File system          */
            MODE(RDWR)             /* (Optional) Can be READ or RDWR.
          MOUNTPOINT('/')          /* Must be entered in quotes.   */

 /*   LDAP Server - Security Server   */
     MOUNT  FILESYSTEM('SYS4.&SYSNAME..OMVS.LDAP.HFS')
          TYPE(HFS)
          MODE(RDWR)
          MOUNTPOINT('/usr/lpp/ldap')
          NOSETUID                 /* ignore setuid/gid mode bits   */
```

File System

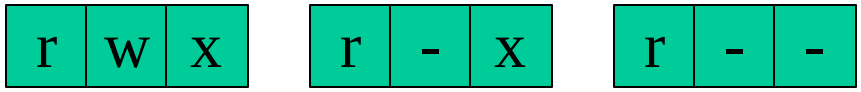
- HFS - Hierarchical File System
  SMS File type
- specified in BPXPRMxx member of parmlib
- Directory tree requires user be authorized
  ALTER level actions (rename, delete) equate to WRITE access to directory.
- Access managed by FSP (File System Packet) instead of profiles
- Access is 3-triplets:
  - owner: read / write / execute (UID)
  - group: read / write / execute (GID)
  - world: read / write / execute

FSP

```
Accesses    Audit   links User   Group Size  Date & Time   file name
-rw-r-xr-x  fff sf-   2   Shari  Sales 1185  Oct 17 10:22  admin.doc
```

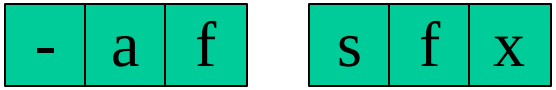
File Access

- “Files” not “data sets”
- Must have SEARCH (execute) access to directory for *chdir* to directory
- Use *ls -alF* to review directory with accesses displayed



owner can read, write, and execute; **group** can read and execute; **world** can read.

- **Owner** set (all (a) success (s), fail (f) or none (-)) and **auditor** set for each level



File System

- Audit
  - 2-triplets with 4 values: owner set and auditor set
  - 4 possible values: ‘a’ all, ‘s’ success, ‘f’ fail, ‘-’ none

FSP

```
Accesses    Audit   links User   Group Size  Date & Time   file name
-rw-r-xr-x  fff sf-   2   Shari  Sales 1185  Oct 17 10:22  admin.doc
```

HFS files

- New file type in OS/390 (PDS, LIBRARY, HFS)
- Should NOT be owned by user, system data set – UACC(NONE), UPDATE by OMVS
- Requires use of data mover, not IEBCOPY, IDCAMS, etc.
- Mounted by BPXPRM0F (or BPXPRMxF or BPXPRMxx)
- Automount is good thing (more later)

/etc Directory

- IBM strategy: place all customized data in the /etc directory.
- Where most important programs and scripts live
- CaSe cOuNtS
- /etc/rc - system startup script (“gentlemen, start your daemons”)

```
 Type  Filename
_ File  log
_ File  logstart
_ Dir   NetQ
_ File  profile
_ File  protocol
```

Display from ISHELL – extract of /etc directory

Automount

- Major goodness!
- Watch out for idle and delay … minutes not seconds.
- Set in /etc/rc
  - **/usr/sbin/automount**
  - Implies /etc/u.map

```
name                *
type                HFS
filesystem          OMVS.<uc_name>.HFS
mode                rdwr
duration            10
delay               0
```

- Recycle via /etc/rc command

Gotcha's

- **Rexx exec from TSO prompt:** Found on Tips and Tricks page
  copytree / NOT copytree '/'

- **whoami** and **ls** (uid > userid lookups) report first match when multiple userids share uids.
  - `-rwxr--r--  1 OMVSKERN OROOTGRP 1757 Apr 27  2000 rc`
  - `drwxrwxrwx  5 OMVSKERN OROOTGRP 0 Apr 21  2000 recover`

System Limits: BPXPRM0L

- Global values set by BPXPRMxx
- Overriden by OMVS segment on userids

```
 MAXCPUTIME(1000)

 MAXSHAREPAGES(131072)           /* System will allow at most 131072
                                      pages of shared storage to be
                                      concurrently in use     */


SUPERUSER(BPXROOT)

 TTYGROUP(TTY)
```

Additional parmlib Controls

- ALLOCxx – Control Allocation Requests
- COFVLFxx – Cache RACF services
  - Activate IRRUMAP and IRRGMAP classes within VLF (Virtual Lookaside Facility)
- CTnBPXxx – Control UNIX System Services tracing
- IEADMR00 – Dump Data Gathering
- SMFPRMxx – SMF parameters
  - Timeout controls

7 New RACF Classes

- DIRACC
  - Directory access (‘r’ or ‘w’)
- DIRSRCH
  - Directory search (‘x’)
- FSOBJ
  - File system object AUDIT(create/delete) LOGOPTIONS(access)
- FSSEC
  - File system security objects
- IPCOBJ
  - InterProcess Communications
  - AUDIT(create/delete) LOGOPTIONS(UID, GID and mode changes)
- PROACT
  - LOGOPTIONS relates users looking at data other processes
- PROCESS

New Resource Classes

- Only two parameters (at most) matter
  - All AUDIT and some LOGOPTNS
  - Profiles, in/active, residency, generic immaterial
  - Impacts degree and scope of **audit** UNIX events
  - NO impact on security decisions
- DIRACC – Directory access
- DIRSRCH – Directory searching (‘x’ to directory)
- FSOBJ – access to files and diectories
- FSSEC – changes to file system security objects

New Resource Classes (2)

- PROACT – accessing data from other processes
- IPCOBJ – INterPreocess Communication

- Do not expect RACF violations for these classes, they show up as SMF080 with ee-qq values:
  DIRSRCH fails with 28-01
  DIRACC fails with 29-01

Key Failure Codes

- UNIX is very directory oriented: operations requiring directory access
  - Succeed with unique event: ee-00
  - Fail with directory access (28-01 or 29-01)
- Directory search (‘x’ level access to directory) 28-nn [DIRSRCH]
  - chdir (change working directory) 32-00 or 28-01
- Directory access(‘r’ or ‘w’ access to directory) 29-nn [DIRACC]
  - mv (rename file) 47-00 or 28-01 or 29-01
- If not directory-base request:
  - Succeed with unique event: ee-00
  - Fail with unique event: ee-nn
  - chmod (change file mode) [FSSEC] 33-00 or 33-01

Users

Type of Users

- Normal users (non-zero)
- Superusers (aka ROOT)
- Scoped Superusers
- Default users
- Daemons

In all cases, SMF records USERID and uid for events

Normal Users

- Have an OMVS segment and access to TSO
- OMVS segment contains: UID, HOME, PROGRAM and xxxMAX parameters

```
lu sysmsh2 omvs noracf
 USER=SYSMSH2

 OMVS INFORMATION
 ----------------
 UID= NONE     ← problem?
 HOME= /u/sysmsh
 PROGRAM= /bin/echo
 CPUTIMEMAX= NONE
 ASSIZEMAX= NONE
 FILEPROCMAX= NONE
 PROCUSERMAX= NONE
 THREADSMAX= NONE
 MMAPAREAMAX= NONE
 READY
```

Superusers

- Have uid(0) within UNIX
- Are considered SPECIAL+OPERATIONS+APF authorized within UNIX System Services
- Gain uid(0) via:
  - UID(0) in OMVS segment
  - READ access to BPX.SUPERUSER
  - TRUSTED or PRIVILEGED STC

uid(0) Discussion

- Who needs uid(0)?
  - Systems programmers / System maintenance / Me / Systems tasks (FTP, etc)
- Alternative: BPX.SUPERUSER in FACILITY
- Alternative #2.08: Scoped using UNIXPRIV SUPERUSER.privilege

```
# who
SYSPMH       ttyp0000          Feb 24 10:18
# whoami
OMVSKERN
```

Scoped Superusers

- Have a subset of superuser powers
- uid(non-zero)
- READ access to SUPERUSER.xxxx profile(s) in the UNIXPRIV class
  xxxx is the privilege: MOUNT, CHOWN

Default Users

- Have no OMVS segment **(VERY IMPORTANT)**
- BPX.DEFAULT.USER defined
- Used for non-shell access (not exclusive)

Daemons

- Daemons (think subsystems) do work on behalf
  Servers with identity of other users
- syslogd, inetd, rlogind, cron are all daemons
- BPX.DAEMON profile
- Long running, start at IPL (by UNIX System Services)
- STARTED profile
- Ensure files protected
- Ensure data sets protected
- Authorization group

OMVS Segments

- User profile – basic support
  - UID
  - HOME
  - PROGRAM
- Group profile
  - GID (0)
  - List of groups allows current connect group + up to 300 connected groups for authorization

User / Group Segments

- Default group or Logon group for user must have GID to enter OMVS shell
- Groups are second level of file access authorization: -rwx**RWX**rwx
  - If owning uid doesn't match, then group (List of Groups), then world.

Accessing OMVS

- Sign on to TSO
  - RACF validation:
    - userid / password / any controls
- Issue OMVS or ISHELL command
  - must have UID in OMVS segment*[38-01]
    - UID is number 0<= UID <= 2,147,483,647 [38-02]
  - must have GID in OMVS segment [38-03]
    - GID is number 0<= GID <= 2,147,483,647 [38-02]
- Enter and leave at will

User Signon

- RACF
  - userid with RACF and OMVS segments
  - RACF segment userid / password and more
  - OMVS segment: UID, GID, directory
- UNIX
  - user definitions: */etc/passwd* (global read) userid:enc-pswd:UID:dft-grp:dir:shell
  - user passwords: */etc/shadow* (no access)
  - user actions are checked / reported using numeric uid, not userid.

UNIX vs. RACF signon

$cat /etc/password
#userid:password:UID:default-group:user-info:directory:shell
markh:cEiQk9zz:0:3:Mark Hahn x2601:/:bin/ksh

LISTUSER markh OMVS
USER=MARKH NAME=MARK HAHN x2601 OWNER=#SALES CREATED=96.015
DEFAULT GROUP=SALES PASSDATE=01.100 PASS-INTERVAL=30
ATTRIBUTES=SPECIAL OPERATIONS
REVOKE DATE=NONE
……
OMVS INFORMATION
UID=0
HOME=/user/markh
PROGRAM=/bin/ksh

Shell users

- Sign onto TSO
- Issue OMVS
- Line commands
  - $ <= std user
  - $su
  - # <= root user
  - #exit
  - $exit

- Sign onto TSO
- Issue ISHELL
- Primarily file system manipulation: edit / browse / maintain directories

Useful Shell commands

- **df** – display filesystem – what is mounted & where
  verifies automount, etc.
- `# df`
- `Mounted on Filesystem  Avail/Total    Files`
  `/u         (*AMD/u)        0/8          0`
- `/tmp       (SYS4.PDQ1.OMVS.TMP.HFS)`
- **who** – displays userid
- **whoami** – displays uid from lookup

Auditing

- RACFRW frozen BEFORE these appeared – RYO!
  - IRRADU00 (SMFDUMP exits)
  - OEM SMF reporters
- LOTS of new events: ??-?? (less a few)

```
09:36 INITOEDP  38   0 SYSPMH2
09:36 CHDIR     32   0 SYSPMH2
09:36 DACCESS   29   0 SYSPMH2
09:36 EXESETID  36   0 SYSPMH21
09:36 SETEUID   50   0 SYSPMH21
09:36 SETUID    52   0 SYSPMH21
09:36 TERMOEDP  39   0 SYSPMH22
```

RACF Support

- 7 new RACF classes
- DIRACC, DIRSRCH, FSOBJ, FSSEC, IPCOBJ, PROACT, PROCESS
  - No profiles allowed
  - set Logging and audit levels regardless of (in)active
- OMVS segments added to userid and group profiles
- 35 events added:
  - 28-nn - 58-nn and 60-nn - 64-nn
  - OMVS events fail with ee-nn -or- directory failures (28-01, 29-01)
  - Unlike traditional: ee-00 success and ee-nn fail
  - RACFRW does NOT support these events
    - IRRADU00 is IBM's recommended solution

Adding Users

- Determine if needs unique uid or default
- If unique, no automated control enforcing unique uid values.
- ISHELL helps

Auditing OMVS

- Multiple userids may be assigned same UID - no automated protection
- Use LISTUSER userid [NO]RACF OMVS to review user definition
  - UID(0) is the root or superuser - should be very restricted
  - HOME(directory) and PROGRAM(pgm) pose no special threat
- Use LISTGRP group [NO]RACF OMVS to review group definition
  - GID(0) means nothing other than group #0.
- MVS considerations
  - VLF for IRRUMAP and IRRGMAP (performance)
  - Review BPXPRMxx parameters (based upon MVS & OS/390 versions
  - IRRSXT00 is both pre and post processing exit
- RACFRW has no support for OMVS events
- IBM recommends building reports:
  - superuser report showing all UID(0) userids
  - cross-reference showing all UIDs and the GIDs connected
  - cross-reference showing all GIDs and connected UIDs

Search UNIXMAP Class

- Simple SR command shows all known Users and Groups
- For specifics (who assigned) issue RLIST for AUTHUSER (or ALL)
- Access list means nothing

```
G0
G1
G10
G12
G2
G3
G4
G999
U0
U411
U414
U415
U416
U417
U418
U419
U913
U999
```

UNIXMAP

- UNIXMAP is ***auto-maintained*** whenever UID or GID fields within OMVS segments updated.
- Confusion:
  - When uid is translated to userid, value may differ when uid shared, e.g. when uid(0) owns a file, OMVSKERN, BPXROOT, other uid(0) userid appears at different times.
  - Resolved in 2.10; until then, not a real problem

```
CLASS      NAME
-----      ----
UNIXMAP    U0
USER     ACCESS
FTPD1      NONE
DDF        NONE
SMTP       NONE
TCPIP      NONE
SYSLOGD    NONE
BPXROOT    NONE
SYSMSH     NONE
IBMUSER    NONE
```

Profiles

Authorization

- Define groups for authorization
- Connect users as needed
  - No profile refresh
- Easily reviewed:
  - LG: who is connected to group (has access)
  - LU: what groups is user part of (has access)

Tracking Profiles

- Authorize by groups:
  ODAEMON, OFTP, OMVS, OROOTGRP, etc.
- OMVSGRP owns all possible profiles
  - GROUP (flow of scope)
  - USER (also NAME('USS-…')?)
  - DATASET
  - General Resources: BPX.** in FACILITY, PROGRAM, STARTED, SURROGAT
- Using IRRDBU00 output, OWNER=OMVSGRP

User / Group Profiles

How to Identify / Track Unix System Services related profiles?

First thought: Name / Installation Data: “USS-…”

Follow-on: Have OMVSGRP own all possible profiles

```
BPXROOT  USS-BPX ROOT USER
FTPD1    USS-FTP DAEMON
ODFLTU   USS-DEFAULT UID
OMVSKERN USS-KERNEL

ODAEMON  USS-BPX.DAEMON ACCESS
ODFLTG   USS-DEFAULT GROUP (USER
OFTP     USS-FTP USERS
OMVS     USS-
OROOTGRP USS-BPX.SUPERUSER ACCESS
```

Users: NAME(‘ ‘)

Groups: DATA(‘ ‘)

Resource Profiles

- FACILITY BPX.DAEMON
- FACILITY BPX.DEFAULT.USER
- FACILITY BPX.SMF
- FACILITY BPX.SUPERUSER
- FACILITY BPX.**
- PROGRAM CRON
- PROGRAM INETD
- PROGRAM LM
- STARTED BPXOINIT.*
- STARTED BPXSTOP.*
- STARTED EZAZSSI.*
- SURROGAT OMVSKERN.SUBMIT

An easy way to find all profiles related to Unix System Services: have a group (OMVSGRP?) own all of them.

Daemons – from STARTED

```
STARTED  ETCINIT%.*
STARTED  EZAZSSI.*
STARTED  FTPD*.*
STARTED  PORTMAP.*
STARTED  SMTP.*
STARTED  SYSLOGD*.*
STARTED  TCPIP.*
```

JCL for Daemons

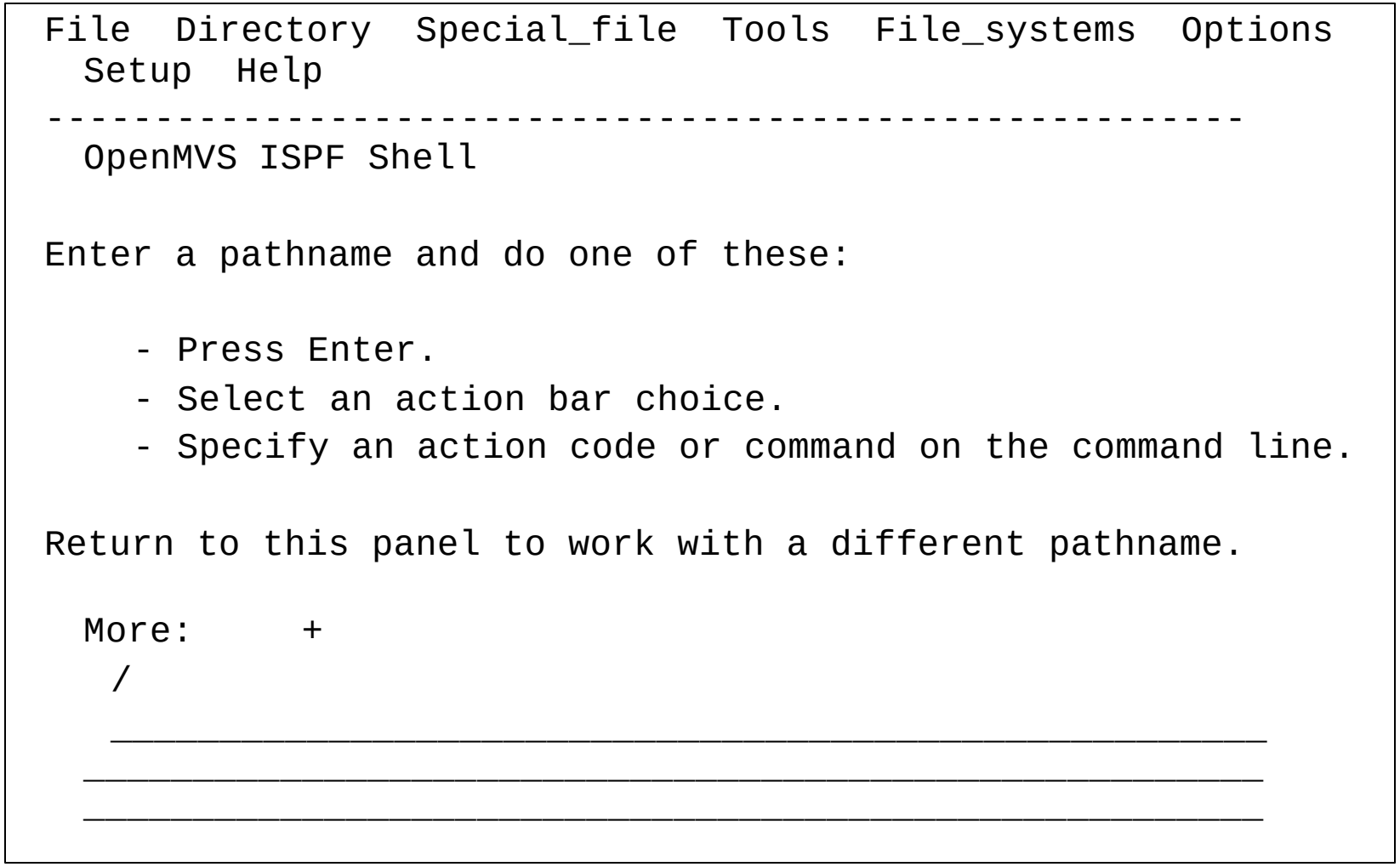
```
//SYSLOGD  PROC
//SYSLOGD  EXEC PGM=SYSLOGD,REGION=30M,TIME=NOLIMIT,
//      PARM='POSIX(ON) ALL31(ON)/ -d '
//*     PARM='/RPTSTG(ON),POSIX(ON) ALL31(ON)/ -d '
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSERR   DD SYSOUT=*

//TCPIP    PROC PARMS='CTRACE(CTIEZB00)',
//         UNIXPRM='ENVAR("RESOLVER_CONFIG=/etc/tcpip.data")'
//TCPIP    EXEC PGM=EZBTCPIP,
//             PARM='&UNIXPRM &PARMS',
//SYSTCPD  DD DISP=SHR,DSN=SYS3.comp.TCPIP.TCPIP.DATA
//PROFILE  DD DISP=SHR,DSN=SYS3.comp.TCPIP.PROFILE.TCPIP
```

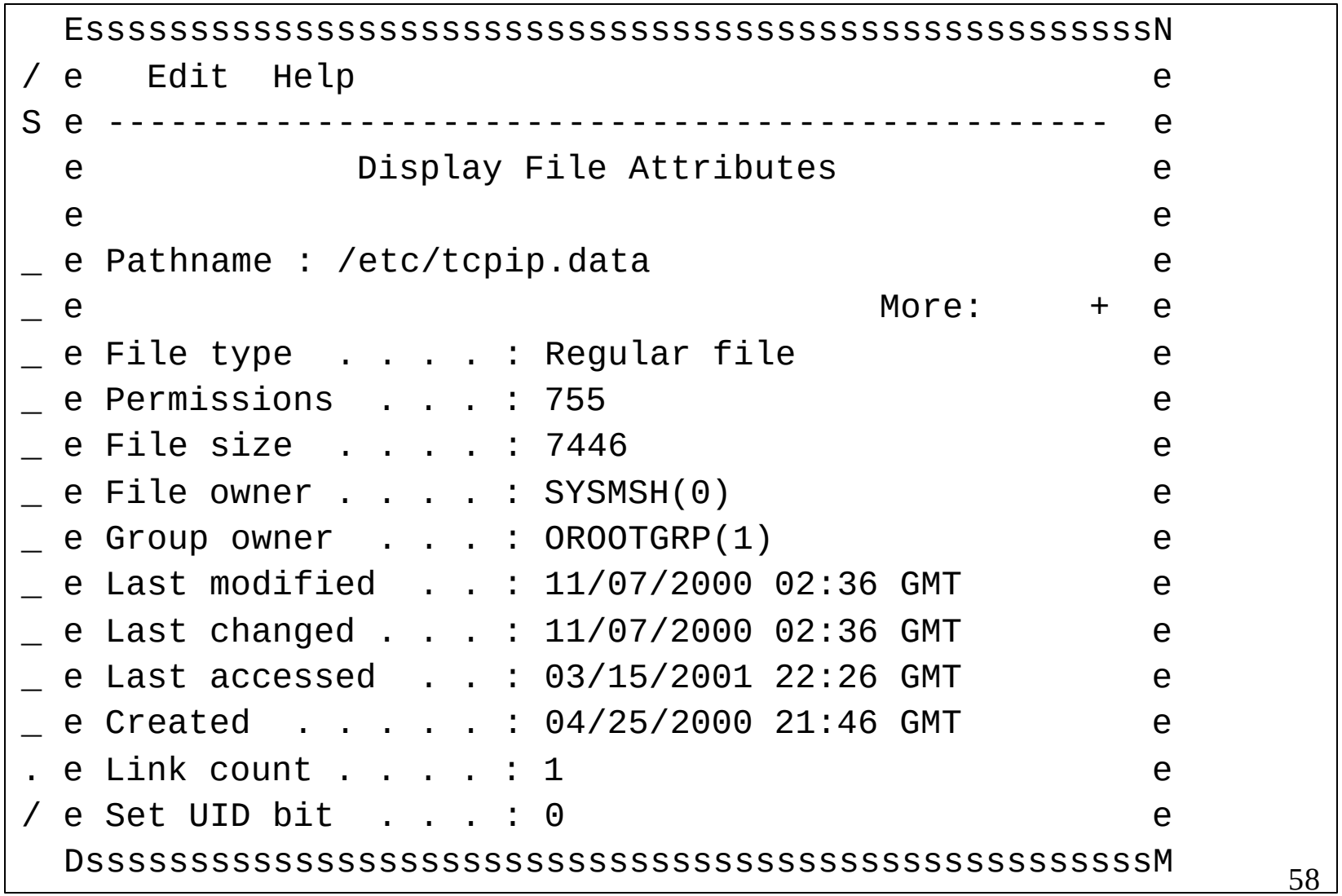
Program Properties Table

```
 BROWSE    SYS1.PARMLIB(SCHED00) - 01.00
 Command ===>
******************** Top of Data ****
MT SIZE(64K)
PPT PGMNAME(EPWINIT) NOCANCEL NOSWAP KEY(0)
    NODSI NOPASS NOPREF
PPT PGMNAME(EZBTCPIP) NOCANCEL KEY(6)
    NOSWAP PRIV SYST SPREF LPREF
PPT PGMNAME(EZAPPFS) NOSWAP KEY(1)
PPT PGMNAME(EZAPPAAA) NOSWAP KEY(1)
```

ISHELL - Entry



ISHELL - Attributes



OSHELL – file attributes

```
$ cd /etc
$ ls –al
. . .
-rw-r--r--   1 SYSMSH   OROOTGRP        2 Feb 18 13:45
   syslog.pid
-rwxr-xr-x   1 SYSMSH   OROOTGRP     7446 Nov  6 18:36
   tcpip.data
-rwx------   1 SYSMSH   OROOTGRP      149 Oct 16 13:05 u.map
. . .
# cat tcpip.data
;
;*************************************************
 ===>
```

ISHELL - browse

```
 BROWSE -- /etc/tcpip.data ------------- Line 00000000 Col 0
 Command ===>                                  Scroll ===
************************** Top of Data *************;
;*****************************************
;
;   Name of Data Set:     SYS3.comp.TCPIP.TCPIP.DATA
;
;   COPYRIGHT = NONE.
;
;
;OURMVSNAME:    HOSTNAME  OURTCPNAME
;YOURMVSNAME:   HOSTNAME  YOURTCPNAME
```

Profiles

- User profiles house
  - uid, home and program
  - xxxMAX overrides (2.8+)
- Group profiles
  - Own files
- Dataset profiles
  - HFS file protections

Profiles

- FACILITY
  - BPX.* - service controls, switches, defaults
- PROGRAM
  - Program controls – required by BPX.DAEMON
- SURROGAT
  - Non-zero uids submit uid(0) jobs

Profiles

- UNIXMAP (pre 2.10)
  - Map uid and gid values to USERID and GROUP
  - May have to populate if not active
- UNIXPRIV (2.8+)
  - Dole out SUPERUSER privileges
  - Set CHOWN switch

HFS – Hierarchical File System

- SYSTEM resource, not user
  recommend naming SYSx.OMVS.userid.HFS
- OMVS needs UPDATE
- Ensure no one has above UPDATE except DASD mgr

Special Userids

- As of 2.8: **irrcerta**, **irrmulti**, **irrsitec**
  - Have no default group
  - Defined in 2.8 Security Server Planning: Install and Migration
  - **irrmulti** documented in SYS1.SAMPLIB(irr40129) w/PTF ow40129

UNIX Related parmlib

- ALLOCxx – Allocation recovery
- COFVLFxx - VLF
- CTnBPXxx – tracing
- IEADMR00 – Dump gathering
- SMFPRMxx – specify timeouts
  When in OMVS, signal-enabled wait and exempt from JWT timeouts. Specify TMOUT in /etc/profile for initial timeout value, timeout USS THEN queue for JWT timeout.

SC28-1890-08

FACILITY Class Profiles

UNIX System Services

- BIG user!
- Define default UID/GID for “generic” UNIX users: FTP
  BPX.DEFAULT.USER
- Controls access to sensitive system resources
  - BPX.SUPERSER
  - BPX.SRV.userid
  - BPX.SMF
  - and more

What FACILITY Profiles Already Exist?

- SR CLASS(FACILTY) [NOMASK | MASK(bpx.*)]

```
BPX.DAEMON
BPX.DEFAULT.USER
BPX.SMF
BPX.SUPERUSER
```

- IRRDBU00

```
----+----1----+----2----+--

0505 BPX.DAEMON
0505 BPX.DAEMON
0500 BPX.DAEMON
0500 BPX.DEFAULT.USER
```

```
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
s
.IRRDBU00.DATA                          Line 00013881 Col 233 312
                                                 Scroll ===> CSR
5----+----6----+----7----+----8----+----9----+----0----+----1-
-
   GLOBAL   NO   017 1998-12-04 SYSPROG  1998-12-04 1998-12-04
   FACILITY SEC      READ     00000
   FACILITY STARTED  READ     00000
   FACILITY YES  019 1999-01-18 SEC      1999-01-18 1999-01-18
```

BPX.** profiles

- **BPX.DAEMON** – restricts access to daemon services
- **BPX.DEBUG** – restricts access to ptrace (via dbx) for debugging APF or server authority programs
- **BPX.FILEATTR.APF** – Authorizes users to set the APF attribute on HFS files. Similar to update access to SYS1.LINKLIB or SYS1.LPALIB
- **BPX.FILEATTR.PROGCTL** – controls setting program-controlled attribute … allows execution with high level of authority
- **BPX.FILEATTR.SHARELIB** – controls use of shared library region
- **BPX.JOBNAME** – who can set their own jobnames using _BPX_JOBNAME . (READ or higher)
- **BPX.SAFFASTPATH-** see separate foil
- **BPX.SERVER** – allows create/delete security environment for caller's thread; also determines authorization to access OS/390 resource

BPX.** profiles (2)

- **BPX.SMF** – whether a user may cut SMF records
- **BPX.STOR.SWAP** – which users may mark address spaces non-swappable
- **BPX.SUPERUSER** – make use of switch user (*su*) command
- **BPX.WLMSERVER** – controls access to WLM server functions

Special Function Profiles

- These set switches / store defaults, no access list
  - **BPX.DEFAULT.USER**
  - **BPX.SAFFASTPATH**

Adding a User

- Setup user for TSO <- *prerequisite*
- Select uid and assign: ALTUSER userid OMVS(UID(nnn) PROGRAM('/bin/sh') HOME('/u/userid') [other parms])
- LISTUSER userid [NORACF] OMVS
- MKDIR '/u/userid' – *if mount fails, recheck mkdir* <*>
- OSHELL chown userid /u/userid <*>
- Set OMVS segment user limit parms as needed
- Set OWNER if daemon id

- User unable to access home directory implies **chown** missing
- <*> ISHELL can be used

When You Return

When You Return (1)

- Review IEASYSxx and BPXPRMxx
- Consider OMVS=(0F,0L,MM) strategy
- Review BPXPRMxx members
  Consider extended parameters via OMVS segment
- Validate and review ALL UID(0)
- Validate and review ALL BPX.SUPERUSER
- Validate BPX.DEFAULT.USER – should NOT be (0)
- Review reporting capability – upgrade?
- Check GROUP naming, installation data and GROUP authorization
- OWNER(OMVSGRP) possibility

When You Return (2)

- SURROGAT for root id access (SMP jobs, etc)
- Review system userids: BPXROOT, default, OMVSKERN
- UNIXPRIV profiles? (2.8+) Validate list
- BPX.DAEMON, BPX.SUPERUSER, BPX.SRV.** in FACILITY (and others)
- BPX.DEFAULT.USER – audit usage: APAR OW33160 (R2.3-R2.7), APAR OW42092. Research
- 2.8+? PROTECT daemon userids

When You Return (3)

- Review BPXPRM0L (system limits) for globally high MAXxxx values and set xxxMAX in user OMVS segment (2.8).
- Review Application Identity Mapping (2.10) and plan migration.
- Check for BPXSTOP if shutdown not smooth.
  Also: Modify
  BPXOINIT,SHUTDOWN=FORKINIT

Using UNIX System Services

- Sign on to TSO first, enter ISPF
- Issue **OMVS**, **ISHELL**, other UNIX command (e.g. MOUNT, FTP, etc)
- To “enter” UNIX,
  - Userid must have OMVS segment
  - Default group must have OMVS segment
- To “use” UNIX services, Default info used if no segments
- Default users (BPX.DEFAULT.USER allows OMVS access)

Why uid(0) Sometimes

- (a) some services used to require UID(0), but don't any more. Original testing and documentation occurred using UID(0) for some servers, and without re-doing testing, etc. the product owners resist changing the documentation.
- (b) some services used to require UID(0), but now don't depending on certain other actions, perhaps dependent on the order of invocation of other services. Again, the test effort to confirm whether something doesn't need UID(0) and document that has a low priority compared to some other work.

Why uid(0) Sometimes (2)

- (c) some servers need UID(0) because of TCP/IP considerations (access to ports numbered below 1024) unless the customer wants to dedicate the port via TCP/IP configuration parms, so documenting a UID(0) requirement gives a simpler set of setup instructions.
- (d) some servers documented a need for UID(0), and we've found they don't need it. But they have changes in plan for the future that could again require UID(0) with no other possibility for workaround planned, so they don't want to document a change to a non-zero UID today, only to have to change back later if we can't invent some other workaround.

Thanks to Walt Farrell.

Ownership of Profiles

BPX.FILEATTR.PROGCTL needs to be owned by Security. It determines who can mark HFS programs as program controlled, which is the same kind of thing as doing RDEFINEs in the PROGRAM class. However, you will need to use PERMIT to give your system programmers who use SMP/E for installing UNIX programs READ access to the profile.

BPX.FILEATTR.APF could reasonably be owned by someone who can tell the system what the APF libraries are, e.g. the system programmer responsible for updating the APF list in SYS1.PARMLIB or for issuing SETPROG commands to change the APF list. Or you could have it owned by Security, and grant access to those system programmers who need it (including those you install UNIX programs using SMP/E).

BPX.FILEATTR.SHARELIB also seems like some sysprogs will need access. Since it's documented as "protecting against misuse of the shared region“ you will probably also want it owned by Security.

Useful Pieces

Resources

- UNIX System Services Planning SC28-1890
- Security in OS/390-Based TCP/IP Networks SG24-5383
- MVS-OE listserv: http://s390.ibm.com/oe/bpxa1dis.html
- UNIX System Services homepage
  - http://s390.ibm.com/unix/
  - Tools and Toys: http://s390.ibm.com/oe/bpxa1toy.html BPXSTOP
  - Conversion guide: http://publibfp.boulder.ibm.com/pubs/pdfs/os390/ich1m121.pdf
  - Important article: changing identities http://s390.ibm.com/oe/secure/chuid.html

Misc pix

```
$ whoami
ODFLTU
$ pwd
/
$ cd /u/SYSPMH
cd: /u/SYSPMH: EDC5111I Permission denied.

ICH408I USER(SYSPMH2 ) GROUP(MDV     ) NAME(MARK S. HAHN
   ) /u/SYSPMH CL(DIRSRCH )
   FID(01C8C6E2C1F1F10001290000000000003)
  INSUFFICIENT AUTHORITY TO LOOKUP
  ACCESS INTENT(--X)  ACCESS ALLOWED(OTHER ---)

SYSPMH2  09:36 MNTFSYS   44-00 SYSPMH2
SYSPMH2  09:36 DIRSRCH   28-01 SYSPMH2
```

Operator Tools

D OMVS

```
BPXO042I 12.43.03 DISPLAY OMVS 823
OMVS     000E ACTIVE          OMVS=(03)
```

D OMVS,A=ALL

```
BPXO040I 12.44.09 DISPLAY OMVS 829
OMVS     000E ACTIVE          OMVS=(03)
USER     JOBNAME  ASID PID PPID STATE   START
OMVSKERN BPXOINIT 00FB  1     0 MFI   13.44.19
LATCHWAITPID=    0 CMD=BPXPINPR SERVER=Init Process
  AF=    0 MF=65535 TYPE=FILE
```

<*snip*>

More Operator Tools

```
D OMVS,F
BPXO044I 12.48.31 DISPLAY OMVS 843
OMVS     000E ACTIVE          OMVS=(03)
TYPENAME   DEVICE ----------STATUS-----------
AUTOMNT         14 ACTIVE
  NAME=*AMD/u
  PATH=/u
HFS             11 ACTIVE
  NAME=SYS4.PDQ1.OMVS.TMP.HFS
  PATH=/tmp
```

Smooth shutdown

- Quiescing the system
  - JES won't shutdown until all work quiesced
  - BPXSTOP (from Tools and Toys page)
    Review carefully if sysplex sharing HFS's
  - MODIFY BPXOINIT,SHUTDOWN=FORKINIT
    Useful when nothing looks to be running

Abstract

It's a big enough challenge to audit what's happening within your everyday OS/390 system; add to it the growing UNIX environment and the challenges multiply. Not only is there a separate file system (with its own security issues (thankfully logged to SMF and formatted through RACF)), but there are new SUPERUSERS, DAEMONS, and all sorts of other creatures. There will be detailed informtion for the new Security Server facilities, including: OMVS segment and its new fields in 2.8, UNIXPRIV to parcel out the SuperUser (UID(0)) privileges, the numerous BPX.** profiles within the FACILITY class. Sample audit reports, checklists and justifications for these services will be included. You will leave this 1/2-day session able to access such available services as DSMON, RACFICE, TSO Commands, screen captures, and more to ease the burden of auditing UNIX events in your OS/390 system. This session will also be offering valuable SecureWay Security Server settings and web resources to make your job easier and more effective.

Thank You

Please fill out your eval sheets and enjoy your day.