# The World of Program Control and PADS

**Walt Farrell, CISSP**
**z/OS Security Server Design**
**IBM Corporation MS P388**
**2455 South Road**
**Poughkeepsie, NY 12601**
**(845) 435-7750**
**wfarrell@us.ibm.com**

RUG Mar. 6, 2003



## Disclaimer

The information contained in this document is distributed on an "as is" basis without any warranty either express or implied. The customer is responsible for use of this information and/or implementation of any techniques mentioned. IBM has reviewed the information for accuracy, but there is no guarantee that customers using the information or techniques will obtain the same or similar results in their own operational environments.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed programs may be used. Functionally equivalent programs may be used instead. Any performance data contained in this document was determined in a controlled environment and therefore, the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environments.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM Products, programming or services in your country.

IBM retains the title to the copyright in this paper as well as title to the copyright in all underlying works. IBM retains the right to make derivative works and to republish and distribute this paper to whomever it chooses.




## Trademarks

- **The following are trademarks or registered trademarks of the International Business Machines Corporation:**
  - IBM, z/OS, OS/390, RACF
- **Other company, product, and service names may be trademarks or service marks of others.**



## Agenda

- **Basic Concepts**
- **Simple Program Control**
- **PADS (Program Access to Data Sets)**
  - Before z/OS R4
  - z/OS R4 in BASIC Program Security Mode
- **EXECUTE Control**
- **z/OS R4 ENHANCED Program Security Mode**
  - Effects on PADS, EXECUTE, & UNIX
  - PADS
  - Migration

# Basic Concepts

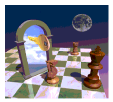


## Basic Concepts

- **Program:**
  - A compiled program in load module or program object format
  - Loaded from LPA, LINKLIST, or private library (JOBLIB, STEPLIB)
  - **NOT:**
    - CLIST, REXX exec, Java, PERL, etc.
    - Program loaded from the UNIX file system (secured and processed differently)
- **Environment:**
  - **job step in a batch job, started task, or started job**
  - **TSO session**
    - Also, a TSO command, CLIST, or REXX exec invoked by TSOEXEC or IKJEFTSR
  - **UNIX address space**



## Basic Concepts...



- **Clean Environment**
  - **An environment (see preceding list) in which all programs that have run are:**
    - **defined in the PROGRAM class; or**
    - **loaded from LPA; or**
    - **loaded from the UNIX file system and marked as program controlled by**

      **extattr +p ...**
  - **Required for some (not all) program control functions**
    - **PADS (program access to data sets)**
    - **EXECUTE**
    - **UNIX server/daemon functions**
  - **Generally easier to maintain in batch or STC or UNIX**
  - **Harder to maintain in TSO, especially for ISPF**



## Basic Concepts...

- **Program Security (PGMSECURITY) Modes**
  - **BASIC**
    - **Only mode before z/OS R4**
  - **ENHANCED**
    - **New, higher security mode for z/OS R4**
    - **More resistant to malicious users and hackers**

# Simple Program Control



## Simple Program Control (Basic or Enhanced mode)

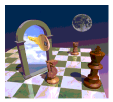
- **Functions:**
  - restrict selected users / groups from running a program
  - audit use of programs
- **Define program via RDEFINE PROGRAM**
- **Permit READ to allow; NONE to restrict**
- **Format:**

```
RDEFINE PROGRAM name +
   ADDMEM('library.name'[/[volser][/[NO]PADCHK]] +
   UACC(NONE or READ)
```

  - volser may be omitted
    - or an actual volser
    - or '******' to represent the IPL volume
  - WARNING will not have any effect



## Simple Program Control (Basic or Enhanced mode)...



- **Example:**

```
RDEFINE PROGRAM DELUSER +
   ADDMEM('SYS1.LINKLIB'//NOPADCHK +
   UACC(READ)

PERMIT DELUSER CLASS(PROGRAM) +
   ID(group1) ACCESS(NONE)
```

- **If program has an alias, protect it, too:**

```
RDEFINE PROGRAM DU +
   ADDMEM('SYS1.LINKLIB'//NOPADCHK +
   UACC(READ)

PERMIT DU CLASS(PROGRAM) +
   ID(group1) ACCESS(NONE)
```



## Simple Program Control (Basic or Enhanced mode)...

- **Recommendations:**
  - Omit the volser when defining programs
  - Usually specify NOPADCHK
  - Use READ or NONE. Avoid EXECUTE unless
    - program contains a sensitive algorithm; or
    - program contains sensitive data
  - Restrict UPDATE access to libraries that contain protected programs, just as for APF libraries
- **Notes:**
  - Program name in RDEFINE can have * at end
    - ABC* protects all programs with names beginning ABC... that are in the library named in the ADDMEM
    - Profiles are discrete, not generic. Some differences:
      - ✓ RLIST PROGRAM ABCD will not display ABC*
      - ✓ A* may be more specific than ABC* depending on ADDMEM value

## Simple Program Control (Basic or Enhanced mode)...

- **Notes, continued:**
  - ADDMEM can specify multiple members
    - allows protection for copies of program in other libraries
    - all ADDMEM operands for a PROGRAM profile should have same PADCHK or NOPADCHK specification
  - Protected programs can reside in
    - public libraries
      - ✓Libraries in the system LINKLIST
    - Private libraries
      - ✓Not in the system LINKLIST
      - ✓Accessed by JOBLIB, STEPLIB, ISPLLIB, etc. or by TSO command CALL 'library.name(program)'
  - They cannot reside in LPA. You can define them, but system does not check user's authority to run them (more later)
  - Specify SETR WHEN(PROGRAM) to activate program control
  - Specify SETR WHEN(PROGRAM) REFRESH after making changes to PROGRAM profiles
  - SETR CLASSACT(PROGRAM) has no effect

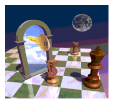


## Simple Program Control (Basic or Enhanced mode)...

- **Notes, continued:**
  - Consider how the library that contains the program is protected
  - Usually allow READ to the program libraries via DATASET profiles and READ to the programs via PROGRAM profiles
    - Most programs operate on data; protect the data (example: IEBGENER, AMASPZAP)
    - Programs needing APF won't work if user copies them to another library
    - Programs using PADS won't work if user copies them to another library
  - NONE can work for program libraries:
    - Programs accessed via system LINKLIST (not STEPLIB, JOBLIB,etc)
    - Programs run under TSO as commands or via
      CALL *(program)
      but not via
      CALL 'library(program)'



## Program Control by System ID (Basic or Enhanced mode)



- **Function: Restricts access to programs based on user / group and system**
- **Uses the (usually) 4-character SMF ID and conditional access lists**
- **Example:**
  - Assume program ABC in library ABC.LOAD
  - You have two systems, PROD and TEST
  - Most Users should only run ABC on the TEST system

```
RDEFINE PROGRAM ABC +
    ADDMEM('ABC.LOAD'//NOPADCHK) UACC(NONE)

PERMIT ABC CLASS(PROGRAM)  +
    ID(*) ACCESS(READ) WHEN(SYSID(TEST))
```



## PADS (Program Access to Data Sets)

## Clean Environments (Basic or Enhanced mode)

- **Users need a clean environment for:**
  - PADS (Program Access to Data Sets: WHEN(PROGRAM(...))
  - EXECUTE access to PROGRAMs or libraries
- **UNIX daemons and servers need a clean environment if you have FACILITY BPX.DAEMON defined**
- **Without this requirement, users could easily bypass security controls if you use PADS or EXECUTE**
- **Creating a clean environment for a user: Ensure that all programs the user runs are:**
  - Defined by PROGRAM profiles; or
  - Loaded from the LPA
- **Problem: How to do that?**



## Clean Environments (Basic or Enhanced mode)...

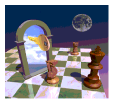
- **One method: Figure out each program the user needs and define a separate PROGRAM profile for it.**
  - Very difficult to figure out
  - Many profiles
  - Massive administrative overhead
- **Recommended method:**
  - RDEFINE PROGRAM ** UACC(READ)
  - ADDMEM at least:

'SYS1.LINKLIB'//NOPADCHK,
'SYS1.CMDLIB'//NOPADCHK,
'tcpip.SEZALINK'//NOPADCHK,
'ftp.userexits'//NOPADCHK,
'db2.DSNEXIT'//NOPADCHK

'SYS1.MIGLIB'//NOPADCHK,
'cee.version.SCEERUN'//NOPADCHK,
'tcpip.SEZATCP'//NOPADCHK,
'db2.DSNLOAD'//NOPADCHK,



## Clean Environments (Basic or Enhanced mode)...



- **Recommended method (continued):**
  - RDEFINE PROGRAM ICHDSM00 + ADDMEM('SYS1.LINKLIB'//NOPADCHK) UACC(NONE)
  - RDEFINE PROGRAM IRRDPI00 + ADDMEM('SYS1.LINKLIB'//NOPADCHK) UACC(NONE)
- **Notes:**
  - UACC(READ) is recommended, and appropriate. With PROGRAM ** you are merely defining everything to keep the environment clean.
    - ID(*) ACCESS(READ) is not the same as UACC(READ)
    - In z/OS R4, or with APAR OW50327, RACF will use UACC(READ) when loading from SYS1.LINKLIB using PROGRAM ** or PROGRAM *
  - For actual **protection** of programs, use separate PROGRAM profiles with desired UACC and access list
  - Potential problems still in TSO. After users run their own programs,
    - May need to use TSOEXEC to use PADS or EXECUTE
    - Or logoff and logon again



## PADS Before z/OS R4

- **PADS (Program Access to Data Sets) allows access to data only when a user is running a particular program**
  - The specified program provides an extra layer of security by controlling what the user does with the data. It
    - Restricts what the user can read, omitting some data
    - Restricts what the user can write, validating the user's data
  - Terminology: the program **mediates** the user's access to the data
  - Uses WHEN(PROGRAM(xyz)) as a conditional access list entry
- **Example 1: To let users read a data set when running program XYZ:**

```
ADDSD 'some.data.set.profile' UACC(NONE)
PERMIT 'some.data.set.profile' ID(*) +
  ACCESS(READ) WHEN(PROGRAM(XYZ))
```

## PADS Before z/OS R4...

- **Notes:**
  - Environment must be clean
  - XYZ must be the program that issues the OPEN
  - Specify exact program name; * not allowed
  - If other non-LPA programs are active in the environment they must be
    - defined with NOPADCHK; or
    - added to the conditional access list with their own WHEN(PROGRAM(...))
- **Example 2: Users run program ABC1, and ABC1 invokes (LINK, ATTACH) ABC2. ABC2 OPENs the data set. ABC1 is defined with NOPADCHK.**

  **PERMIT 'some.data.set.profile' ID(*) + ACCESS(READ) WHEN(PROGRAM(ABC2))**



## PADS Before z/OS R4...

- **Example 3: Users run program ABC1, and ABC1 invokes (LINK, ATTACH) ABC2. ABC2 OPENs the data set. ABC1 is defined with PADCHK.**

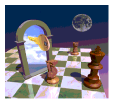
  **PERMIT 'some.data.set.profile' ID(*) ACCESS(READ) + WHEN(PROGRAM(ABC2))**

  **PERMIT 'some.data.set.profile' ID(*) ACCESS(READ) + WHEN(PROGRAM(ABC1))**

- **Problem for Example 2 and 3: Administrator must know design of program**
  - User runs ABC1
  - Administrator must permit access via ABC2, and possibly ABC1
  - Solved by z/OS R4



## PADS In z/OS R4 (BASIC mode)



- **Considering only BASIC Program Security Mode for now... ENHANCED described later**
- **You can use same conditional access list as before z/OS R4**
- **Or, with all systems on R4, you can simplify PADS example 2: Users run program ABC1, and ABC1 invokes (LINK, ATTACH) ABC2. ABC2 OPENs the data set. ABC1 is defined with NOPADCHK.**

  **PERMIT 'some.data.set.profile' ID(*) ACCESS(READ) + WHEN(PROGRAM(ABC1))**

  **(Permitting ABC2 would also work, as before R4)**
- **This resolves the problem, as the administrator doesn't need to know details of the application design**



## PADS In z/OS R4 (BASIC Mode)...

- **z/OS R4 also allows more flexible security**
- **Example: You want a system programmer to update SYS1.LINKLIB only via SMP/E**

  **PERMIT 'SYS1.LINKLIB' GENERIC + ID(SYSPROG) ACCESS(UPDATE) + WHEN(PROGRAM(GIMSMP))**

  - This assumes that you have defined GIMSMP explicitly or via PROGRAM **, and that you have defined the utilities it will invoke. (Remember, environment must still be clean for PADS)
  - In contrast with example 2 previously, you do not need to PERMIT the other programs that GIMSMP invokes.
- **z/OS R4 Security Administrator's Guide has more examples and scenarios (JCL, TSO, ISPF, REXX, ...)**

## PADS (Any Release)

- **PADS works for granting READ and UPDATE**
- **PADS generally does not work for ALTER (data set creation or deletion)**
  - The user's program is not running during allocation / deallocation, which happen before / after the user's program runs
  - Exception: Dynamic Allocation
- **Example: User executes program ABC and you want to allow creation of data set ABC.DATA**

```
//stepname EXEC PGM=ABC
//DD1        DD    DSN=ABC.DATA,DISP=(NEW ...
```

- **You can not use**

```
PERMIT 'ABC.DATA' GENERIC ID(user1) +
    ACCESS(ALTER) WHEN(PROGRAM(ABC))
```

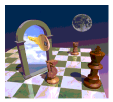


## PADS (Any Release)...

- **You can:**
  - Write a new program ABC1
  - Have ABC1 dynamically create the data set, rather than using JCL
  - Then have ABC1 LINK, ATTACH, or XCTL to ABC
  - Modify the JCL to run ABC1, and to delete the DD statement.
- **//stepname EXEC PGM=ABC1**
  - PERMIT 'ABC.DATA' ID(user1) +
    ACCESS(ALTER) WHEN(PROGRAM(ABC1))
- **NOPADCHK vs PADCHK:**
  - Use NOPADCHK for programs you trust not to try to bypass PADS protection
  - Consider PADCHK if you have a user-created program that you have defined in the PROGRAM class but don't completely trust
  - Use same value for all ADDMEM operands in a PROGRAM profile



## PADS (Any Release)...



- **PADCHK example:**
  - user needs to run user-created program XYZ, and also use other programs for PADS.
  - you must define XYZ in PROGRAM class to keep environment clean and allow PADS (e.g., in TSO or ISPF split-screen)
  - you have not completely verified the processing that XYZ does, so you don't completely trust it
- **Steps:**
  - If possible, copy XYZ to a library the user cannot update
  - RDEFINE PROGRAM XYZ UACC(READ or NONE) +
    ADDMEM('library'//PADCHK)
  - For each data set using PADS, where you have specified
    PERMIT ... WHEN(PROGRAM(ABC))
    and you want to allow user to have XYZ active while the data set is OPEN, also specify
    PERMIT ... WHEN(PROGRAM(XYZ))



## EXECUTE Control

## EXECUTE Control (Pre-R4, and R4 BASIC)

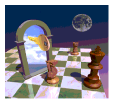
- **Purpose: Use when you must prevent the user from copying a program or viewing it**
  - Generally because the program
    - Contains sensitive data
    - Contains sensitive algorithms
- **Use ACCESS(EXECUTE) or UACC(EXECUTE)**
  - For DATASET profile protecting a private (non-LINKLIST) library; or
  - For PROGRAM profile that protects one or more programs
- **User needs at least EXECUTE authority to run a program; READ is easier to setup; Avoid EXECUTE if possible**
- **Use of EXECUTE requires a clean environment**
  - For EXECUTE with a DATASET profile, ensure all programs in the data set have PROGRAM profiles



## EXECUTE Control (Pre-R4, and R4 BASIC)...

- **EXECUTE on PROGRAM profile:**
  - Useful for programs in LINKLIST
    - Specify ACCESS(NONE) on DATASET profile to prevent user from OPENing data set to copy or view the program
    - Specify ACCESS(EXECUTE) on specific PROGRAM profile for the program to
      - ✓prevent dumps via SYSUDUMP, SYSABEND
      - ✓require loading into a clean environment (to prevent viewing from an active user program)
- **EXECUTE on DATASET profile:**
  - Most useful for private (non-LINKLIST) libraries
    - ACCESS(NONE) would prevent user from accessing the programs via JOBLIB, STEPLIB, ISPLLIB, etc.
  - Allows user to OPEN the library but
  - Prevents user from accessing the library except via LINK, LOAD, XCTL, ATTACH, or other supervisor-state processing
    - Thus user cannot copy the program, or view it via Browse, etc.





## z/OS R4 ENHANCED PGMSECURITY



## z/OS R4: ENHANCED Program Security Mode

- **z/OS R4 offers two program security (PGMSECURITY) modes that affect PADS and EXECUTE**
  - BASIC -- Functions like pre-R4 (but with PADS enhancement)
  - ENHANCED -- Better security and resistance to hackers and malicious users
- **ENHANCED mode offers a WARNING option for migration**
  - Performs all checks as though in ENHANCED mode
  - If checks fail, but would have worked in BASIC mode
    - Issues messages, creates SMF record
    - Allows function to continue successfully

## z/OS R4: ENHANCED Program Security Mode...

- **FACILITY profile IRR.PGMSECURITY controls the mode**
  - BASIC:
    - IRR.PGMSECURITY not defined; or
    - RDEFINE FACILITY IRR.PGMSECURITY APPLDATA('BASIC')
  - ENHANCED:
    - RDEFINE FACILITY IRR.PGMSECURITY APPLDATA('ENHANCED')
  - ENHANCED with WARNING
    - RDEFINE FACILITY IRR.PGMSECURITY APPLDATA('anything else')
- **RACF does not validate the APPLDATA value during RDEFINE or RALTER, but inspects it during SETR WHEN(PROGRAM) [REFRESH] processing**
- **SETROPTS LIST shows the mode:**
  **WHEN(PROGRAM -- BASIC)**
  **WHEN(PROGRAM -- ENHANCED)**
  **WHEN(PROGRAM -- ENHANCED WARNING)**

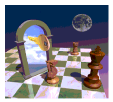


## z/OS R4: ENHANCED Program Security Mode...

- **ENHANCED mode supports 3 kinds of PROGRAM definitions:**
  - BASIC:
    - A PROGRAM profile protecting 1 program name (no * at end) with APPLDATA('BASIC')
  - MAIN:
    - A PROGRAM profile protecting 1 program name (no * at end) with APPLDATA('MAIN')
  - Normal:
    - Any PROGRAM profile with a * in the name, or with any other APPLDATA value
- **RACF does not validate APPLDATA value during RDEFINE or RALTER**



## z/OS R4: ENHANCED Program Security Mode...



- **ENHANCED mode affects PADS and EXECUTE processing**
  - PADS and EXECUTE work only if
    - a program with the MAIN attribute established the environment
    - the current program, or the first program executed in the current task or a parent task, has the BASIC attribute
- **Also affects UNIX server and daemon processing if you define FACILITY profile BPX.MAINCHECK**
  - Server and daemon functions work only if a MAIN program established the environment
    - May require copying some programs from UNIX file system to load library



## z/OS R4: ENHANCED Program Security Mode...

- **PADS Example 1: The user executes program ABC via JCL**
  **// EXEC PGM=ABC**
  **or via TSO**
  **TSOEXEC ABC**
  **or**
  **TSOEXEC CALL *(ABC)**
- **You want to allow the user to read data set ABC.DATA while running ABC**
- **You can define ABC as a MAIN program**
  - RDEFINE PROGRAM ABC + ADDMEM('load.library'//NOPADCHK) APPLDATA('MAIN')
  - PERMIT 'ABC.DATA' ID(*) ACCESS(READ) WHEN(PROGRAM(ABC))

## z/OS R4: ENHANCED Program Security Mode...

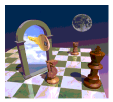
- PADS Example 2: The user executes program ABC in TSO:
  CALL 'load.library(ABC)'
  or
  ISPEXEC SELECT PGM(ABC)
- You want to allow the user to read data set ABC.DATA while running ABC
- ABC will not create the environment (IKJEFT01 did that) so you cannot define ABC as a MAIN program. Instead, define it as BASIC
  - RDEFINE PROGRAM ABC +
    ADDMEM('load.library'//NOPADCHK) APPLDATA('BASIC')
  - PERMIT 'ABC.DATA' ID(*) ACCESS(READ) WHEN(PROGRAM(ABC))
- Avoid use of BASIC where possible; MAIN is more secure
  - For some TSO (or other) users, MAIN cannot work, so for those cases you must use BASIC.



## z/OS R4: ENHANCED Program Security Mode...

- z/OS R4 Security Administrator's Guide contains more examples and scenarios
- Avoid use of BASIC where possible; MAIN is more secure
  - For some TSO (or other) users, MAIN cannot work, so for those cases you must use BASIC.
- Most controlled programs should not have (or need) MAIN or BASIC attributes
- New ICETOOL report will show programs with MAIN or BASIC
- Auditors may want to question
  - use of BASIC PGMSECURITY mode
  - use of programs with BASIC attribute
    - However, some cases require either
      - ✓ BASIC attribute or
      - ✓ Not using PADS or EXECUTE





## Migration to ENHANCED Mode



## Migration to ENHANCED PGMSECURITY Mode

- Figure out which programs need MAIN or BASIC attributes
  - Examine conditional access lists via IRRDBU00 0402 records
    - For each program found, determine whether users actually execute it, or some other program
    - Also determine intended method of execution (batch, TSO) and for TSO whether user can use TSOEXEC
    - This will give initial list of MAIN and BASIC candidates
  - Examine IRRDBU00 records 0400, 0402, 0404 to find EXECUTE-controlled data sets
    - Examine programs in those data sets to see which need MAIN or BASIC, based on intended usage
  - Similarly, examinine IRRDBU00 records 0500, 0505 to find EXECUTE-controlled programs
- Use RDEFINE to define new profiles or RALTER to add APPLDATA('MAIN') or APPLDATA('BASIC') as needed

## Migration to ENHANCED PGMSECURITY Mode...

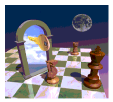
- **RDEFINE FACILITY IRR.PGMSECURITY APPLDATA('ENHWARN')**
  - Any APPLDATA value except BASIC or MAIN gives Enhanced-Warning mode
  - SETR RACLIST(FACILITY) REFRESH if needed
  - SETR WHEN(PROGRAM) REFRESH
- **z/OS R4 now in ENHANCED-WARNING mode**
  - has no effect on pre-R4 systems
  - has no effect on jobs, STCs, TSO sessions that are already running
- **Watch messages and collect SMF records**
- **Refine list of PROGRAM profiles that need MAIN or BASIC attributes**
- **IPL at least once to test jobs, STCs, and TSO users who were active when you enabled ENHANCED-WARNING**
- **Repeat as needed**



## Migration to ENHANCED PGMSECURITY Mode...

- **Switch to ENHANCED mode**
  - RALTER FACILITY IRR.PGMSECURITY APPLDATA('ENHANCED')
  - SETR RACLIST(FACILITY) REFRESH if needed
  - SETR WHEN(PROGRAM) REFRESH
  - Does not affect running jobs, STCs, or TSO users
- **Watch for any problems**
  - If any, fix by adjusting PROGRAM profiles or IRR.PGMSECURITY profile
- **IPL to make change effective for all jobs, STCs, and TSO users**





## Appendix



## IBM Manuals

- **On the web, in either PDF or Book Manager formats:**
  - Security Server for OS/390 V2R10:
    - http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ICH1K132
  - Security Server for z/OS R4:
    - http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/ICHZBK30
  - OS/390 library: http://www.ibm.com/servers/s390/os390/bkserv/index.html
  - z/OS library: http://www.ibm.com/servers/eserver/zseries/zos/bkserv/
- **Quick Message Lookup:**
  **http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html**
- **Redbooks:**
  **http://www.ibm.com/redbooks**

## RACF Home Page

- **http://www.ibm.com/servers/eserver/zseries/zos/racf/**
  - Latest release information on RACF
  - Links to announcement letters
  - Sample code and tools
  - Frequently Asked Questions
  - RACF user group information
  - RACF-L information
  - Presentations on RACF-related topic



## OS/390 Security Home Page

- **http://www.ibm.com/servers/eserver/zseries/zos/security**
  - Overview of security concepts, including animations
  - Overview of S/390, zSeries, OS/390, and z/OS security functions
  - Links to related web sites for OS/390 and z/OS components