JES2 RACF Calls, Control Points, and Profiles

NY RACF Users Group
Thursday May 3, 2007

Tom Wasik
JES2 Design/Development/Service
Poughkeepsie, NY
wasik@us.ibm.com

Trademarks

Overview

- **Authentication processing**
  - Input phase job authentication
  - RJE sign on
  - NJE sign on
- **Protection of resources**
  - Data sets on SPOOL
  - Operator commands
  - Output destinations
- **JOBs and SYSOUT from NJE**
  - NODES class
- **JES2 installation exits**

Concepts and terms

- **Profile vs Entity names**
  - Profile is what you define in RACF, entity name is what we pass to RACF that is matched to a profile
- **Submitter vs Owner**
  - Submitter is the identity of the process that placed a job in an internal reader, Owner is the identity assigned to the job.
- **JES2 is "just the messenger".**
  - At least as far as user authentication/assignment
  - JES2 does not assign a userid to a job
  - JES2 transports the RACF TOKEN to AUTH calls

RACFVARS &RACLNDE

- **RACFVARS class &RACLNDE profile**
  - Defines the JES2 local NJE NODE to RACF
  - Place node name(s) in ADDMEM for profile
  - Multiple names useful if
    - node name changed
    - multiple nodes share a single RACF data base
- **Example**
  - Local node is NY but was NEWYORK
  - RALT RACFVARS &RACLNDE ADDMEM(NY NEWYORK)
- **Ideas for use?**
  - In **EVERY** profile that has a node name
  - E.g. SUBMIT.&RACLNDE.*jobname*.*owner_userid*
    - JESJOBS profile to control submit
  - Cheap now, expensive later

JOB Input Processing

- **The RACF controls**
  - JES-BATCHALLRACF and JES-XBMALLRACF
    - SETR options
  - JESINPUT, JESJOBS, PROPCNTL, SURROGAT, SECLABEL, NODES
    - RACF Classes
- **TSO SUBMIT controls (not discussed here)**
  - TSO Exits (eg. IKJEFF53)
  - JCL option (in UADS or clear PSCBJCL via exit)

JOB Input processing

- **JES2 issues a VERIFYX SAF call**
  - This obtains a SAF TOKEN that represents the job
    - This token has owner, submitter, and other info
  - All information about the job is passed to SAF
    - No checks are made directly by JES2
- **Information passed to SAF (not a complete list)**
  - Submitter information
    - token, userid, group
  - JOB card information
    - user, password (new and old), group, seclabel, jobname, accounting string
  - Source information
    - Input device (POE), submitting node
  - Session type
    - INTBATCH, NJEBATCH, RJEBATCH, EXTBATCH
    - INTXBM, NJEXBM, RJEXBM, EXTXBM

JOB Input processing

- **JES-BATCHALLRACF and JES-XBMALLRACF**
  - Determines if a userid must be assigned to a job
  - BATCH is for most batch jobs
  - XBM is for an older way to submit jobs call execution batch monitor
    - Not used much
    - Requires JES2 JOBCLASS(*n*) to have XBM= set
- **SETR BATCHALLRACF**
  - All jobs MUST have a valid userid
- **SETR NOBATCHALLRACF**
  - Jobs can run without a userid

JOB Input processing

- **JESINPUT class**
  - Controls who can submit jobs using an input device
  - Entity name passed on the check is the name of the input device
    - Examples: INTRDR, RDR2, R15.RD1, *nodename*
  - The owner of the job is checked for READ access to the profile
- **Example**
  - User BILL submits job with USER=HILLARY using INTRDR
  - Job submitter is BILL, job owner is HILLARY
  - Check is does HILLARY have read access to profile matching INTRDR
- **Ideas for use?**
  - User GEORGE should never be used with jobs from an RJE
  - Create profile R*.RD* and do not permit user GEORGE to it
  - Even if you know the password, you cannot submit a GEORGE job from an RJE
  - Does not work for “privileged” userids

JOB Input processing

- **JESJOBS class**
  - Controls who can submit jobs by name and userid
  - Entity name passed on the check is:
    - SUBMIT.*node*.*jobname*.*owner_userid*
  - The submitter is checked for read access to the profile
  - Can give conditional access based on **submitter**’s POE
    - WHEN(JESINPUT(…..))
    - WHEN(TERMINAL(…..))
    - Do not confuse with POE of job being submitted.
  - Does not apply to RJE and card readers
    - No real submitter in these cases
    - Session EXTBATCH, RJEBATCH, EXTXBM and RJEXBM

JOB Input processing

- **Example**
  - User BILL submits job with USER=HILLARY and a jobname of FIRSTMAN (at node NY)
  - Job submitter is BILL, job owner is HILLARY
  - Check is does BILL have read access to profile matching SUBMIT.NY.FIRSTMAN.HILLARY
- **Ideas for use?**
  - This profile can limit
    - What users (submitter or owner) can use a particular jobname
    - Who can submit a job with a particular userid
    - Restrict user, jobname, and submitter combinations that are not allowed.
    - Cannot restrict “privileged" submitter but can restrict privileged userids as owners

JOB Input processing

- **PROPCNTL class**
  - Controls a userid can be propagated from a submitter to submitted jobs
  - Profile names are the userid that should not propagate
  - The existence of a profile is all that is checked
- **Example**
  - User HILLARY does not want her userid propagated to jobs she submits
  - Profile HILLARY is created in PROPCNTL class
  - Jobs submitted by HILLARY without USER= and PASSWORD= are not assigned a userid
  - Jobs fail authentication if BATCHALLRACF is active
- **Ideas for use?**
  - Prevent the scheduling package userid from being assigned to jobs it submits.

JOB Input processing

- **SURROGAT class**
  - Controls whether a job can be submitted with a USER= and no password
  - Entity name is *execution_userid*.SUBMIT
  - The check is if the submitting userid has READ access to the profile
- **Example**
  - User BILL submits jobs with USER=HILLARY and no password
  - Profile HILLARY.SUBMIT is checked in SURROGAT class
  - If userid BILL has read access the job runs
- **Ideas for use?**
  - One common use is scheduler programs to allow jobs to be submitted with application userid and no need for a password

JOB Input processing

- **SECLABEL class**
  - Controls Mandatory Access Control (MAC)
    - Protects data based on categories/levels
    - Contrast with protection by profiles (DAC)
  - Profile name is the seclabel
  - Actual use is beyond the scope of this presentation

JOB Input processing

- **NODES class**
  - Used to determine submitter information for NJE jobs
  - Profiles of the form *node.keyword.value* where keyword is
    - USERJ, GROUPJ, SECLJ
  - UACC controls processing, ADDMEM controls translation
- **Example**
  - Job arrives from userid BILL at node DC
  - Profile DC.USERJ.BILL exists with UACC(UPDATE) and ADDMEM(HILLARY)
  - Owning userid is determined as job was submitted by local userid HILLARY
    - If UACC(NONE) job fails validation
    - If UACC(READ) submitter is “unknown” userid
    - If no ADDMEM, then submitter is local userid BILL
  - Similar processing for group and SECLABEL
- **Ideas for use?**
  - Control processing for jobs arriving from NJE

RJE Signon processing

- **JES2 does a VERIFYX at RJE signon**
  - Passes RJE name as userid, local node as EXENODE, password(s) from RJE signon, and SESSION is RJEOPER
  - No passwords if
    - BSC dedicated line
    - SNA autologon or $S RMTnnnn
- **RACF processing**
  - Look for facility class profile RJE.*rmtname*
    - Not found, RJE used JES2 signon rules
    - Found then RACF builds token using RMT name as userid
- **Example**
  - RMT123 signs on
    - Facility class profile is RJE.RMT123. Userid is RMT123
- **Note**
  - Remote userids are treated as submitters of jobs, but they cannot propagate

NJE Signon processing

- **Secure form of NJE signon**
  - Exchanges DES-encrypted passwords in I/J signon records
  - Controlled by **SIGNON=SECURE|COMPAT** on **NODE** statement
  - Uses **APPCLU** class in RACF/SAF
    - Entity is **NJE.*node1.node2***
    - Uses **SESSKEY** associated with profile for encryption
  - Can be used by BSC, SNA or TCP/IP
- **NODE(n) PASSWORD=** is **<u>not</u>** used when **SIGNON=SECURE**

NJE Signon processing

- **Support for SSL/TLS with TCP/IP**
  - Application Transparent TLS support in TCP/IP (AT-TLS) used
    - All definitions for SSL/TLS are in TCP/IP policy definitions, not JES definitions
    - Future standards for TLS will automatically be supported
  - Controled by **SECURE=YES/NO** on **SOCKET** statement

Protecting data on SPOOL

- **JESSPOOL class**
  - Protects data sets on SPOOL
  - Normal entity name is
    - *node.userid.jobname.jobid*.D*dsnum.dsname*
    - Eg. POK.IBMUSER.IBMUSERY.JOB00017.D0000101.?
  - READ/UPDATE access needed to access data
- **AUTH calls made by JES2 processing**
- **SAF/RACF token of owning job passed as resource token (RTOKEN)**
  - Allows owner to access data without need for profile
- **Also used to “audit” data set creation/deletion**
  - Calls pass RTOKEN=UTOKEN so check passes but cuts SMF record

Protecting data on SPOOL

- **Special case JESSPOOL calls**
  - *node.user.jobname.jobid.specialname*
    - Only 5 qualifiers instead of 6
    - *Specialname* is JESMSGLG, JESYSMSG, JCL
    - Used for SPOOL browse to access logical data sets
  - *node*.+TEMP*+.jobname.jobid*.D*dsnum*.SYSIN
    - SYSIN data sets prior to conversion processing
    - Used for SPOOL browse
  - *node.user.jobname.jobid*.GROUP.*groupname*
    - This form is used by SWB modify and SDSF

Protecting data on SPOOL

- **Application using APIs need JESSPOOL access**
  - PSO (external writer and conversation)
    - If just looking, READ access
    - If altering attributes or purging, ALTER access
  - SAPI
    - Indicate if only looking (SSS2SRON) READ access
      - Error if trying to update attributes or purge
    - Otherwise UPDATE access
  - SPOOL browse
    - Always READ access
- **Special case check for “RECEIVE”**
    - AUTH call passes destination userid to SAF/RACF
      - RECVR= keyword
    - If RECVR userid matches requestor userid, access granted
      - No profile checked
    - Available in PSO, SPOOL browse, and SAPI

Protecting data on SPOOL

- **JESJOBS class**
  - Protect jobs from being CANCELed via TSO cancel SSI (JES2 check)
  - Entity name
    - CANCEL.*node.owner_userid.jobname*
  - Check that SSI caller has ALTER access to this profile
  - Job's SAF/RACF token passed as RTOKEN
    - Allows user to cancel jobs they own
  - **NOTE:** Order of *owner_userid.jobname* is not consistent with SUBMIT form of check
- **Example**
  - Userid HILLARY uses TSO CANCEL for job HUBBY submitted by Userid BILL (node is NY)
  - Check if userid HILLARY has ALTER access to profile matching CANCEL.NY.BILL.HUBBY
  - If HILLARY has access, cancel is allowed
- **TSO exit also available to control**
  - IKJEFF53 gets control for the CANCEL TSO command

Protecting data on SPOOL

- **SWB modify SSI**
  - Alters OUTPUT JCL characteristics for a JOE
    - Can add, delete, or alter SWBs
  - Special JESSPOOL class profile used
    - *node.user.jobname.jobid.*GROUP.*groupname*
  - Option to use ISFAUTH class check
    - ISFAUTH.DEST.*sysout_destination*
  - Both checks pass RTOKEN and UTOKEN
  - These forms are used by SWB modify and SDSF
  - ALTER access is needed

Operator Commands

- **JES2 uses the CMDAUTH service to validate operator commands**
  - CMDAUTH makes checks in the OPERCMDS class
- **Format of JES2 entity names is**
  - *jesx.verb.object*
  - *jesx* is the subsystem name
  - *verb* is the action (DISPLAY, MODIFY, START, etc)
  - *object* is the high level object (BAT, BATOUT, NJEDEF, etc)
- **Object is not specific to a particular job**
  - JES2 has “owner” checking at job/SYSOUT level

Operator Commands

- **User associated with command depends on source**
  - From a MCS (SVC 34, console, etc) the SAF/RACF token of the issuer is used
  - From a batch job (/*command JECL) the submitter of the command (owner of device)
  - From //COMMAND JCL the owner of the job
  - JES2 automatic commands the issuer of the command
  - JES2 init deck token associated with JES2 address space
  - RJE is JES2 token if RJE is not logged on via RACF, or the RJE userid if logged on with RACF
  - NJE is JES2 token if source node not defined to RACF, or the NJE node name id defined to RACF

Operator Commands

- **Token associated with commands from NJE**
  - Userids (and SAF/RACF tokens) can be associated with an NJE node.
  - JES2 does a VERIFYX when command is received
    - EXENODE is local node, SNODE is source node, USERID is source node and SESSION is NJEOPER.
  - RACF processing attempts to get a USERID
    - Checks for facility class profile NJE.*source_node*
    - Not found, no NJE signon, commands use JES2 authority
    - Found, check NODES class profile *snode*.RUSER.*snode*
      - Found with UACC=NONE, fail VERIFYX (and command)
      - Found with other UACC, continue VERIFYX
    - Build token using USERID = source node
      - If USERID exists an is valid, pass back token
- **Example profiles**
  - For command source of NY
    - Facility class of NJE.NY, NODES class NY.RUSER.NY, and a USERID of NY

Protecting Output destinations

- **WRITER class**
  - Controls where output can be sent
  - Entity names are
    - *jesx.devtype.devname*
  - *jesx* is the JES2 subsystem name
  - *devtype* is the type of device (LOCAL, RJE, NJE)
  - *devname* is the JES2 name of the device
  - SYSOUT owner needs READ access to profile
  - Node portion of destination is checked when data set is created
    - if not the local node
  - For local output, check for LOCAL or RJE device is made when output is printed
  - Check for access to node also made in notify user SSI
- **EXAMPLE**
  - User BILL wants to printer on LOCAL PRT3
  - Check is made of JES2.LOCAL.PRT3
  - If BILL has access, output will print

NJE JOBs and SYSOUT

- **VERIFYX done when JOBs/SYSOUT arrive**
  - Pass information from NJE headers
  - For jobs pass JOB card information
  - POE is the adjacent node name
  - Session for SYSOUT is NJSYSOUT
  - Session for jobs is NJEBATCH or NJEXBM
  - NODES class checks and others are internal to RACF

NJE JOBs and SYSOUT

- **NODES class controls inbound SYSOUT and JOBs**
  - Three ways to control things (profile names)
    - USERID, GROUP, SECLABEL
  - Three levels of “trust” (UACC value)
    - Accept information and trust authentication
    - Accept information but re-validate
    - Reject job or SYSOUT
  - Can accept information as is or with translation (ADDMEM data)
- **NODES class checks internal to RACF**
  - Occurs on VERIFYX and some AUTH calls
- **Goals of NODES class processing**
  - Determine submitter information for JOBs
  - Determine OWNER information for SYSOUT
- **Nodes can be uplevel or downlevel**
  - Uplevel nodes provide security tokens in NJE headers
  - Downlevel nodes do not provide security information
- **Also applies to SPOOL reload processing**

NJE JOBs and SYSOUT

- **For JOBs arriving via NJE the profiles are:**
  - *node*.USERJ.*userid*
  - *node*.GROUPJ.*group*
  - *node*.SECLJ.*seclabel*
- **In all cases the node is the origin node for the job stream**
- **UACC for USERIDs controls how job is processed**
  - READ - accept information but submitter is blank (as if from a card reader)
  - UPDATE – accept information from uplevel nodes and translate if needed
    - Same as READ for downlevel nodes
  - CONTROL – accept information from all node types
  - NONE – Accept the job from the sending node but delete it.
- **UACC for SECLABEL and GROUP control rejection and translation**
  - READ - Accept value and translate if needed
  - NONE – Accept the job from the sending node but delete it.

NJE JOBs and SYSOUT

- **EXAMPLES**
  - NODE1.USERJ.* UACC(READ)
    - Accept all jobs from node 1 but re-validate
  - NODE1.USERJ.* UACC(UPDATE)
    - Accept jobs and use submitter information in header as local submitter
  - NODE1.USERJ.* UACC(UPDATE) ADDMEM(UNODE1)
    - Translate all jobs from NODE1 to have a submitting userid of UNODE1
    - UNODE1 can then be used to control access JESJOBS resources based on the origin being NODE1
  - NODE1.SECLJ.CONF UACC(NONE)
    - Reject all CONF seclabel jobs from NODE1

NJE JOBs and SYSOUT

- **For SYSOUT arriving via NJE the profiles are:**
  - *node*.USERS.*userid*
  - *node*.GROUPS.*group*
  - *node*.SECLS.*seclabel*
- **In all cases the node is the execution node for the job stream**
- **UACC for USERIDs controls how SYSOUT is processed**
  - READ – Owner is NJE default user
  - UPDATE –Uplevel nodes owner is creator and translate if needed
    - Same as READ for downlevel nodes
  - CONTROL – Same as update for all node types
  - NONE – Accept the SYSOUT from the sending node but delete it.
- **UACC for SECLABEL and GROUP control rejection and translation**
  - READ - Accept value and translate if needed
  - NONE – Accept the SYSOUT from the sending node but delete it.

NJE JOBs and SYSOUT

- **Translation of “&SUSER” in USERS profiles**
  - Assigns ownership to the submitting userid
  - Applies to all UACCs except NONE
  - Allows jobs submitted locally (according to &RACLNDE), executed on remote node with SYSOUT returning locally, to be assigned the submitters userid
    - *.USERS.* UACC(READ) ADDMEM(&SUSER)
  - If not local, a second lookup is done using the submitting information
  - Second lookup is used to assign owner
    - UACC of CONTROL required in second lookup
    - Only if there is translation
    - A second &SUSER results in submitter userid kept

NJE JOBs and SYSOUT

- **Examples**
  - NODE1.USERS.* UACC(READD)
    - Accept SYSOUT and assign ownership to default userid
  - NODE1.USERS.* UACC(READ) ADDMEM(&SUSER)
    - Accept SYSOUT and if the submitting node is in &RACLNDE, assign ownership to the submitting userid
  - NODE1.USERS.* UACC(UPDATE) ADDMEM(UNODE1)
    - Assign ownership of all SYSOUT from NODE1 to user UNODE1
  - NODE1.SECLS.CONF UACC(NONE)
    - Delete all CONF SECLABEL SYSOUT from NODE1

NJE JOBs and SYSOUT

- **Protection at intermediate nodes**
  - SYSOUT or job going from A to C through B
  - B is a store and forward node
  - NODES class profiles do not apply to store and forward
  - SYSOUT and JOB protected by “NJE userid”
  - Only exposure is as POE for inbound NJE data
    - POE for jobs and SYSOUT from NJE at final destination is the adjacent node
    - In case above POE is B
  - If concerned, define critical nodes as DIRECT=YES
    - Will only send to direct node if adjacent
    - Must have direct connections to DIRECT=YES nodes

Security related exits

- **Pre and post SAF exits (36 and 37)**
  - Called in the USER environment
  - Parameters include
    - SAF/RACF or CMDAUTH parameter list
    - $WAVE data area
      - Function code in WAVE
    - Related JES2 control block (and eyecatcher)
  - Can bypass call, alter parameter, change return code, perform additional checks
  - Advantage over SAF exits is access to JES2 data areas

Security related exits

- **Exit 36/36 FUNCODES**
  - Function codes are specified on $SEAS call
  - Describe reason for SAF call
  - Aids in deciding what function to do in the exit
- **Exit 36 example outline**
  - FUNCODE=$SEAINIT is VERIFYX for job create
  - CB passed is $SAFINFO
  - If SFIDEVTP=DCTRJR job is from RJE
  - Use VERIFYX MF=M to set SESSION=INTBATCH

Security related exits

- **Exit 38**
  - Called when data set can never be received
  - User does not have access to a SECLABEL that dominates the SYSOUT
  - Exit can reroute data set or log problem
- **Exit 39/55**
  - Called when NJE VERIFYX indicates to delete data
  - Can override delete processing
  - Useful when first implementing NJE security