z/OS 2.4 IBM Education Assistant (IEA)

Solution (Epic) Name: Improved PassTicket Security

Element(s)/Component(s): RACF

Agenda

- Trademarks
- Session Objectives
- Overview
- Usage & Invocation
- Interactions & Dependencies
- Migration & Coexistence Considerations
- Installation
- Session Summary
- Appendix

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.
- Additional Trademarks:
  - None

Session Objectives

- What is a PassTicket
- How can PassTicket keys be protected?
- Why you should always use encrypted keys
- How we are making it easier for you to do that

Overview

- Who (Audience)
  - RACF security administrators and auditors
- What (Solution)
  - RACF provides a function to convert masked PassTicket keys to encrypted keys and better interoperability with ICSF
- Wow (Benefit / Value, Need Addressed)
  - Strengthen security of PassTicket keys without having to change them
  - Start performing PassTicket key management tasks using ICSF, in a manner consistent with Pervasive Encryption
  - Easily prove compliance by being able to report on the protection mechanism for PassTicket keys

Overview …

- What is a PassTicket?
  - A one-time-use password substitute based on a user ID, an application name, the current time, and a shared symmetric key between the application generating the PassTicket and the application (RACF) evaluating it
- Keys are defined in the SSIGNON segment of a PTKTDATA profile
  - The KEYMASKED option results in a masked key in the RACF database
  - The KEYENCRYPTED option stores a key token, encrypted under the CKDS master key, in the ICSF CKDS and saves the resulting key label in the RACF database
- Examples:
  - `RDEFINE PTKTDATA MYAPPL SSIGNON(KEYMASKED(1234567812345678))`
  - `RDEFINE PTKTDATA MYAPPL SSIGNON(KEYENCRYPTED(1234567812345678))`

Overview …

- If a PassTicket key is compromised, an attacker can logon to any user who is authorized to use that application.
- You should never used masked keys outside of a test environment.
- You wanna see something really scary?

  - https://github.com/bigendiansmalls/passticket-tools/blob/master/unmask_passticket.py

Usage & Invocation

- Two new SSIGNON keywords are being introduced
  - ENCRYPTKEY – converts a KEYMASKED key to a KEYENCRYPTED key in place
  - KEYLABEL – Specified a pre-existing CKDS key label to use

- These will be described in more detail, but first we describe how to determine the method currently used to protect a key

Usage & Invocation … How do I know?

- RLIST tells you the format. For KEYENCRYPTED, RLIST previously gave no information about the key label that RACF generated for you.
  - `KEYMASKED DATA NOT DISPLAYABLE`
  - `KEYENCRYPTED DATA NOT DISPLAYABLE`
- In 2.4, RLIST can also indicate the presence of a key token (see next slide)
  - `KEYTOKEN DATA NOT DISPLAYABLE`
- And, RLIST will display the key label for KEYENCRYPTED
  - `KEYENCRYPTED LABEL: IRR.SSIGNON.SY1.07192018.185056.915782`
- Which externalizes the format of the label, which was not previously documented
  - So, you can now recognize PassTicket keys when using ICSF's CKDS browser (introduced in HCR77C1)
  - However, this does not help to associate them with PTKTDATA profiles

Usage & Invocation … How do I know?

- Additional reporting mechanisms will indicate the method of protection, and if KEYENCRYPTED, the key label name
  - IRRDBU00 starts creating general resource record type 0530, which contains
    - *MASKED* if KEYMASKED keyword was used
    - *KEYTOKEN* if KEYENCRYPTED was used, and the key exists within a key token, perhaps due to an error with ICSF
    - Key label name if KEYLABEL or KEYENCRYPTED was used, and the key is stored in ICSF with a key label. The output is the label name, which is a 64-character value padded with blanks if necessary.
    - *UNKNOWN* if the format of the data is unrecognized. (This is an unexpected error condition)
  - R_admin general resource profile extract starts emitting a SSIGNON segment with a KEYLABEL field containing the same as above
    - IRRXUTIL gets this for free
- Now you have several ways of demonstrating compliance!

Usage & Invocation

- Use IRRXUTIL to find your KEYMASKED keys

```
/* REXX */
/*********************************************************************
 Purpose: Display the names of PTKTDATA class profiles that use
          masked PassTicket keys.

 Authorization required:
          - READ access to IRR.RADMIN.RLIST in FACILITY plus
            authority to list the profile's SSIGNON segment
*********************************************************************/
RES.PROFILE = ' '
RES.GENERIC = "FALSE"
say "The following PTKTDATA profiles used KEYMASKED keys:"
Do Until (myrc = '12 12 4 4 4')
  myrc=IRRXUTIL("EXTRACTN","PTKTDATA",RES.PROFILE,"RES",,RES.GENERIC)
  If myrc /= '0 0 0 0 0'   &,
     myrc /= '12 12 4 4 4' Then Do
    say "An error occurred:" myrc
    exit 1
  End
  If RES.SSIGNON.KEYLABEL.1 = "*MASKED*" Then
    say "  " RES.PROFILE
End
say "All finished!"
```

Usage & Invocation

- The new ENCRYPTKEY keyword can be used to move a masked key into ICSF:
  - `RALTER PTKTDATA MYAPPL SSIGNON(ENCRYPTKEY)`
- To do this in bulk:
  - `SEARCH CLASS(PTKTDATA)`
    `CLIST('RALTER PTKTDATA ' ' SSIGNON(ENCRYPTKEY)')`
  - Edit `'MYUSER.EXEC.RACF.CLIST'` to your satisfaction
  - `EXEC 'MYUSER.EXEC.RACF.CLIST'`

  - When the profile already contains a key label, a message will let you know, and the label will remain unchanged
    - `IRR52254I The encrypted key found in the MYAPPL profile in the PTKTDATA class cannot be further encrypted using ENCRYPTKEY. Command processing is terminated.`

- ENCRYPTKEY will also move a key token into the CKDS and replace it with a key label in the SSIGNON segment
- This is not reversable. As usual, make a backup first.

Usage & Invocation

- “But I want to do all my key management using ICSF, using my own key label naming conventions, like I can do for Pervasive Encryption!”

- New KEYLABEL keyword:
  - `RALTER PTKTDATA MYAPPL SSIGNON(KEYLABEL(IRR.PASSTICKET.MYAPPL.KEY))`

- You can now securely define (DES) keys to ICSF and simply tell RACF what key label to use for a given PTKTDATA class profile
- Your naming convention can associate the key label with the PTKTDATA profile name

Usage & Invocation

- Note that you cannot both convert a key and specify the key label you wish to use.
- Options:
  1. If you like our labels, keep using KEYENCRYPTED for key changes.
  2. Convert the key and use it until you wish to change it. Then, define a new key in ICSF and use KEYLABEL to store the label you specified in RACF.
  3. Convert the key and then use ICSF's Key Generator Utility Program (KGUP) to rename the key, then store the label in RACF.
- Be aware of key distribution/timing issues to avoid application outages

Usage & Invocation … ICSF considerations

- Keep in mind the consistency of RACF and ICSF ‘domains’ when using ICSF (these are not new considerations)
- The RACF database on any given system will use the CKDS on that system when generating or evaluating a PassTicket.
  - If other systems sharing the RACF database use a different CKDS, you must use ICSF services to export the key from the system on which the commands specifying SSIGNON(KEYENCRYPTPED(...)) or SSIGNON(ENCRYPTKEY) were executed to the CKDS of all systems sharing the RACF database.
    - PassTicket operations will not work on systems which lack the proper key in the CKDS.
  - If SSIGNON segment updates are propagated via RRSF, the CKDS on the target RRSF system (RRSF Main) is updated properly by RACF. However, if that target is the MAIN system of a multisystem node, you must use ICSF services to export the key from the MAIN system’s CKDS to the CKDS of all non-MAIN systems in that node.

Usage & Invocation … ICSF considerations

- Also not new: ICSF authorization
  - Users specifying SSIGNON(KEYENCRYPTED(..)) and SSIGNON(ENCRYPTKEY) must have at least READ authority to the following CSFSERV services:
    - CSNBCKI – Clear key import
    - CSNBKRC – CKDS key record create
    - CSNBKRW – CKDS key record write
    - CSNBKRD – CKDS key record delete
      - may be called to clean up if CSNBKRW fails.
  - Also requires READ authority to CSFKEYS profile IRR.SSIGNON.*sysid*.########.#######.#######
  - The same authority is required on target systems if the command is propagated via RRSF.

Usage & Invocation

- RACF currently requires you to copy a number of ICSF modules into the link pack area (LPA) in order to use KEYENCRYPTED
- Failure to do so can result in a difficult to diagnose error when generating a PassTicket

- With 2.4, RACF no longer requires this step!

- And, speaking of diagnostics …

Usage & Invocation … enhanced diagnostics

- When a PassTicket does not evaluate, it can be difficult to debug
- Original MFA support created SMF relocate 443 for RACROUTE VERIFY records. It indicates whether a PassTicket was used for authentication and whether it was successful. When not, internal return/reason codes for failure also appear in relocate 443. Good! But it still needs more…
- Relocate 443 will now also contain
  - New reason codes indicating how far from validity (time offset) the PassTicket was
  - The application name used in the evaluation process
    - If the caller of RACROUTE REQUEST=VERIFY specified an application name, this is it (and it is redundant with an existing relocate section that also contains it)
    - But when a value is not passed in, one is derived internally, and this is the difficult part to debug (note that the process ***is*** documented)

Usage & Invocation … enhanced diagnostics

- When the PassTicket generation service anchored in RCVTPTGN fails, it can be difficult to debug
- Register 0 will now contain a reason code

Usage & Invocation … dirty laundry exposed

- Along the way, we discovered a couple of “features”
  - Sometimes KEYENCRYPTED doesn’t complete, and a key token is stored in the PTKTDATA profile instead of a key label
    - It’s still encrypted under the CKDS master key, and the function continues to work
  - Sometimes KEYENCRYPTED creates two key labels in the CKDS.
    - One is useless, but harmless. Can use ICSF key auditing to detect which keys are being used.
- OA56729/OA56831 will fix this on 2.3 (only). Will also include:
  - KEYLABEL keyword support
  - ENCRYPTKEY keyword support
  - RLIST enhancement

  - OA56829 is a PER APAR. OA56831 is a new function SPE. They are both associated with the same PTF (TBD).

Interactions & Dependencies

- To exploit this item, all systems in the Plex must be at the new z/OS level: No

- Software Dependencies
  - None
- Hardware Dependencies
  - None
- Exploiters
  - None

Migration & Coexistence Considerations

- None
- But note that if sharing a RACF database with a downlevel system, you can issue RLIST from an uplevel system to see a key label

Installation

- List anything that a customer needs to be aware of during installation and include **examples** where appropriate - clients appreciate these:
  - None

Session Summary

- Encrypted keys GOOD.  Masked keys BAD. 

Appendix

- z/OS Security Server RACF Security Administrator's Guide
  - Overview and general setup instructions are in the "**Protecting general resources**" chapter, under the "**Using the secured signon function**" heading
- z/OS Security Server RACF Command Language Reference
  - RDEFINE and RALTER SSIGNON segment keywords
- z/OS Security Server RACF Callable Services
  - **Appendix B** R_admin general resource tables updated with new keywords and description of KEYLABEL field contents for profile-extract
- z/OS Security Server RACF Macros and Interfaces
  - New IRRDBU00 record type for SSIGNON segment
  - New reason codes for PassTicket-generate service