RACF Password Security Enhancements (OA43998/OA43999)

Agenda for this presentation

- Trademarks
- Session Objectives
- Overview of the new functions
- Interactions & Dependencies
- Migration & Coexistence Considerations
- Session Summary
- Affected publications

Trademarks

- See url http://www.ibm.com/legal/copytrade.shtml for a list of trademarks.

- Additional trademarks:
  - None.

Session Objectives

- Understand the new enhancements being made for password security:
  - Stronger encryption algorithm for passwords and password phrases
  - Special character support for passwords
  - Support for users to have a password phrase without a password
  - Ability to expire a password/phrase without having to change its value
  - Password/phrase history clean-up function
- These functions are available on z/OS V1R12 and higher releases via:
  - SAF APAR OA43998
  - RACF APAR OA43999
  - Get documentation at:
    ftp://public.dhe.ibm.com/eserver/zseries/zos/racf/pdf/oa43999.pdf

Overview – password/phrase encryption

- Problem Statement / Need Addressed
  - DES does not stand up to modern password cracking attempts in the event that a copy of the RACF database is exfiltrated
    - Graphics Processing Units are screamingly fast, can be used in parallel, and are now viable economically to normal consumers
    - RACF password cracking tools, including John the Ripper, are freely available on the internet
- Solution
  - Beef up the password/phrase encryption algorithm used
- Benefit / Value
  - Defense in depth (You still need to carefully protect copies of the RACF database!)
  - Demonstrate to an auditor your compliance with regard to stronger encryption

Overview – the KDFAES algorithm

- Key Derivation Function with AES

1)Start with:
  - DES hash for passwords – maintains upward compatibility in some cases
  - Clear-text password phrase

2) Append random text (salt)

3) Iteratively hash (SHA256) this text a (large) number of times to derive a 256-bit AES key
  - This step is intentionally slowing down the encryption process!

4) Encrypt the RACF user ID with the AES key

Overview – the algorithm ...

- Similar in principle to bcrypt
- Salting the password makes every instance of a RACF password unique and considerably increases the work factor in cracking the password
  - Defeats “Rainbow Tables”, which are pre-computed hashes of dictionary words and well-known password values which are simply compared against stolen password hashes
  - Salting the password forces the attacker to pre-compute the table for every **instance** of a password hash
- Iteratively hashing it requires the brute-force attack to perform the entire computation – no shortcuts
- Designed for longevity – as CPUs get faster, the number of iterations can be increased

Usage & Invocation

- Enable the new algorithm with

  `SETROPTS PASSWORD(ALGORITHM(KDFAES))`

  - No ICHDEX01 exit required to enable it!

- Confirm this with SETROPTS LIST

```
PASSWORD PROCESSING OPTIONS:
  THE ACTIVE PASSWORD ENCRYPTION ALGORITHM IS KDFAES
  PASSWORD CHANGE INTERVAL IS  60 DAYS.
  PASSWORD MINIMUM CHANGE INTERVAL IS   3 DAYS.
  MIXED CASE PASSWORD SUPPORT IS IN EFFECT
  SPECIAL CHARACTERS ARE ALLOWED.
  10 GENERATIONS OF PREVIOUS PASSWORDS BEING MAINTAINED.
  AFTER   5 CONSECUTIVE UNSUCCESSFUL PASSWORD ATTEMPTS,
      A USERID WILL BE REVOKED.
  PASSWORD EXPIRATION WARNING LEVEL IS  15 DAYS.
  INSTALLATION PASSWORD SYNTAX RULES:
    RULE 1  LENGTH(8)      xxxxxxxx
   LEGEND:
    A-ALPHA C-CONSONANT L-ALPHANUM N-NUMERIC V-VOWEL W-NOVOWEL *-ANYTHING
    c-MIXED CONSONANT m-MIXED NUMERIC v-MIXED VOWEL $-NATIONAL s-SPECIAL
    x-MIXED ALL
```

Usage & Invocation ...

- After enabling,
  - For unchanged passwords/phrases/history, RACF will evaluate them using “legacy” rules (with the one exception that the masking algorithm will never be used)
  - When a password/phrase is changed, KDFAES will be used
  - New history entries will be created in KDFAES format

Usage & Invocation … Performance

- Password evaluations are more computationally intensive than DES
  - **ACEE caching in VLF will be crucial** for subsequent LOGONs
- Same for password changes, only even more so
  - Evaluate current password
  - Encrypt new password
  - Compare new password against password history
    - Where every history entry requires an additional encryption
  - If you tend to have password expirations clustered around certain dates (e.g. end of quarter), and if this becomes a problem, you can use the new password expiration function to get them more evenly distributed
- Keep an eye on RACF database growth due to increased size of password/history
  - Could result in fragmentation which can be defragged by IRRUT400

Usage & Invocation … conversion from DES

- I enabled KDFAES, but I want all my passwords to be stronger **NOW!**
- A system-SPECIAL user can convert a user's password and history in-place, without requiring them to be changed, with

```
ALTUSER userid PWCONVERT
```

  - Or, to do it in bulk

```
SEARCH CLASS(USER) CLIST('ALTUSER ' ' PWCONVERT') NOLIST
EX 'prefix.EXEC.RACF.CLIST'
```

  - Password phrases and phrase history are **not** converted
- If you have passwords/phrases/history which are currently masked, or encrypted under an installation-defined method (by use of an ICHDEX01 exit), the conversion will result in unusable entries. The current password/phrase must be changed, and history entries will be ineffective.

Usage & Invocation … RACF Database Unload (IRRDBU00)

- The Database Unload utility (IRRDBU00) is being enhanced to provide additional password-related information:
  - The algorithm used to encrypt the current password
  - The algorithm used to encrypt the current phrase
  - The number of legacy password history entries
  - The number of KDFAES password history entries
  - The number of legacy password phrase history entries
  - The number of KDFAES password phrase history entries
- Possible uses:
  - Demonstrate to an auditor that all your current passwords/phrases are encrypted under KDFAES
    - Sample query provided in SYS1.SAMPLIB(RACDBUQR)
  - Identify users in need of PWCLEAN/PWCONVERT
    - When using SEARCH/CLIST, you will at least **reference** all USER profiles regardless of whether they actually need to be changed

Usage & Invocation … exit considerations

- The new password exit (ICHPWX01) will no longer receive the history array when KDFAES is enabled. This is only a consideration if you have an ICHDEX01 exit performing its own encryption. You probably don't.

- If you have a new phrase exit (ICHPWX11) for the sole purpose of allowing a phrase from 9 to 13 characters, this will not be necessary when KDFAES is enabled. That restriction is relaxed under the new algorithm.
  - If you are using the IBM provided sample, it requires an update (included in the APAR) to relax the restriction

- If you have an encryption exit (ICHDEX01/11) whose sole purpose is to tell RACF to use DES, and never masking, then this exit will no longer be necessary after enabling KDFAES.
  - But keep it around for a little while in case you need to fall back

Usage & Invocation … RACF downloads on the RACF web site

- CUTPWHIS – clean up password history
  - Continues to work prior to activating KDFAES, but will cause harm thereafter. Made obsolete by ALTUSER PWCLEAN. No code update is planned, but the web page will be updated
- PWDCOPY – copy passwords from one RACF database to another
  - Continues to work prior to activating KDFAES on source or target, but will cause harm thereafter. An update is planned.
- IRRXUTIL – Various sample REXX execs using IRRXUTIL
  - XSETRPWD reports on password-related SETROPTS options. This is updated to report on KDFAES enablement.

Overview – special characters

- Problem Statement / Need Addressed
  - Better password quality is needed, both in terms of the number of characters allowed, and the syntactical constraints that can be enforced
- Solution
  - Allow an additional 14 characters to be specified in passwords
  - Enhance password syntax rules such that a mix of character types can be enforced
- Benefit / Value
  - Defense in depth. A brute-force offline password attack will need to try that many more password combinations (You still need to carefully protect copies of the RACF database!)

Overview ...

- Support 14 additional characters
  - Currently, there are 65 possible password characters if mixed-case is in effect
    - 65**8 = 318,644,812,890,625 possible 8-char passwords
  - With the additional 14 characters
    - 79**8 =1,517,108,809,906,561

Overview … the cast of characters

Hexadecimal value	Symbol*
4B	.
4C	<
4E	+
4F	|
50	&
5A	!
5C	*
60	-
6C	%
6D	_
6E	>
6F	?
7A	:
7E	=

* using the EBCDIC 1047 code page

Usage & Invocation

- Enable special characters with

```
SETROPTS PASSWORD(SPECIALCHARS)
```

  - Confirm this with SETROPTS LIST

```
PASSWORD PROCESSING OPTIONS:
  THE ACTIVE PASSWORD ENCRYPTION ALGORITHM IS KDFAES
  PASSWORD CHANGE INTERVAL IS  60 DAYS.
  PASSWORD MINIMUM CHANGE INTERVAL IS   3 DAYS.
  MIXED CASE PASSWORD SUPPORT IS IN EFFECT
  SPECIAL CHARACTERS ARE ALLOWED.
  10 GENERATIONS OF PREVIOUS PASSWORDS BEING MAINTAINED.
  AFTER   5 CONSECUTIVE UNSUCCESSFUL PASSWORD ATTEMPTS,
      A USERID WILL BE REVOKED.
  PASSWORD EXPIRATION WARNING LEVEL IS  15 DAYS.
  INSTALLATION PASSWORD SYNTAX RULES:
    RULE 1  LENGTH(8)     xxxxxxxx
   LEGEND:
    A-ALPHA C-CONSONANT L-ALPHANUM N-NUMERIC V-VOWEL W-NOVOWEL *-ANYTHING
    c-MIXED CONSONANT m-MIXED NUMERIC v-MIXED VOWEL $-NATIONAL s-SPECIAL
    x-MIXED ALL
```

Usage & Invocation .. syntax rules

- New SPECIAL content-keyword for password syntax rules
  - Allows the existing national (@ # $) characters as well
- New MIXEDALL content-keyword for password syntax rules
  - Forces a mixture of character types based on the number of positions specifying the content-keyword
  - The categories are: upper case letters (which does **not** include the national characters), lower case letters (if MIXEDCASE is in effect), digits, and special characters (which includes the national characters)
  - Therefore, you can enforce that at least one of each category is specified with a single rule. Even without special characters active, the nationals can satisfy the special requirement.
- Existing NOVOWEL content-keyword allows special characters
- National characters behavior unchanged in existing content-keywords (e.g. NATIONAL, ALPHA, ALPHANUM, MIXEDNUM)

Usage & Invocation ...

- Use the new content-keywords

```
SETROPTS PASSWORD(RULE1(LENGTH(8) MIXEDALL(1:8)))
SETROPTS PASSWORD(RULE2(LENGTH(7:8) SPECIAL (2,5)))
```

- Confirm this with SETROPTS LIST

```
PASSWORD PROCESSING OPTIONS:
  ...
  INSTALLATION PASSWORD SYNTAX RULES:
    RULE 1  LENGTH(8)      xxxxxxxx
    RULE 2  LENGTH(7:8)    *s**s***
  LEGEND:
   A-ALPHA C-CONSONANT L-ALPHANUM N-NUMERIC V-VOWEL W-NOVOWEL *-ANYTHING
   c-MIXED CONSONANT m-MIXED NUMERIC v-MIXED VOWEL $-NATIONAL s-SPECIAL
   x-MIXED ALL
```

Usage & Invocation ...

- RACF ISPF panel updates not included in the APAR (for any of the functions described herein)
  - Same for TSO helps

- These will appear in a future release

Usage & Invocation … RRSF

- The RACF Remote Sharing Facility (RRSF) exchanges some information during the handshaking process as a connection is being established between two systems
- It issues a warning message when certain settings are out of synch, but allows the connection to proceed
- Such a warning will be issued when SPECIALCHARS is different

```
IRRI006I (<) ATTENTION: LOCAL NODE NODE2 HAS SETROPTS
       OPTION PASSWORD(SPECIALCHARS).  PARTNER NODE
       NODE1 HAS SETROPTS OPTION PASSWORD(NOSPECIALCHARS).
```

Usage & Invocation … RRSF

- When a password is propagated by automatic command direction (SET AUTODIRECT, with the ADDUSER, ALTUSER and PASSWORD commands), it is subject to command rules at target
  - If the password contains special characters, it will fail on the target system and the user's password will be out of sync

- When a password is propagated by automatic password direction (SET AUTOPWD) or password sync (SET PWSYNC) , it will be successful
  - The user **will** be able to TSO LOGON at target
  - The user will **not** be able to change the password using the PASSWORD command if the APAR is not applied, but **will** be able to if it is, even if SPECIALCHARS is not enabled.
  - The user **will** be able to change the password during LOGON
  - If password rules on an uplevel system require a special character, passwords propagated from downlevel systems will fail

Usage & Invocation … RRSF

- The RACLINK DEFINE command allows specification of target user's password for implicit approval of the user ID association
  - RACLINK ID(*thisuser*) DEFINE(*thatnode*.*thatuser/thatpwd*) PEER(PWSYNC)
- If the password starts with "*", TSO treats "/*" as a comment and ignores the rest of the command!
- You might not even notice it didn't work!
  - E.G. if issued by a SPECIAL user, it will be authorized for that reason. PEER(PWSYNC) is the default.
- Solution: single quotes are now allowed around the entire string in all cases, but are only required when the password starts with "*"

```
RACLINK DEFINE('NODE2.USER2/*pwd-TSO!') PEER(PWSYNC)
```

Usage & Invocation … RACF downloads on the RACF web site

- IRRXUTIL – Various sample REXX execs using IRRXUTIL
  - XSETRPWD reports on password-related SETROPTS options. This is updated to report on special character enablement.

- REXXPWEXIT – sample REXX-based new password exit
  - Being updated to add special characters to list of allowable characters

```
special = '$@#.<+|&!*-%_>?:='
```

  - If you are currently using this exit, you will need to update it
    - IRRPWREX in your System REXX concatenation (perhaps SYS1.SAXREXEC)
  - Also being updated with a number of additional checks that have been requested, and with a list function that can be invoked from the console

```
F AXR,IRRPWREX LIST
```

Overview - phrase-only

- Problem Statement / Need Addressed
  - Password phrases are effective against offline attacks because their possible lengths make brute-force attacks computationally infeasible
  - While RACF allows the use of phrases, it also requires a password. As long as that password can be hacked, the user can be compromised.
- Solution
  - Allow users to have a phrase without also requiring a password
- Benefit / Value
  - Defense in depth against offline attacks (You still need to carefully protect copies of the RACF database!)
  - Consistency with RACF on z/VM

Overview – Summary of current behavior wrt to NOPASSWORD

Command	Result
ADDUSER user PHRASE() NOPASSWORD	ICH01002I NOPASSWORD OPERAND IGNORED
ALTUSER user PHRASE() NOPASSWORD	ICH21041I NOPASSWORD OPERAND IGNORED
ALTUSER user NOPASSWORD when the user currently has a password and a phrase	ICH21041I NOPASSWORD OPERAND IGNORED
ALTUSER user PHRASE() when the user currently has no password (i.e. is PROTECTED)	ICH21040I PHRASE OPERAND IGNORED

Usage & Invocation

- Uses existing NOPASSWORD keyword of ADDUSER/ALTUSER
- It is now simply allowed in the cases shown on the previous slide
- Displayed by LISTUSER using existing attributes

```
USER=PHRONLY   NAME=UNKNOWN   OWNER=IBMUSER    CREATED=14.206
 DEFAULT-GROUP=SYS1      PASSDATE=N/A     PASS-INTERVAL=  30 PHRASEDATE=00.000
 ATTRIBUTES=NOPASSWORD  PASSPHRASE
```

- Authority required: same required to add or remove a phrase:
  - system special, group special or owner
    - “helpdesk authority” does not apply

Overview – expire a password/phrase

- Problem Statement / Need Addressed
  - Customers sometimes have the need to force their users to change their passwords ASAP, for example, after implementing a new password quality rule. Customers must now either write their own ICHEINTY program or play games with the password interval.
- Solution
  - Allow expiration of a password with ALTUSER, without needing to change the password
- Benefit / Value
  - Quicker adoption of new password rules in a simple fashion
  - Ability to evenly distribute password change intervals in case password encryption performance spikes are encountered

Usage & Invocation

- Uses existing EXPIRED keyword of ALTUSER
- When specified without PASSWORD and PHRASE, sets the changed-dates to 0, thus expiring the password
  - Previously, EXPIRED was ignored in this situation

- Before:

```
USER=GRONK   NAME=UNKNOWN   OWNER=IBMUSER    CREATED=14.206
 DEFAULT-GROUP=SYS1      PASSDATE=14.206  PASS-INTERVAL=  30 PHRASEDATE=14.206
 ATTRIBUTES=PASSPHRASE
```

- ALTUSER GRONK EXPIRED
- After:

```
USER=GRONK   NAME=UNKNOWN   OWNER=IBMUSER    CREATED=14.206
 DDEFAULT-GROUP=SYS1      PASSDATE=00.000  PASS-INTERVAL=  30 PHRASEDATE=00.000
 ATTRIBUTES=PASSPHRASE
```

Usage & Invocation ...

- To do it in bulk:

```
SEARCH CLASS(USER) CLIST('ALTUSER ' ' EXPIRED') NOLIST
EX 'prefix.EXEC.RACF.CLIST'
```

Overview – password history cleanup

- Problem Statement / Need Addressed
  - When SETROPTS PASSWORD(HISTORY(*n*)) is lowered, some history values are stranded and are checked forever. The CUTPWHIS download on the RACF web site can be used to remove it.
- Solution
  - Provide a PWCLEAN keyword of the ALTUSER function to provide this function
- Benefit / Value
  - Remove useless history values that could be expensive to evaluate when in KDFAES format.
  - Eliminate need to download, assemble, link-edit the (authorized) sample program

Usage & Invocation

- For an individual user

```
ALTUSER userid PWCLEAN
```

- Or, in bulk:

```
SETROPTS PASSWORD(HISTORY(new-value))
SEARCH CLASS(USER) CLIST('ALTUSER ' ' PWCLEAN') NOLIST
EX 'prefix.EXEC.RACF.CLIST'
```

- We recommend doing this **any** time the history value is changed, not just when it is lowered
  - When it is raised, there are circumstances, on a per-user basis, when the new value may not immediately take effect. And it's impossible to know from the outside looking in.
- p.s. PWCLEAN will also delete history from PROTECTED users

Interactions & Dependencies with other software products

- A fix category is an identifier used to group and associate PTFs to a particular category of software fixes. The following fix categories identify the group of fixes that are required to support these password enhancements.

Category	Description	Keyword
IBM.Function.RACF.Password Characters	Fixes for z/OS Security Server RACF to support additional special characters in passwords, and fixes for other z/OS software to support this enhancement.	RACFPWCHAR/K
IBM.Function.RACF.Password Encryption	Fixes for z/OS Security Server RACF to support a stronger password encryption algorithm, and fixes for other z/OS software to support this enhancement.	RACFPWENCR/K

- Informational APAR II14765 documents known restrictions, and will be maintained over time

Migration & Coexistence Considerations - KDFAES

- Do **not** enable KDFAES if you are sharing the RACF database, until the service has been applied on all sharing systems

- Note there are no RACF remote sharing consideration. The active algorithm on the target node determines password format on the target.

Migration & Coexistence Considerations – Special characters

- Do **not** use special characters if you are sharing the RACF database, until the service has been applied on all sharing systems
- Also, the RRSF considerations mentioned earlier

Session Summary

- RACF is providing a number of functions to enhance different facets of password security on z/OS
- Many functions satisfy long-standing requirements
- A number of other software components and products are affected by this change and we are trying to make it as easy as possible for you to identify them

Affected publications

- RACF: Security Administrator's Guide (SA23-2289)
- RACF: System Programmer's Guide (SA23-2287)
- RACF: Command Language Reference (SA23-2292)
- RACF: Callable Services (SA23-2293)
- RACF: Macros and Interfaces (SA23-2288)
- RACF: Messages and Codes (SA23-2291)
- RACF: Data Areas (GA32-0885)
- RACF: Diagnosis Guide (GA32-0886)
- RACF: RACROUTE Macro Reference (SA23-2294)