z/OS

APARs OA34258 and OA34259: Enhanced RACF support for z/OS identity propagation

Preface

This information applies to APARs OA34258 and OA34259 for enhanced RACF support for z/OS identity propagation.

Overview

APARs OA34258 (RACF) and OA34259 (SAF) enhance RACF support for z/OS identity propagation by adding a new query function for distributed identity filters that can be used to find the matching RACF user ID associated with a particular filter. The new query function is available with the RACMAP command for use by the RACF administrator and with the R_usermap (IRRSIM00) callable service for programmatic use. Additionally, enhancements to the R_cacheserv (IRRSCH00) callable service provide support for subsystem callers, such as CICS, to create reusable ICRX objects and to validate user-built ICRX objects.

Software requirements

Support for APAR OA34258 and APAR OA34259 requires one of the following software releases:

- z/OS Security Server RACF Version 1 Release 11 (FMID HRF7760)
- z/OS Security Server RACF Version 1 Release 12 (FMID HRF7770)

Updated RACF publications

The chapters of this document supplement the V1R11 and V1R12 levels of the following RACF publications:

Chapter	Supplements ...
Chapter 1, “Security administration updates,” on page 1	*z/OS Security Server RACF Security Administrator’s Guide*
Chapter 2, “Command updates,” on page 3	*z/OS Security Server RACF Command Language Reference*
Chapter 3, “Messages updates,” on page 9	*z/OS Security Server RACF Messages and Codes*
Chapter 4, “Macros and interface updates,” on page 11	*z/OS Security Server RACF Macros and Interfaces*
Chapter 5, “Callable services updates,” on page 13	*z/OS Security Server RACF Callable Services Reference*
Chapter 6, “RACROUTE macro updates,” on page 21	*z/OS Security Server RACROUTE Macro Reference*
Chapter 7, “Data area updates,” on page 23	*z/OS Security Server RACF Data Areas*

© Copyright IBM Corp. 2011

Chapter 1. Security administration updates

This information supplements *z/OS Security Server RACF Security Administrator's Guide*.

The following information is updated in the topic called "Profiles in the IDIDMAP class" in "Chapter 24. Distributed identity filters".

> The name of an IDIDMAP profile is the user name portion of the filter, specified as the USERDIDFILTER value, stripped of any leading or trailing blank or null characters, normalized according to the rules described in "Updates to the MAP function of the RACMAP command" on page 3, and encoded as UTF-8 data.

© Copyright IBM Corp. 2011

Chapter 2. Command updates

This information supplements *z/OS Security Server RACF Command Language Reference*.

This chapter contains the following topics:

- "Update to the LISTMAP function of the RACMAP command"
- "Updates to the MAP function of the RACMAP command"
- "Description of the new QUERY function of the RACMAP command"

Update to the LISTMAP function of the RACMAP command

The following note is added to the description for the LABEL parameter of the LISTMAP function:

Note: When you define a distributed-identity user name as an X.500 distinguished name (DN), the DN appears in its normalized form in the LISTMAP output. For details about how a DN is normalized, see the description of the USERDIDFILTER operand of the MAP function.

Updates to the MAP function of the RACMAP command

The following information is updated in the parameter descriptions for the following operands of the MAP function:

USERDIDFILTER

The following updates are made to the description of the USERDIDFILTER operand:

- The USERDIDFILTER operand is required for the MAP and QUERY functions and ignored for other RACMAP functions.
- Any leading or trailing blank or null characters are removed from the user name value before it is stored in the IDIDMAP profile.
- Do not escape the equal sign (=), semicolon (;), or comma (,) when you specify them as delimiters of an RDN.
- Do not specify a blank character immediately preceding or following the equal sign (=) when using the equal sign as a delimiter of an attribute type or an RDN.

The following information is added to the description of the USERDIDFILTER operand:

Normalization of the X.500 distinguished name (DN): When you specify the user name as a DN, the name is normalized before it is stored in the IDIDMAP profile. The normalized form of the DN appears in the output of the RACMAP LISTMAP command.

Normalization of the DN is done as follows:

- Any leading blank or null characters at the beginning of each RDN are removed.
- Any trailing blank or null characters at the end of each RDN are removed with the following exception.

© Copyright IBM Corp. 2011

Exception: The last escaped blank or null character that precedes an RDN delimiter (an unescaped semicolon or comma) is not removed unless it appears in the last RDN.

– Any unescaped semicolon delimiter is replaced by a comma.

– Any lowercase characters that appear in the attribute type of each RDN are translated to uppercase characters.

Note: During normalization, a character is processed as an escaped character when it is preceded by an odd number of consecutive backslash characters.

REGISTRY

The following updates are made to the description of the REGISTRY operand:

- The REGISTRY operand is required for the MAP and QUERY functions and ignored for other RACMAP functions.
- Any leading or trailing blank or null characters are removed from the registry name value before it is stored in the IDIDMAP profile.

Description of the new QUERY function of the RACMAP command

The following information about the new QUERY function is added to the authorization, syntax, and parameter information for the RACMAP command.

Authorization required

To use the QUERY function of the RACMAP command, you must have SPECIAL authority or READ access to the IRR.IDIDMAP.QUERY resource in the FACILITY class.

Syntax

Syntax information about the QUERY function of the RACMAP command is added as follows:

```
RACMAP
        QUERY
          USERDIDFILTER(NAME('distributed-identity-user-name'))
          REGISTRY(NAME('distributed-identity-registry-name'))
```

Parameters

Information describing the QUERY function of the RACMAP command is added as follows:

QUERY

Specifies the QUERY function of the RACMAP command. Use the QUERY function to find the matching RACF user ID that is associated with a distributed identity filter.

Rule: When you specify the QUERY function, you must specify USERDIDFILTER and REGISTRY.

USERDIDFILTER(NAME(*'distributed-identity-user-name'***))**

Specifies the significant portion of the distributed-identity user name. RACF® uses the user name as part of the distributed identity filter to map a distributed identity to a RACF user ID.

The USERDIDFILTER operand is required for the MAP and QUERY functions and ignored for other RACMAP functions.

Specify the user name value enclosed in single quotation marks. If a single quotation mark is intended to be part of the name, specify two single quotation marks together for each single quotation mark in the name, and enclose the entire name value in single quotation marks.

The maximum length for a user name is 246 bytes.

In general, the user name can contain blank and mixed-case characters. Any leading or trailing blank or null characters are removed from the value before it is stored in the IDIDMAP profile.

You cannot specify the name value as a hexadecimal character string.

Examples:

```
USERDIDFILTER(NAME('DENICE'))
USERDIDFILTER(NAME('UID=GUSKI,OU=Tools,O=IBM,C=US'))
USERDIDFILTER(NAME('Rich''s ID'))
USERDIDFILTER(NAME('Dev\+Test219'))
```

Restriction for names containing multibyte characters: Because RACF converts the name value you specify from EBCDIC to UTF-8 format prior to storing it in the RACF database, if your value contains multibyte characters, the resulting UTF-8 value might be longer than 246 bytes. If this occurs, the command fails and message IRRW213I is issued.

Format of the user name value: Specify the user name value in either of the following two formats:

1. As a simple character string, such as a user ID defined in a non-LDAP registry.

   Typically, special characters do not appear in user names stored within a registry. However, if you need to specify a user name value that includes certain characters, they must be preceded by the backslash (\) escape character.

   These characters include the plus sign (+), semicolon (;), comma (,), quotation mark ("), backslash (\), less than symbol (<), greater than symbol (>) and the equal sign (=).

2. As a character string that represents an X.500 distinguished name (DN).

   A DN consists of one or more relative distinguished names (RDNs). Each RDN consists of an attribute type and attribute value, separated by an equal sign (=). RDNs are separated by a comma (,).

   When you use mixed-case characters to specify the user name as a DN, the RACMAP command translates the attribute types to uppercase characters, and preserves the mixed-case characters of the attribute value.

   The RACMAP command performs no validity checking of the X.500 names you specify.

   Rules for specifying the user name as a distinguished name (DN):

   - Specify the user name value in its canonical form, as it is defined within the registry, with any special characters preceded by the backslash (\) escape character. You must specify the RDNs in their correct sequence.

For example, for users of WebSphere® Application Server applications, the canonical form of the user name must match the value returned by the `WSCredential` interface method called `getUniqueSecurityName()`.

- Typically, special characters do not appear in user names stored within a registry. However, if you need to specify a user name value that includes certain characters, including LDAP special characters, they must be preceded by the backslash (\) escape character.

  These characters include the plus sign (+), semicolon (**;**), comma (**,**), quotation mark ("), backslash (\), less than symbol (<), greater than symbol (>), and the equal sign (=).

  Exception: Do not escape the equal sign (=), semicolon (**;**), or comma (**,**) when you specify them as delimiters of an RDN.
- Do not specify a blank character immediately preceding or following the equal sign (=) when using the equal sign as a delimiter of an attribute type or an RDN.

Normalization of the X.500 distinguished name (DN): When you specify the user name as a DN, the name is normalized before it is used to find the matching user ID that is associated with the distributed identity filter. For details about how the DN is normalized, see the description of the USERDIDFILTER operand of the MAP function.

REGISTRY(NAME(*'distributed-identity-registry-name'* | *'*'***))**

Specifies the registry that contains the distributed-identity user name. RACF uses the registry name as part of the distributed identity filter to map a distributed identity to a RACF user ID.

The REGISTRY operand is required for the MAP and QUERY functions and ignored for other RACMAP functions.

Specify the registry name value enclosed in single quotation marks. If a single quotation mark is intended to be part of the name, specify two single quotation marks together for each single quotation mark in the name, and enclose the entire name value in single quotation marks.

The maximum length for a registry name is 255 bytes.

Examples:

```
REGISTRY(NAME('ldaps://us.richradioham.com'))
REGISTRY(NAME('ldap://12.34.56.78:389'))
```

The registry name can contain blank and mixed-case characters. Any leading or trailing blank or null characters are removed from the value before it is stored in the IDIDMAP profile.

You cannot specify the name value as a hexadecimal character string.

Restriction for names containing multibyte characters: Because RACF converts the name value you specify from EBCDIC to UTF-8 format prior to storing it in the RACF database, if your value contains multibyte characters, the resulting UTF-8 value might be longer than 255 bytes. If this occurs, the command fails and message IRRW213I is issued.

Defining registry names for LDAP servers: When the user's distributed identity is based on an LDAP registry, specify the *distributed-identity-registry-name* value as the URL of the LDAP server where the user is defined. The URL is defined with a `listen` option in the `ds.conf`

configuration file of the LDAP server, or overridden using the **-l** command-line parameter when the LDAP server is started.

For information about LDAP URLs, see *IBM Tivoli Directory Server Administration and Use for z/OS*.

For users of WebSphere Application Server applications: The registry name must match the value returned by the `WSCredential` interface method called `getRealmName()`.

The RACMAP command performs no validity checking of the registry names you specify.

Chapter 3. Messages updates

This information supplements *z/OS Security Server RACF Messages and Codes*.

Updated information about the RACMAP command messages

IRRW214I **The** *KeyWord-Name* **keyword is ignored when specified with the** *Function-Name* **function.**

Explanation: You specified a keyword that is not needed by the function.

System action: Command processing continues.

User response: Do not do anything now, however, the next time you issue the command you can avoid specifying this keyword.

IRRW215I **No user ID found associated with the specified USERDIDFILTER and REGISTRY name.**

Explanation: The information you provided with the USERDIDFILTER and REGISTRY names is not associated with any RACF user ID.

System action: Command processing ends.

User response: If you want to define a distributed identity filter that is associated with this USERDIDFILTER and REGISTRY name, use the RACMAP MAP function.

IRRW216I **Unexpected** *Callable-Service-Name* **callable service error encountered during command processing. SAF RC = x'***RetCode***', RACF RC = x'***RetCode***', RACF RSN = x'***RsnCode***'.**

Explanation: During command processing, RACMAP issued a call to this callable service and received a return code and reason code that were not expected.

System action: Command processing ends.

User response: Report this message to the system programmer, and provide the exact text of the command you issued.

System programmer response: Use the return code information in *z/OS Security Server RACF Callable Services* to determine the error condition and fix the error. If necessary, report the problem to the IBM support center.

© Copyright IBM Corp. 2011

Chapter 4. Macros and interface updates

This information supplements *z/OS Security Server Macros and Interfaces*.

In “Chapter 5. SMF records”, the topic called “Table of data type 6 command-related data” is updated to include information about a new flag added for the QUERY keyword of the RACMAP command. The new flag is located in bit 2 of byte 0 for SMF type 80 event code 87(57).

© Copyright IBM Corp. 2011

Chapter 5. Callable services updates

This information supplements *z/OS Security Server RACF Callable Services Reference*.

Updated information about R_cacheserv (IRRSCH00): Cache services

R_cacheserv (IRRSCH00): Cache services was updated as follows:

Function_codeX'0007'-Manage an extended read/write cache parameter was updated as follows:

If the **Option** parameter is X'0004' (**store** and return **reusable** ICRX), then processing proceeds as with Option X'0001', but the returned ICRX will be marked as being reusable. This ICRX will be valid for multiple Option X'0002' **RetrieveAppl** calls until it times out after an hour of inactivity.

If the **Option** parameter is X'0005' (validate the input ICRX), we will validate the input ICRX and the IDID included in it. This function is intended to validate the user-built ICRX that is provided by the application to a subsystem like CICS, not the completed ICRX that is returned by RACF to the caller of R_cacheserv as is done with Option X'0001', which contains the RACF user-id in ICRXUSER and the ICR. The caller must be in supervisor state or system key.

The following fields will be validated:

- **ICRX** fields:
  - **ICRXID** – The value must be the literal 'ICRX'
  - **ICRXVERS** – The value must be greater than or equal to **ICRXVR01** and less than or equal to the current version – **ICRXCURV** (currently set to 2)
  - **ICRXOFFN** – The number of offsets must be 3 (for versions 1 and 2 of the ICRX)
  - **ICRXFLGS** – The value of this flag-byte must be 0
  - **ICRXLEN** – The value must be greater than or equal to the length of the ICRX header – **ICRXHLN**, and less than or equal to the length of the ICRX provided in the R_cacheserv parameter list
  - **ICRXICRO** – Must be 0
  - **ICRXDIDO** – Must be equal to the length of the ICRX header – **ICRXHLN**
  - **ICRXUSRO** – Must be 0
- **IDID** fields:
  - **IDIDID** – The value must be the literal 'IDID'
  - **IDIDVERS** – The value must be greater than or equal to **IDIDVR01** and less than or equal to the current version – **IDIDCURV** (currently set to 1)
  - **IDIDOFFN** – The number of offsets must be 5 (for version 1 of the IDID)
  - **IDIDHSHN** – The three high order bits of this bit string must be off (0), and for any IDID section not specified (offset is 0), the corresponding bit must be off (0)
  - **IDIDSP** – The value must be the same as the value in **ICRXSP**

© Copyright IBM Corp. 2011

- **IDIDLEN** – The value must be greater than the length of the IDID header – **IDIDHLN**, and the sum **ICRXDIDO** + **IDIDLEN** must be less than or equal to the length of the ICRX – **ICRXLEN**
- **IDIDOFF1** – Must be equal to the length of the IDID header – **IDIDHLN**
- **IDIDOFF2** – Must be 0
- **IDIDOFF3**, **IDIDOFF4**, and **IDIDOFF5** – Must be either 0, or (if non-zero) must be contained within the IDID and be in sequential order.

- **IDID Section1** fields:
  - **IDID1OF1** – Must be equal to the length of the IDID Section1 header
  - **IDID1UDL** – Must be greater than or equal to 1 and less than or equal to the value indicated by **RCVTDNL**
  - **IDID1OF2** – Must be equal to the sum of the length of the IDID Section1 header, the length of **IDID1UDL**, and the value specified in **IDID1UDL**
  - **IDID1RL** – Must be greater than equal to 1 and less than or equal to the value indicated by **RCVTRL**

Return and reason codes were updated as follows:

SAF return code	RACF return code	RACF reason code	Explanation
8	12	11	An invalid option was specified. Valid options for function code X'0006' are X'0001' through X'0007'. Valid options for function code X'0007' are X'0001' through X'0005'.
8	100	Offset to the ICRX field in error.	This return/reason code is issued when R_cacheserv is invoked to validate an ICRX. It indicates non-valid data in the ICRX portion of ICRX. The offsets are calculated from the beginning of the ICRX.
8	104	Offset to the IDID field in error.	This return/reason code is issued when R_cacheserv is invoked to validate an ICRX. It indicates non-valid data in the IDID portion of ICRX. The offsets are calculated from the beginning of the IDID.
8	108	Offset to the IDID Section 1 field in error.	This return/reason code is issued when R_cacheserv is invoked to validate an ICRX. It indicates non-valid data in Section 1 of the IDID portion of ICRX. The offsets are calculated from the beginning of Section 1 of the IDID.
8	112	Offset to the User's Distinguished Name field in error.	This return/reason code is issued when R_cacheserv is invoked to validate an ICRX. It indicates non-valid data in the User's Distinguished Name Data Section in Section 1 of the IDID portion of ICRX. The offsets are calculated from the beginning of the User's Distinguished Name Data Section.

SAF return code	RACF return code	RACF reason code	Explanation
8	116	Offset to the Registry Name field in error.	This return/reason code is issued when R_cacheserv is invoked to validate an ICRX. It indicates non-valid data in the Registry Name Data Section in Section 1 of the IDID portion of ICRX. The offsets are calculated from the beginning of the Registry Name Data Section.

Parameter usage was updated as follows:

Function	**Manage an extended read/write cache**				
Function_code	X'0007'				
Option	X'0001' **Store**	X'0002' **RetrieveAppl**	X'0003' **Remove**	X'0004' **Store** a reusable ICRX	X'0005' **Validate** a user-built ICRX
ParmALET	In	In	In	In	In
NumParms	In	In	In	In	In
Version	N/A	N/A	N/A	N/A	N/A
Version_length	N/A	N/A	N/A	N/A	N/A
Cache_name	N/A	N/A	N/A	N/A	N/A
Record_name_ptr	N/A	N/A	N/A	N/A	N/A
Record_name_length	N/A	N/A	N/A	N/A	N/A
Data_ptr	N/A	Out	N/A	N/A	N/A
Data_length	N/A	Out	N/A	N/A	N/A
Data_timeout	N/A	N/A	N/A	N/A	N/A
Source_ptr	N/A	N/A	N/A	N/A	N/A
Source_length	N/A	N/A	N/A	N/A	N/A
Reference_timeout	N/A	N/A	N/A	N/A	N/A
Reference_userID	N/A	N/A	N/A	N/A	N/A
Reference	N/A	N/A	N/A	N/A	N/A
Subpool	In	In	N/A	In	N/A
ACEE_ALET	In*	N/A	N/A	In*	N/A
ACEE	In*	N/A	N/A	In*	N/A
ICRX_area	Out	In	In	Out	In
ICRX_length	Out	In	In	Out	In
* This parameter is optional, see parameter description.					

Updated information about R_usermap (IRRSIM00): Map application user

R_usermap is updated as follows.

Function

The **R_usermap** service enables z/OS application servers to determine the application user identity associated with a RACF user ID, or to determine the RACF user ID associated with an application user identity or digital certificate, except for Identity Propagation in which case a user's Distinguished Name and a Registry/Realm Name will be used to determine the associated RACF user ID, but not the reverse.

For Identity Propagation, the distributed identity (user's Distinguished Name) must be associated with a RACF user ID. Use the RACMAP command to create the association between the distributed identity and a RACF defined user ID (this association is also known as a ‘filter’).

RACF authorization

Function codes X'0001' through X'0006' only: The use of the **R_usermap** service is authorized by the resource **IRR.RUSERMAP** in the **FACILITY** class for servers not running in system key or supervisor state. The application server must be running with a RACF user or group that has at least **READ** authority to this resource. Only servers running in system key or supervisor state may use the **R_usermap** service if the class is inactive or the resource is not defined.

Function codes X'0008' only: The use of the **R_usermap** service is authorized by the resource **IRR.IDIDMAP.QUERY** in the **FACILITY** class for servers not running in system key or supervisor state. The application server must be running with a RACF user or group that has at least **READ** authority to this resource. Only servers running in system key or supervisor state may use the **R_usermap** service if the class is inactive or the resource is not defined.

Parameters

Function_code
The name of a halfword containing the function code. The function code has one of the following values:

X'0008'
Return the RACF user ID associated with the supplied user's Distinguished Name and Registry/Realm Name.

Distinguished_Name
The name of an area that consists of a 2-byte length field followed by the distinguished name (distributed user ID), in UTF-8 format, of up to the maximum length allowed by the RCVT field RCVTDNL (currently 246). If not specified, the length must equal 0. For a non-zero length, the field cannot be all blanks (x'20'), all nulls (x'00'), or a combination of blanks and nulls.

Notes:

1. The following operations are performed on a copy of the data. The original data is not modified.
   - All leading and trailing blanks (x'20'), nulls (x'00'), or combination of blanks and null characters will be removed from the string and the length will be appropriately adjusted.
   - If the distributed-identity-user-name (user name) is in X.500 format the name will be normalized before it is used to find the matching RACF user ID that is associated with the distributed identity filter.
2. The normalization rules are described in detail under RACMAP MAP.

Registry_Name
The name of an area that consists of a 2-byte length field followed by the registry/realm name, in UTF-8 format, of up to the maximum length allowed by the RCVT field RCVTRL (currently 255). If not specified, the length must equal 0. For a non-zero length, the field cannot be all blanks (x'20'), all nulls (x'00'), or a combination of blanks and nulls.

Note: All leading and trailing blanks (x'20'), nulls (x'00'), or combination of blanks and null characters will be removed from the string and the

length will be appropriately adjusted. This operation is performed on a copy of the data. The original data are not modified.

Return and reason codes

R_usermap may return the following values in the reason and return code parameters:

SAF return code	RACF return code	RACF reason code	Explanation
8	8	36	High order bit was not set to indicate last parameter.
8	8	40	The Distinguished Name length is not valid, or the Distinguished Name string is all blanks (x'20'), all nulls (x'00'), or a combination of blanks and nulls.
8	8	44	The Registry Name length is not valid, or the Registry Name string is all blanks (x'20'), all nulls (x'00'), or a combination of blanks and nulls.
8	8	48	There is no distributed identity filter mapping the supplied distributed identity to a RACF user ID, or The IDIDMAP RACF general resource class is not active or not RACLISTed

Parameter usage

Table 1. Parameter usage

Parameter	Function code 1 (RACF to Notes)	Function code 2 (Notes to RACF)	Function code 3 (RACF to NDS)	Function code 4 (NDS to RACF)	Function code 5 (RACF to KERB)	Function code 6 (KERB to RACF)	Function code 8 (DID to RACF)
SAF_return_code	Output	Output	Output	Output	Output	Output	Output
RACF_return_code	Output	Output	Output	Output	Output	Output	Output
RACF_reason_code	Output	Output	Output	Output	Output	Output	Output
Function_code	Input	Input	Input	Input	Input	Input	Input
Option_word	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
RACF_userid	Input	Output	Input	Output	Input	Output	Output
Certificate	N/A	Input	N/A	Input	N/A	Input	N/A
Application_userid	Output	Input	Output	Input	Output	Input	N/A
Distinguished_Name	N/A	N/A	N/A	N/A	N/A	N/A	Input
Registry_Name	N/A	N/A	N/A	N/A	N/A	N/A	Input

Usage notes

The following usage notes have been either added or updated.

- The parameter list for this callable service is intended to be variable length to allow for future expansion. To allow for this, the last word in the parameter list must have a 1 in the high-order bit. If the last word in the parameter list does not have a 1 in the high-order (sign) bit, the caller receives a parameter list error.

- For function codes 1-6, the first parameter that can have the high-order bit on, ending the parameter list, is the Application_userid parameter.
- For function code 8, the first parameter that can have the high-order bit on, ending the parameter list, is the Registry_Name parameter.

- For function codes 1-6:
  - The caller receives a parameter list error if the Function_code indicates that a RACF user ID is to be returned, and no Application_userid or Certificate is supplied.
  - Specification of a RACF_userid with a length greater than 8 or an Application_userid with a length greater than 246 will result in a parameter list error.
  - If the Function_code specifies that a RACF user ID is to be returned and the length supplied for the Application_userid is greater than the maximum allowed, such as greater than 64 for a Lotus Notes® for z/OS user identity, or greater than 240 for Security Server Network Authentication Service user identity, the caller receives the "no mapping between RACF and an application" error.
  - If the Function_code indicates that a RACF user ID is to be returned, and both an Application_userid and a Certificate are supplied, the Application_userid will be used.
- For function code 8:
  - The length of the Distinguished_Name must be greater than 0 and less than or equal to the maximum length allowed by the RCVT field RCVTDNL (currently 246). A length of 0 or greater than the maximum allowed will result in the, Distinguished Name length not valid, error (**SAF Return Code = 8, RACF Return Code = 8, RACF Reason Code = 40**).
  - The length of the Registry_Name must be greater than 0 and less than or equal to the maximum length allowed by the RCVT field RCVTRL (currently 255). A length of 0 or greater than the maximum allowed will result in the, Registry Name length not valid, error (**SAF Return Code = 8, RACF Return Code = 8, RACF Reason Code = 44**).
  - The Distinguished_Name and Registry_Name must be in UTF-8 format. If they are not in UTF-8 format the associated RACF user ID will not be found (**SAF Return Code = 8, RACF Return Code = 8, RACF Reason Code = 48**).
- Specification of an unknown function code will result in a parameter list error.
- The Distinguished Name value can be in any of the following formats:
  1. As a simple character string, such as a user ID defined in a non-LDAP registry.

     Typically, special characters do not appear in user names stored within a registry. However, if you need to specify a user name value that includes certain characters, they must be preceded by the backslash (\) escape character.

     These characters include the plus sign (+), semicolon (;), comma (,), quotation mark ("), backslash (\), less than symbol (<), greater than symbol (>) and the equal sign (=).
  2. As a character string that represents an X.500 distinguished name (DN).

     A Distinguished Name (DN) consists of one or more relative distinguished names (RDNs). Each RDN consists of an attribute type and attribute value, separated by an equal sign (=). RDNs are separated by a comma (,).

     Like the RACMAP command, R_usermap does not perform validity checking of the X.500 name.

- The following rules are used to define a user name as Distinguished Name when the RACMAP command is used to define the mappings:
  - Specify the user name value in its canonical form, as it is defined within the registry, with any special characters preceded by the backslash (\) escape character. You must specify the RDNs in their correct sequence.

    For example, for users of WebSphere® Application Server applications, the canonical form of the user name must match the value returned by the WSCredential interface method called getUniqueSecurityName().
  - Typically, special characters do not appear in user names stored within a registry. However, if you need to specify a user name value that includes certain characters, including LDAP special characters, they must be preceded by the backslash (\) escape character.

    These characters include the plus sign (+), semicolon (;), comma (,), quotation mark ("), backslash (\), less than symbol (<), greater than symbol (>), and the equal sign (=).

    Exception: Do not escape the equal sign (=), semicolon (;), or comma (,) when you specify them as delimiters of an RDN.
  - Do not specify blank characters immediately preceding or following the equal sign (=) when using the equal sign as a delimiter of an attribute type or an RDN.

Chapter 6. RACROUTE macro updates

This information supplements *z/OS Security Server RACROUTE Macro Reference*.

Update to RACROUTE REQUEST=EXTRACT (standard form)

In the description of RACROUTE REQUEST=EXTRACT (standard form) DIDCT and DIDLIST1 have been added to the list of extractable fields under the description of the BRANCH keyword, as follows.

,BRANCH=YES
,BRANCH=NO
specifies whether you want RACF to use a branch entry.

The following applies to TYPE=EXTRACT with BRANCH=YES:

The RACROUTE REQUEST=EXTRACT macro supports an SRB-compatible branch entry when you specify BRANCH=YES and TYPE=ENCRYPT or BRANCH=YES and TYPE=EXTRACT with no change in function. with TYPE=EXTRACTN.

Cross memory mode is supported to obtain general resource profiles.

- General resource profiles that can be brought into storage are candidates for branch entry EXTRACT.
  - You can use the SETROPTS RACLIST command or RACROUTE REQUEST=LIST, GLOBAL=YES command to create a global listing of profiles in a data space. You can use this list only in the address space it was issued from.
  - You can also use RACROUTE REQUEST=LIST, GLOBAL=NO to create a listing of profiles in the user's address space, but this does not create a global listing of profiles.
- User data that is defaulted from the ACEE is a candidate for branch entry EXTRACT. This occurs when the USER class is specified or CLASS= is not specified, no ENTITY or ENTITYX is specified or ENTITYX is specified with zero for buffer length and zero for the actual entity name length, and no SEGMENT or FIELDS information is specified. The user's ID and default connect group are extracted from the current ACEE.

  If the user's primary and secondary languages are available, they are also extracted from the current ACEE, along with a code (U) indicating that the reported languages are defined in the user's profile. If the user's primary and secondary languages are not available in the user's profile, the installation default primary and secondary languages set by SETROPTS are returned, along with a code (S) indicating that the reported languages are the installation default.

  Additionally, if the user's work attributes (WORKATTR) information is available, it is also extracted from the ACEE. For the format of the WORKATTR information returned from the ACEE, see “RXTW: RACROUTE REQUEST=EXTRACT Result Area Mapping” in z/OS Security Server RACF Data Area.
- RACF can extract the following fields of the general-resource profile:

© Copyright IBM Corp. 2011

NOT programming interface information

CATEGORY, IPLOOK, MEMCNT, MEMLST, and NUMCTGY. **Exception:** The MEMCNT and MEMLST fields of the SECLABEL profile are programming interfaces.

End of NOT programming interface information

ACL2, ACL2ACC, ACL2CNT, ACL2NAME, ACL2RSVD, ACL2UID, ACLCNT, APPLDATA, AUDIT, CONVSEC, CSFAUSE, CSFSCPW, CSFSEXP, CSFSCLBS, CSFSCLCT, CSFSKLBS, CSFSKLCT, DIDCT, DIDLIST1, GAUDIT, INSTDATA, KEYDATE, KEYINTVL, LEVEL, LOGDAYS, LOGTIME, LOGZONE, NOTIFY, OWNER, SECLABEL, SECLEVEL, SESSKEY, SLSFLAGS, UACC, USERACS, USERID, and WARNING.

- RACF searches RACLISTed profiles in the following order:
  - Those off the ACEE (if ACEE is specified),
  - Those off the TCB ACEE in the PRIMARY address space,
  - Those off the ASXB ACEE in the PRIMARY address space.

  If the profile is not found off any ACEE, RACF searches globally RACLISTed profiles.

To specify the BRANCH keyword, you must specify Release=1.9 or later.

Chapter 7. Data area updates

This information supplements *z/OS Security Server RACF Data Areas*.

Updated information about IRRPCOMP (COMP: Common SAF/RACF Parameter List for z/OS UNIX System Services)

New constants, UMAP_ID_PROPAGATION_LEN and UMAP_TOTAL_LEN.

New function codes, UMAP_DID_TO_R, CACH_EXT_STORE_MULT, and CACH_EXT_ICRX_VAL.

Offsets			Len	Name (Dim)	Description
Dec	**Hex**	**Type**			
0	(0)	STRUCTURE	32	UMAP	
. . .					
24	(18)	STRUCTURE	8	UMAP_ID_PROPAGATION	
24	(18)	ADDRESS	4	UMAP_DISTINGUISHED_NAME@	Address of an input area that consists of a 2-byte length field followed by the distinguished name (distributed user ID), in UTF-8 format, of up to the maximum length allowed by the RCVT field RCVTDNL (currently 246). If not specified, the length must equal zero.
28	(1C)	ADDRESS	4	UMAP_REGISTRY_NAME@	The name of an area that consists of a 2-byte length field followed by the registry or realm name, in UTF-8 format, of up to the maximum length allowed by the RCVT field RCVTRL (currently 255). If not specified, the length must equal zero.
		1...		UMAP_IDPROP_LAST_PARM	Variable length parameter list (for function code 8).

Offsets			Len	Name (Dim)	Description
Dec	**Hex**	**Type**			
0	(0)	STRUCTURE	248	UMAP_DISTINGUISHED_NAME_DS	Distinguished name mapping
0	(0)	UNSIGNED	2	UMAP_DISTINGUISHED_NAME_LEN	Distinguished name length
2	(2)	CHARACTER	246	UMAP_DISTINGUISHED_NAME	Distinguished name string

Offsets			Len	Name (Dim)	Description
Dec	**Hex**	**Type**			
0	(0)	STRUCTURE	257	UMAP_REGISTRY_NAME_DS	Registry or realm name mapping
0	(0)	UNSIGNED	2	UMAP_REGISTRY_NAME_LEN	Registry or realm name length
THIS IS THE LAST PARAMETER FOR FUNCTION CODE 8					
2	(2)	CHARACTER	255	UMAP_REGISTRY_NAME	Registry or realm name string

© Copyright IBM Corp. 2011

UMAP function codes

Len	Type	Value	Name	Description
2	DECIMAL	8	UMAP_DID_TO_R	Return the RACF ID that is mapped by this combination of Distinguished Name and Registry/Realm Name

CACH option values for function code 7

Len	Type	Value	Name	Description
4	DECIMAL	4	CACH_EXT_STORE_MULT	Store data in the read/write cache and return multi-use ICRX
4	DECIMAL	5	CACH_EXT_ICRX_VAL	Validate an ICRX

Cross Reference

Name	Hex Offset	Hex Value	Level
UMAP_DISTINGUISHED_NAME	2		
UMAP_DISTINGUISHED_NAME@	18		
UMAP_DISTINGUISHED_NAME_DS	0		
UMAP_DISTINGUISHED_NAME_LEN	0		
UMAP_ID_PROPAGATION	18		
UMAP_IDPROP_LAST_PARM	1C	80	
UMAP_REGISTRY_NAME	2		
UMAP_REGISTRY_NAME@	1C		
UMAP_REGISTRY_NAME_DS	0		
UMAP_REGISTRY_NAME_LEN	0		

Updated information about IRRPICRX (ICRX: Extended identity context reference)

New flag, ICRXFLGS.

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
4	(4)	UNSIGNED	1	ICRXVERS	Version number
5	(5)	UNSIGNED	1	ICRXOFFN	Number of offsets - 3 for versions 1 and 2
6	(6)	BITSTRING	1	ICRXFLGS	Flag bits
		1...		ICRXMULT	Multi-use ICRX

Constants

Len	Type	Value	Name	Description
1	DECIMAL	2	ICRXVR02	Version 2 of extended identity context reference
1	DECIMAL	2	ICRXCURV	Current version of extended identity context of ID context reference

Cross Reference

Name	Hex Offset	Hex Value	Level
ICRXFLGS	6		2
ICRXMULT	6	80	3

Updated information about ICHPISP (ISP: RACF In-Storage Profile)

New fields, RPEDIDCT, RPEDIDLN, and RPEDIDOF.

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
0	(0)	STRUCTURE	90	RACRPE	RESOURCE PROFILE ELEMENT

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
84	(54)	UNSIGNED	2	RPEDIDCT	NUMBER OF DIDLIST1 ENTRIES
86	(56)	UNSIGNED	2	RPEDIDLN	LENGTH OF DISTRIBUTED IDENTITY INFORMATION (DIDLIST1)
88	(58)	UNSIGNED	2	RPEDIDOF	OFFSET TO DISTRIBUTED IDENTITY INFORMATION (DIDLIST1)
90	(5A)	CHARACTER		RPEEND	END OF FIXED PART OF ELEMENT

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
0	(0)	STRUCTURE	*	RPEDID1	DIDLIST1 REPEAT GROUP, PART 1
0	(0)	UNSIGNED	1	RPEDIDLL	LABEL LENGTH
1	(1)	CHARACTER	*	RPEDIDLB	LABEL

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
0	(0)	STRUCTURE	*	RPEDID2	DIDLIST1 REPEAT GROUP, PART 2
0	(0)	CHARACTER	8	RPEDIDUS	USER ID
8	(8)	UNSIGNED	1	RPEDIDRL	REGISTRY LENGTH
9	(9)	CHARACTER	*	RPEDIDRG	REGISTRY NAME

Cross Reference

Name	Hex Offset	Hex Value	Level
RPEDID1	0		1
RPEDID2	0		1

Name	Hex Offset	Hex Value	Level
RPEDIDCT	54		3
RPEDIDLB	1		2
RPEDIDLL	0		2
RPEDIDLN	56		3
RPEDIDOF	58		3
RPEDIDRL	8		2
RPEDIDRG	9		2
RPEDIDUS	0		2

Updated information about ICHPRCVT (RCVT: RACF Communication Vector Table)

New programming interface, RCVTIDPV.

Offsets			Len	Name (Dim)	Description
Dec	Hex	Type			
376	(178)	UNSIGNED	1	RCVTIDPV	A value of 1 indicates that Identity Propagation 2 services are available on the system.
377	(179)	CHARACTER	3	*	RESERVED

Cross Reference

Name	Hex Offset	Hex Value	Level
RCVTIDPV	178		1

Trademarks

IBM®, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Other company, product, and service names may be trademarks or service marks of others.

© Copyright IBM Corp. 2011