RACF Identity Token (IDT) Support:
V1.00 APARs: RACF OA55926, SAF OA55927

Summary of Changes		
Version	**Date**	**Nature of Change**
V1.00		Initial version: RACF APAR OA55926 & SAF APAR OA55927

1 Introduction

RACROUTE Support for Identity Tokens:
An Identity Token (IDT) is used to encode information about a user which can be trusted by the consumer of the token.

Support is added to RACROUTE authentication processing to generate and validate IDTs. An application can call RACROUTE authentication processing and request that an IDT be returned. An application can also call RACROUTE to authenticate a user with an IDT specified instead of another credential like a password.

RACROUTE supports Identity Tokens in the format of a JSON Web Token (JWT). For more information on JSON Web Tokens, refer to the following RFC:
https://tools.ietf.org/html/rfc7519

For more details on using RACROUTE to generate and validate IDTs, please refer to the RACROUTE Macro Reference publication section below.

IDT Configuration:
The security administrator can create profiles in the IDTDATA class to configure how certain fields in an IDT are generated and validated. The profiles can be used to control options such as which key is used to sign the IDT and its validity period.

Support for Identity Tokens must be enabled on the system by activating the IDTDATA class before they will be generated or validated by RACROUTE.

For more details on configuring IDTDATA class profiles, please refer to the Command Language Reference publication section below.

Restriction: The ISPF panels and TSO helps are not updated for the new command operands with OA55926 and OA55927.

2 Planning

When installing service like this, consider the following before making changes:

- Create a backup copy of your RACF database.
- Apply the RACF IDT APARs to all systems sharing the RACF database.

2.1 Create a backup copy of your RACF database

Creating a backup of the RACF database is recommended whenever significant changes are being made to RACF and the RACF database.

2.2 Apply the RACF IDT APARs to all systems that share the RACF database

Make sure that the service is applied on all sharing systems, and that all the ++HOLD documentation has been reviewed.

2.3 RACF exit considerations

The ICHRIX01 preprocessing and ICHRIX02 postprocessing exits can alter the behavior of RACROUTE REQ=VERIFY authentication processing. When the IDTDATA class is activated, RACROUTE REQ=VERIFY will begin processing the IDTA parameter for any application which has specified it.
The Identity Token provides a new way to authenticate a user with RACROUTE REQ=VERIFY. In many cases when an Identity Token is specified, the password and password phrase parameters will be ignored.
Before activating the IDTDATA class, the installation must ensure that any ICHRIX01 and ICHRIX02 exits are compatible with Identity Token processing. For example, if these exits inspect the password or password phrase parameters to make processing decisions, they must take into account the new RACROUTE IDTA parameter processing.

The ICHRIX02 return code 4 is not valid when the IDTA keyword is specified and the IDTDATA class is active. In this case RACROUTE REQ=VERIFY will fail with 8/70/6 return codes.

2.4 Performance considerations

None

3 Updated RACF publications

Chapters of the following RACF publications are affected by the new function:

Publication Name	Publication Number
z/OS Security Server RACF Command Language Reference	SA23-2292
z/OS Security Server RACROUTE Macro Reference	SA23-2294
z/OS Security Server RACF Callable Services	SA23-2293
z/OS Security Server RACF Macros and Interfaces	SA23-2288
z/OS Security Server RACF Data Areas	GA32-0885
z/OS Security Server RACF Messages and Codes	SA23-2291
z/OS Security Server RACF System Programmer's Guide	SA23-2287

In the following sections, highlighting is used to denote changed information in existing documentation. Sections, tables, messages, command keywords, etc. without highlighting contain new information.

3.1 z/OS Security Server RACF Command Language Reference

This document supplements information for the following RACF commands:

- RDEFINE
- RALTER
- RLIST

3.1.1 RDEFINE

The IDTPARMS segment contains information for the generation and validation of an Identity Token (IDT).

Syntax

[IDTPARMS(
[SIGTOKEN(*pkcs11-token-name*)]
[SIGSEQNUM(*pkcs11-sequence-number*)]
[SIGCAT(*pkcs11-category*)]
[SIGALG(HS256 | HS384 | HS512)]
[ANYAPPL(YES | NO)]
[IDTTIMEOUT(*timeout-minutes*)]
)]

IDTPARMS

IDTPARMS

Specifies information for the IDTDATA class profile being changed.

SIGTOKEN(*pkcs11-token-name*)

Specifies the name of an ICSF PKCS#11 token name for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The token name may consist of alphanumeric characters, national characters (@, #, $) and the period symbol. The token name is not case sensitive.

The minimum token name length is 1. The maximum token name length is 32. There is no default value.

SIGSEQNUM(*pkcs11-sequence-number*)

Specifies the ICSF PKCS#11 sequence number of the key for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The sequence number must be a hexadecimal number.

The minimum sequence number is 1. The maximum sequence number length is

8 hexadecimal digits. The default value is 1.

SIGCAT(*pkcs11-category*)

Specifies the ICSF PKCS#11 category of the key for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The category must be one of the following values:

T – Specifies a clear token object.

Y – Specifies a secure token object.

The default value is T.

SIGALG(HS256 | HS384 | HS512)

Specifies the signature algorithm for the generation of Identity Token (IDT) signatures associated with this profile. The default value is HS256.

HS256

Specifies the signature algorithm as HMAC with SHA-256.

HS384

Specifies the signature algorithm as HMAC with SHA-384.

HS512

Specifies the signature algorithm as HMAC with SHA-512.

ANYAPPL(YES | NO)

Specifies whether the IDT that RACROUTE generates can be used for any application name or only for the application name that performed authentication. The default value is YES.

When ANYAPPL(YES) is specified, RACROUTE will generate the IDT so it can be used for any application name.

When ANYAPPL(NO) is specified, RACROUTE will generate the IDT so that it can only be used by the application name that performed authentication.

When an IDT is generated by RACROUTE which is not for an end user by specifying the IDTA parameter field IDTA_End_User_IDT set to off, RACROUTE will ignore this setting and generate the IDT so that it can be used with any application name.

IDTTIMEOUT(*timeout-minutes*)

Specifies the number of minutes that the Identity Token (IDT) associated with the profile is active.

The value of timeout-minutes can be between 1 and 1440. The default value is 5.

3.1.2 RALTER

The IDTPARMS segment contains information for the generation and validation of an Identity Token (IDT).

Syntax

[IDTPARMS(
[SIGTOKEN(*pkcs11-token-name*) | NOSIGTOKEN]
[SIGSEQNUM(*pkcs11-sequence-number*) | NOSIGSEQNUM]
[SIGCAT(*pkcs11-category*) | NOSIGCAT]
[SIGALG(HS256 | HS384 | HS512) | NOSIGALG]
[ANYAPPL(YES | NO)]
[IDTTIMEOUT(*timeout-minutes*) | NOIDTTIMEOUT]
)
NOIDTPARMS]

IDTPARMS | NOIDTPARMS

IDTPARMS

Specifies information for the IDTDATA class profile being changed.

SIGTOKEN | NOSIGTOKEN

SIGTOKEN(*pkcs11-token-name*)

Specifies the name of an ICSF PKCS#11 token name for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The token name may consist of alphanumeric characters, national characters (@, #, $) and the period symbol. The token name is not case sensitive.

The minimum token name length is 1. The maximum token name length is 32. There is no default value.

NOSIGTOKEN

Deletes the SIGTOKEN in the profile.

SIGSEQNUM | NOSIGSEQNUM

SIGSEQNUM(*pkcs11-sequence-number*)

Specifies the ICSF PKCS#11 sequence number of the key for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The sequence number must be a hexadecimal number.

The minimum sequence number is 1. The maximum sequence number length is

8 hexadecimal digits. The default value is 1.

NOSIGSEQNUM

Deletes the SIGSEQNUM in the profile.

SIGCAT | NOSIGCAT

SIGCAT(*pkcs11-category*)

Specifies the ICSF PKCS#11 category of the key for the generation and validation of Identity Token (IDT) signatures associated with this profile.

The category must be one of the following values:

T – Specifies a clear token object.

Y – Specifies a secure token object.

The default value is T.

NOSIGCAT

Deletes the SIGCAT in the profile.

SIGALG | NOSIGALG

SIGALG(HS256 | HS384 | HS512)

Specifies the signature algorithm for the generation of Identity Token (IDT) signatures associated with this profile. The default value is HS256.

HS256

Specifies the signature algorithm as HMAC with SHA-256.

HS384

Specifies the signature algorithm as HMAC with SHA-384.

HS512

Specifies the signature algorithm as HMAC with SHA-512.

NOSIGALG

Deletes the SIGALG in the profile.

ANYAPPL(YES | NO)

Specifies if the IDT that RACROUTE generates can be used for any application name or only for the application name that performed authentication. The default value is YES.

When ANYAPPL(YES) is specified, RACROUTE will generate the IDT so it can be used for any application name.

When ANYAPPL(NO) is specified, RACROUTE will generate the IDT so that it can only be used by the application name that performed authentication.

When an IDT is generated by RACROUTE which is not for an end user by specifying the IDTA parameter field IDTA_End_User_IDT off, RACROUTE will ignore this setting and generate the IDT so that it can be used with any application name.

IDTTIMEOUT | NOIDTTIMEOUT

IDTTIMEOUT(*timeout-minutes*)

Specifies the number of minutes that the Identity Token (IDT) associated with the profile is active.

The value of timeout-minutes can be between 1 and 1440. The default value is 5.

NOIDTTIMEOUT

Deletes the IDTTIMEOUT in the profile. The default value of 5 goes into effect.

NOIDTPARMS

Deletes the IDTPARMS segment.

3.1.3 RLIST

A new IDTPARMS segment is added. RLIST is enhanced to display IDTPARMS information.

Syntax

[IDTPARMS]

IDTPARMS

Specifies that the IDTPARMS segment information should be listed for profiles in the IDTDATA class.

Example RLIST output for the IDTPARMS segment

```
RLIST IDTDATA JWT.APPL01.USER01.SAF IDTPARMS
...
IDTPARMS INFORMATION
--------------------
SIGNATURE TOKEN NAME = NETHK.TKN1
SIGNATURE SEQUENCE NUMBER = 00000003
SIGNATURE CATEGORY = T
SIGNATURE ALGORITHM = HS256
IDT TIMEOUT = 00000005
ANYAPPL = NO
```


3.2 z/OS Security Server RACF Messages and Codes

This information supplements RACF messages and Codes.

3.2.1 RACF abend codes

Abend code 283 is updated to add new reason codes.

Code	Explanation
74	The IDTA keyword was specified with either the ICTX or ICRX parameter.
78	The IDTA block is not valid. Either the ID, version, length, buffer pointer or type is not valid.

3.2.2 RACROUTE REQUEST=VERIFY Messages

ICH408I LOGON/JOB INITIATION – IDENTITY TOKEN AUTHENTICATION FAILURE

Explanation: User authentication with an Identity Token (IDT) failed.

System Action: RACF prevents the user from logging on.

User Response: Correct any errors in the credentials and try again.

3.3 z/OS Security Server RACF RACROUTE Macro Reference

This information supplements the information in Chapter *System Macros* in the *RACROUTE REQUEST=VERIFY* and *RACROUTE REQUEST=EXTRACT* sections.

3.3.1 RACROUTE REQUEST=VERIFY

Authorization:
The following existing statement regarding authorization is updated to include the IDTA keyword:

To issue the RACROUTE REQUEST=VERIFY macro, the calling module must be authorized (APF-authorized, in system key 0–7, or in supervisor state) or the NEWPASS, PHRASE, NEWPHRASE, ICRX, ICTX, IDID and IDTA keywords must be omitted and the calling module must be in the RACF-authorized caller table and fetched from an authorized library and reentrant.

The RELEASE parameter description is updated:

,RELEASE=number
…
Starting with HRF77C0 or RACF APAR OA55926, the naming convention of the RELEASE keyword is updated to correspond to a parameter list version number. Version PLV0001 is the initial parameter list version number.

- …
- 77A0 corresponds to FMID HRF77A0 (z/OS Security Server V2R2)
- 77B0 corresponds to FMID HRF77B0 (z/OS Security Server V2R3)
- PLV0001 corresponds to RACF APAR OA55926 or FMID HRF77C0 (z/OS Security Server V2R4)

The SYSTEM parameter description is updated:
SYSTEM=YES is used to provide a fast path though RACROUTE. The IDTA parameter is added to the list of keywords which nullify the benefits of the SYSTEM=YES fast path.

The IDTA parameter is added:

,IDTA=idta data addr

Specifies the address of the data structure that describes an Identity Token (IDT) to be generated or validated by RACROUTE. The address points to a data structure defined in Table A.

An application can request that RACROUTE generate an IDT when authenticating a user. An application can also specify an IDT in place of other authentication credentials like a password.

The IDTA parameter is not processed by RACROUTE unless the IDTDATA class is active.

The Identity Token is useful for two main authentication scenarios:
1) Linking Multiple Authentication API Calls
2) Replaying Proof of Authentication

Linking Multiple Authentication API Calls:
In some cases, an application which performs user authentication may be required to call the RACROUTE authentication APIs multiple times to complete the authentication process. Applications can use an IDT to allow RACROUTE to link authentication status information between these multiple authentication API calls.

An example of an authentication scenario which requires multiple API calls is when a “password expired” event occurs. An application may initially attempt to authenticate a user by prompting a user for a password and subsequently call RACROUTE REQUEST=VERIFY. This RACROUTE call may fail with a return code which indicates that the password is expired. The application must then prompt the user for a new password and re-call RACROUTE REQUEST=VERIFY again for the same user with an updated parameter list. The IDTA parameter allows these multiple API calls to be linked together for more intelligent authentication processing. This is especially important when users are authenticated with one time use Multi-Factor Authentication (MFA) credentials.

Replaying Proof of Authentication:
Some applications have a need to authenticate a user, and then replay that authentication multiple times. These applications may cache the user provided credential and replay it back to RACROUTE. This authentication pattern does not work well for users provisioned with single use MFA token codes instead of passwords. The Identity Token support allows applications to authenticate a user and receive proof of that authentication. The returned IDT can be specified on subsequent calls to RACROUTE instead of other authentication credentials like a password. A signed IDT can be returned to an end user for later use by an application.

Identity Token Formats:
The Identity Token Area (IDTA) is a flexible interface designed to support different types of Identity Tokens (IDT) going forward. Currently RACROUTE has support for an IDT in the format of a JSON Web Token (JWT). In the future RACROUTE may support other IDT formats.

RACROUTE JWT Claims:
The IDT contains information regarding the end user including how the user was authenticated. When the IDT is in the format of a JSON Web Token (JWT) this information is encoded in a set of JWT claims. Some of the claim values that RACROUTE encodes in the JWT can be controlled by the IDTDATA class profile.

The supported JWT claims are (all required):

- **The “iss” (Issuer) Claim:**
  - The “iss” claim is used to identify the issuer of the JWT.
  - For JWT generation, the “iss” value is set to “saf”.
  - For JWT validation, the “iss” value must be set to “saf”.
- **The “sub” (Subject) Claim:**
  - The “sub” claim is used to identify the subject user ID for the JWT.
  - For JWT generation, the “sub” value is set to the USERID parameter.
  - For JWT validation, the “sub” value is used as the RACROUTE USERID. When the USERID parameter is also specified it must match the “sub” value.
- **The “aud” (Audience) Claim:**
  - The “aud” claim is used to identify which application names can use this JWT. When a JWT “aud” claim contains the “*ANYAPPL*” string it can be used by any application.
  - For JWT generation, the “aud” value includes the specified APPL application name parameter or the default application name if one was not specified. An IDTDATA class profile can be created to configure if the “*ANYAPPL*” string is also included in “aud” value. By default, the “*ANYAPPL*” string is included in the “aud” value.
  - For JWT validation, the “aud” must contain a value that matches the APPL parameter or the “aud” must contain a value that matches the default application name when an APPL parameter is not specified or the “aud” must contain the “*ANYAPPL*” string.
- **The “exp” (Expiration Time) Claim:**
  - The “exp” claim is used to indicate JWT expiration time.
  - For JWT generation, an IDTDATA class profile can be created to configure the JWT timeout value. By default, the “exp” value is set using a timeout value of 5 minutes.
  - For JWT validation, the “exp” value must not be less than the current time.
- **The “iat” (Issued At) Claim:**
  - The “iat” claim is used to indicate the time the JWT was issued.
  - For JWT generation, the “iat” value is set to the current time.
  - For JWT validation, the “iat” value must be a number.
- **The “jti” (JWT ID) Claim:**
  - The “jti” claim is used to encode a unique identifier for the JWT.
  - For JWT generation, the “jti” value is set to a unique value.
  - For JWT validation, the “jti” value must be at least 8 characters long and no more than 64 characters long.
- **The “txn” (Transaction Identifier) Claim:**
  - The “txn” claim is used to encode a unique identifier which can be shared between a set of related JWTs.
  - For JWT generation, when an input JWT was not specified the “txn” value is set to a unique value. When an input JWT was specified the output “txn” value is set to same “txn” value as the input JWT.
  - For JWT validation, “txn” value must be at least 8 characters long and no more than 64 characters long.

- **The “amr” (Authentication Method References) Claim:**
  - The “amr” claim is used to indicate which methods were used to authenticate the subject.
  - For JWT generation, the “amr” values are set based on which methods were used to authenticate the user.
  - For JWT validation, the “amr” values are used as part of the authentication process.
  - More details on the “amr” claim are provided below.

RACROUTE use of the AMR Claim:
The “amr” claim is used by RACROUTE to indicate which methods were used to authenticate the subject. The RACROUTE authentication methods are divided into MFA methods and SAF methods. In some cases, there may be both MFA and SAF authentication methods for a single user.

For JWT generation, the “amr” are created values based on the methods that were used to authenticate the user.
For JWT validation, the “amr” values are used as part of the authentication process. RACROUTE will validate that the “amr” values are a valid combination of the supported values as listed below.

These are the “amr” claim values for the SAF authentication methods:
1. **“saf-pwd”** – The user was authenticated with a password.
2. **“saf-phr”** – The user was authenticated with a password phrase.
3. **“saf-ptkt”** – The user was authenticated with a PassTicket.

These are the “amr” claim values for the MFA authentication methods:
1. **“mfa-only”** – The user was authenticated with MFA.
2. **“mfa-ptkt”** – An MFA provisioned user was authenticated with a PassTicket.
3. **“mfa-comp”** – The user was authenticated with MFA. A SAF authentication method name (“saf-pwd” or “saf-phr”) is required.
4. **“mfa-pwfb”** – An MFA provisioned user was not authenticated with MFA. A SAF authentication method name is required.
5. **“mfa-bypass”** – An MFA provisioned user was not authenticated with MFA for this application. A SAF authentication method name is also required. A JWT with this claim can only be used on an application which is configured to be bypassed.
6. **“mfa-exp”** – The user was authenticated with MFA, but the knowledge factor portion of the credential was expired. The user must set a new knowledge factor before they can successfully logon.
7. **“mfa-newinv”** – The user was authenticated with MFA, but the user attempted to set a new knowledge factor which was not valid. The user must set a new knowledge factor before they can successfully logon.
8. **“mfa-nmi”** – MFA processing requires more information to complete authentication.

In some cases, a JWT may be returned when the user is not yet fully authenticated. The following MFA “amr” claims indicate that the user is still in the process of being

authenticated: “mfa-exp”, “mfa-newinv” and “mfa-nmi”. A JWT with these claims may be specified back to RACROUTE along with any other required parameters to complete the authentication process. RACROUTE may then generate a new output JWT with updated “amr” claims.

Signed and Unsigned Identity Tokens:
RACROUTE can create a signed or unsigned IDT. The security administrator may configure profiles in the IDTDATA class to control how RACROUTE creates the IDT including which ICSF key is used to generate and validate the IDT signature. An installation can create the key in ICSF by calling the PKCS#11 Generate Secret Key (CSFPGSK) or Token Record Create (CSFPTRC) callable services.

Applications may return a signed IDT to an end user. An unsigned IDT should not be returned to an end user because RACROUTE will not accept an unsigned IDT from an end user. When an application will return an IDT or accept an IDT from and end user, it must turn on the IDTA_End_User_IDT flag. When the IDTA_End_User_IDT is on, RACROUTE will not return an unsigned IDT or accept a specified unsigned IDT. An authorized application which keeps an unsigned IDT under its own control may generate and specify an unsigned IDT to RACROUTE.

When a signed IDT is specified, and signature validation fails RACROUTE returns the 8/8/0 return codes and the specified user’s revoke count is incremented.

When a signed IDT is specified, and there is no associated IDTDATA class profile with a signature key configured, RACROUTE indicates that signature validation cannot be performed by returning the 8/6C/15 return codes.

The JWT signature algorithm is encoded in the “alg” claim value in the JWT header. RACROUTE supports generating and validating JWTs with the “HS256” HMAC256, “HS384” HMAC384 or “HS512” HMAC512 signature algorithms. RACROUTE does not support encrypted JWTs.

Protecting an IDT:
When an application receives an Identity Token (IDT) from RACROUTE, it must keep it in a protected location. An IDT can be used to authenticate a user though RACROUTE and therefore must be kept in protected storage. Also note that some applications may allow IDTs to be specified from an end user.

Identity Token Expiration:
An IDT has a defined expire date, after which it is no longer considered valid. When an expired IDT is passed into RACROUTE REQUEST=VERIFY the request will fail with the 8/6C/F return codes. The calling application should discard the expired IDT, and re-prompt the user for authentication credentials before calling RACROUTE again.

When RACROUTE creates an IDT the default timeout value is 5 minutes after creation. The

security administrator can create an IDTDATA class profile to configure a different timeout value.

Expired Password or Password Phrase:
An IDT includes information regarding how the user originally authenticated. When the user is being authenticated with an IDT which indicates the original authentication was a password or password phrase and that authenticator is expired on the system, RACROUTE will fail the request with the 8/C/0 “password or password phrase expired” return codes. In this case the user must provide a new password or password phrase before the RACROUTE will complete with a successful return code. The application should prompt the user to enter their new password or new password phrase value before calling RACROUTE again.

Changing the password or password phrase with a specified Identity Token:
A valid specified IDT can be used instead of the current password or password phrase when specifying a new password or new password phrase. The normal password change rules apply. When the IDT indicates that the user has authenticated with a password a new password may be specified. When the IDT indicates the user has authenticated with a password phrase a new password phrase may be specified. When the IDT indicates the user has authenticated with a PassTicket a new password or new password phrase may be specified.

IDT Configuration:
The security administrator can configure an IDTDATA class profile to control how certain fields in an IDT are generated and how a specified IDT is validated. The IDTDATA class profile name is based on the IDT type, application name, user ID and IDT issuer. Generic characters are allowed in the profile name.

IDTDATA class profile format:

```
<IDT Type>.<application name>.<user ID>.<IDT issuer name>
```

For example, the following command will create an IDTDATA class profile for an IDT type of ‘JWT’, an application name of ‘APPL01’, user ID of ‘USER01’ and an issuer of ‘saf’:

```
RDEFINE IDTDATA JWT.APPL01.USER01.SAF IDTPARMS(SIGTOKEN(mytoken)
SIGSEQNUM(1) SIGCAT(T) SIGALG(HS256) ANYAPPL IDTTIMEOUT(30))
```

For more information on configuring IDT parameters refer to the section containing the command language reference updates.

When the IDTDATA class is not active, RACROUTE will not process the IDTA parameter. When the IDTA is specified with a supplied IDT and the IDTDATA class is not active, RACROUTE will return the 8/6C/1A return codes and set the IDTA_IDT_Len to zero.

The IDTDATA class must be active and RACLISTed before RACROUTE will use any profiles for the generation or validation of an IDT.
When there is no covering IDTDATA class profile, RACROUTE will generate an IDT with

the default values. An IDT created with the default values will be unsigned, be accepted by any application name and will have a timeout value of 5 minutes.

IDTA Parameter Format:
The IDTA parameter is used to generate an Identity Token or supply an input Identity Token. The IDTA format is described in table A.

Table A: IDTA - Identity Token Area – (Mapped by SAF Macro IRRPIDTA)

Name	Len	In/Out	Description
IDTA_ID	4	Input	**Eyecatcher** – "IDTA".
IDTA_Version	2	Input	**Version** - "1".
IDTA_Length	2	Input	**Length** - "36" – Total length of the IDTA.
IDTA_IDT_Buffer_Ptr	4	Input	**Identity Token Buffer Pointer** – Points to a caller allocated buffer for the Identity Token.
IDTA_IDT_Buffer_Len	4	Input	**Identity Token Buffer Length** – Length of the Identity Token buffer. The current minimum size is 1024. In a future update the minimum size may be increased. When the input buffer length is smaller than the minimum or insufficient to hold the output IDT, RACROUTE will return the return codes 8/70/1 and the required size will be set in the IDTA_IDT_Len Identity Token Length field.
IDTA_IDT_Len	4	Input / Output	**Identity Token Length** – Length of the Identity Token. Caller should set to zero if there is no supplied Identity Token. When RACROUTE generates an IDT, the IDT length will be set on output. A new output IDT may be written over an existing input IDT. When the input IDTA_IDT_Buffer_Len Token Buffer Len field is not sufficient size to hold the output token, RACROUTE will set the IDTA_IDT_Len to the required size and fail with the return codes 8/70/1. The caller must reallocate the buffer with the larger size and reset to the IDTA_IDT_Len to zero or the length of an supplied IDT. When an there is an error processing a specified IDT RACROUTE will set the IDTA_IDT_Len to zero.
IDTA_IDT_Type	2	Input	**Identity Token Type** – Indicates the IDT type. **X'0001'** – Indicates IDT is a JSON Web Token (JWT). All other values reserved.
IDTA_IDT_Gen_RC	2	Output	**Identity Token Generation Return Code:** **X'0000'** – **IDTA_IDT_GEN_RC_SUCC -** When IDTA_SAF_IDT_Return is set ON, successfully generated IDT. When IDTA_SAF_IDT_Return is set OFF, did not attempt to generate IDT. **X'0003'** – **IDTA_IDT_GEN_RC_UNSIGN** - The IDTA_End_User_IDT is set ON but signed IDTs are not configured.

			X'0004' – **IDTA_IDT_GEN_RC_ICSF_UNAVIL** – ICSF is not available to generate signature. **X'0005'** – **IDTA_IDT_GEN_RC_ICSF_ERR -** ICSF error detected attempting to generate signature.
IDTA_IDT_Prop_Out	2	Output	**Identity Token Output Properties bits:** **IDTA_SAF_IDT_Return - Bit 1** – Identity Token Returned – IDT was returned by SAF. **IDTA_IDT_Auth_Done - Bit 2** – Authentication Complete – IDT returned is fully authenticated. **IDTA_IDT_Signed - Bit 3** – Identity Token is signed – IDT returned is signed. **Bits 4-16** – Reserved.
IDTA_IDT_Prop_In	2	Input	**Identity Token Input Properties bits:** **IDTA_End_User_IDT - Bit 1** – Identity token will be used by an end user. When this bit is set on during Identity Token generation and signed Identity Tokens are not configured, RACROUTE will not return an unsigned Identity Token and instead set the **IDTA_IDT_Gen_RC** to **IDTA_IDT_GEN_RC_UNSIGN**. When this bit is set on during Identity Token validation, RACROUTE will not accept an unsigned Identity Token and instead return the Return Codes: 8/6C/14. **Bits 2-16** – Reserved
*	8	N/A	Reserved

Initial RACROUTE REQ=VERIFY Call with an IDTA:
When an application initially calls RACROUTE REQEST=VERIFY for a user the IDTA parameter should be setup as follows:

Header Fields:

- Set the IDTA_ID to "IDTA".
- Set the IDTA_VERSION to 1.
- Set the IDTA_LENGTH to 36.

IDT Buffer Parameters Fields:

- Set IDTA_IDT_Type to 1.
- Set IDTA_IDT_Buffer_Ptr to point to a caller-allocated buffer for returning an IDT.
- Set the IDTA_IDT_Buffer_Len to the length of the allocated IDT buffer. The minimum IDT buffer length is 1024.
- When an IDT is not specified, set IDTA_IDT_Len to 0.
  When an IDT is specified, copy the supplied IDT into the IDT buffer and set the IDTA_IDT_Len to the length of the supplied IDT.
- When the IDT is provided by an end user or the IDT is going to be returned to an end user, set the IDTA_End_User_IDT flag on. When IDTA_End_User_IDT is off, an unsigned IDT will be accepted by RACROUTE and may be returned by RACROUTE.

Other fields: Initialize all other fields to binary zero.

Subsequent RACROUTE REQ=VERIFY Calls with an IDTA:
On subsequent calls to RACROUTE REQUEST=VERIFY for the same user, within the same set of authentication API calls, the IDTA parameter should be passed back in as it was returned from the previous RACROUTE call. Depending on the authentication scenario RACROUTE may overwrite a supplied IDT with a new IDT within the IDT buffer.

When an application switches to authenticate a different user ID, the IDTA should be reinitialized as new. When the IDTA is not reinitialized, the RACROUTE call will fail when the IDTA token user ID does not match the supplied RACROUTE user ID.

IDTA Error Handling:
Applications which specify the IDTA keyword must be prepared to handle certain error scenarios.

When there is an error processing a specified IDT RACROUTE will return one of the 8/6C/x return code combinations. In this case, the IDTA_IDT_Len field to also set to zero.

In some cases, an IDT may not be generated when an application has requested one. When an IDT is successfully generated, the IDTA_SAF_IDT_Return field will be set ON. When an IDT is not generated the IDTA_SAF_IDT_Return field will be set OFF. When there was an error encountered attempting to generate the IDT, the IDTA_IDT_Gen_RC field will be set to indicate the error reason.

When the IDTA keyword is specified and the IDTDATA class is active RACROUTE may return the 8/74/1 return code combination in some cases while authenticating MFA provisioned users. In this case, the application should prompt the user for current credentials again and call RACROUTE with the returned IDT.

Other RACROUTE REQUEST=VERIFY Parameters:
When a specified IDTA contains a supplied IDT, RACROUTE attempts to use the IDT for authentication. When the supplied IDT contains a SAF AMR claim the PASSWORD and PHRASE parameters are ignored by SAF.

When an IDT is supplied and the USERID parameter is omitted, the subject name from the IDT will be used as the user ID.

The new password or new password phrase parameters may be specified with a specified IDT. The normal new password or new password phrase rules apply.

The IDTA keyword is only processed when RELEASE is set to PLV0001 or higher and when the ENV keyword is set to CREATE.

The IDTA keyword is not valid when the ICRX or IDID keywords are specified.

The IDTA keyword is ignored when PASSCHK=NO or ENVIRIN is specified.

The IDTA keyword is only processed with REQUEST=VERIFY.

New RACROUTE REQUEST=VERIFY Return codes and reason codes:

SAF RC
Meaning
8 Requested function has failed.

RACF RC
Meaning
6C Indicates that Identity Token (IDT) processing failed while attempting to validate an IDT. RACROUTE sets the IDTA_IDT_Len to zero. The calling application should reauthenticate the user.

Reason Code
Meaning
1 Memory error parsing supplied IDT.
2 Error parsing IDT structure.
3 Error Base64 decoding IDT.
4 Error JSON parsing IDT.
5 IDT Subject is not valid.
6 IDT Subject does not match the current user ID.
7 IDT Audience is not valid.
8 IDT Audience does not match the current application name.
9 IDT Signature Algorithm not valid.
A IDT Signature Algorithm does not match the SIGALG from the IDTDATA class profile.
B IDT Authentication Method References not valid.
C IDT Authentication Method References indicates MFA authentication for non-MFA user.
D IDT Authentication Method References indicates SAF authentication for MFA user.
E IDT Expiration Date is not valid.
F IDT Expiration Date indicates IDT is expired.
10 IDT Signature Algorithm is not supported.
11 IDT Unique ID is not valid.
12 IDT Transaction ID is not valid.
13 IDT Issuer is not valid.
14 IDT is not signed but is specified from an end user.
15 IDT is signed but key is not configured in IDTDATA class profile.
16 ICSF is not available to validate signature.
17 ICSF error detected attempting to validate signature.
18 Error attempting to call MFA to process MFA AMR claim.
19 IDT Authentication Method References indicates SAF fallback for NOPWFALLBACK user.
1A IDT supplied when the IDTDATA class is not active.
1B IDT Issued At is not valid.

70 Indicates that Identity Token (IDT) processing has failed.
Reason Code
Meaning
1 IDT buffer length insufficient. The IDT length field has been updated to the required size.
2 Specified User ID is not valid for IDT generation.
6 The ICHRIX02 postprocessing exit returned RC 4 with the IDTA keyword specified and the IDTDATA class active.

74 Indicates Multi-Factor Authentication processing has failed.
This return code is only returned when the IDTA parameter is specified and the IDTDATA class is active. When the IDTA keyword is not specified or the IDTDATA class is not active, the 8/8/0

return codes are returned instead.

Reason Code	Meaning
1	MFA processing needs more information to complete authentication.

3.4 z/OS Security Server RACF Callable Services

This information supplements the information in Chapter *Callable services descriptions*

- The *R_Admin* section is updated to add new IDTPARMS segment and fields

3.4.1 R_Admin (IRRSEQ00): RACF administration API

The R_admin reference appendix is updated:

- The table *IDTPARMS fields* is added

IDTPARMS segment fields:

Field name	SAF field name	Flag byte value	RDEFINE/RALTER keyword reference	Allowed on add requests	Allowed on alter requests	Returned on extract requests
SIGTOKEN	sigtoken	'Y'	IDTPARMS (SIGTOKEN(xx))	Yes	Yes	Yes
		'N'	IDTPARMS (NOSIGTOKEN)	No	Yes	
SIGSEQN	sigseqnum	'Y'	IDTPARMS (SIGSEQNUM(xx))	Yes	Yes	Yes
		'N'	IDTPARMS (NOSIGSEQNUM)	No	Yes	
SIGCAT	sigcat	‘Y’	IDTPARMS (SIGCAT)	Yes	Yes	Yes
		‘N’	IDTPARMS (NOSIGCAT)	No	Yes	
SIGALG	sigalg	'Y'	IDTPARMS (SIGALG(xx))	Yes	Yes	Yes
		'N'	IDTPARMS (NOSIGALG)	No	Yes	
IDTTIMEO	idttimeout	'Y'	IDTPARMS (IDTTIMEOUT(xx))	Yes	Yes	Yes
ANYAPPL	anyappl	'Y'	IDTPARMS (ANYAPPL(YES))	Yes	Yes	Yes
		'N'	IDTPARMS (ANYAPPL(NO))	Yes	Yes	

3.5 z/OS Security Server RACF Macros and Interfaces

This information supplements information in the following chapters and sections:

- Chapter *RACF database unload* in the Record formats produced by the database unload utility section.
- Chapter *SMF records* in the Format of SMF type 80 records section.
- Chapter *The format of the unloaded SMF type data* in the The JOBINIT record extension section.
- Appendix *Supplied class descriptor table entries.*
- Appendix *RACF database templates* in the User template for the RACF database and General template for the RACF database sections.

3.5.1 RACF database unload

1 Record formats produced by the database unload utility

The general resource IDTPARMS definition record (05K0) is added.

Field Name	Type	Start	End	Comments
GRIDTP_RECORD_TYPE	Int	1	4	Record type of the Identity Token data record (05K0)
GRIDTP_NAME	Char	6	251	General resource name as taken from the profile name.
GRIDTP_CLASS_NAME	Char	253	260	Name of the class to which the general resource profile belongs, namely IDTDATA.
GRIDTP_SIG_TOKEN_NAME	Char	262	293	The ICSF PKCS#11 token name.
GRIDTP_SIG_SEQ_NUM	Char	295	302	The ICSF PKCS#11 sequence number.
GRIDTP_SIG_CAT	Char	304	307	The ICSF PKCS#11 category.
GRIDTP_SIG_ALG	Char	309	340	The signature algorithm.
GRIDTP_TIMEOUT	Int	342	351	IDT timeout setting.
GRIDTP_ANYAPPL	Char	353	355	Is the IDT allowed for any application? Valid values include “Yes” and “No”.

3.5.2 SMF records

Type 80 event code 1 (RACROUTE REQ=VERIFY/X) record:

New event code qualifiers are added for the Type 80 event code 1 (RACROUTE REQ=VERIFY/X) record:

- 44(2C) – IDTVALF – Identity Token validation error
- 45(2D) – IDTF – Identity Token error
- 46(2E) – INVIDT – Failed Identity Token authentication

The “Table of extended-length relocate section variable data” is updated to add a new bit to indicate that authentication is from an Identity Token.

<table>
<tr><th>Data type (SMF80TP2) dec(hex)</th><th>Data length (SMF80DL2)</th><th>Format</th><th>Audited by event code</th><th>Description (SMF80DA2)</th></tr>
<tr><td>443(1BB)</td><td>variable</td><td>mixed</td><td>1</td><td>Byte 1: Authentication information:
<table>
<tr><th>Bit</th><th>Meaning when set</th></tr>
<tr><td>0</td><td>Authenticated from VLF</td></tr>
<tr><td>1</td><td>User has active MFA factor(s)</td></tr>
<tr><td>2</td><td>MFA user allowed to fall back when no MFA decision can be made</td></tr>
<tr><td>3</td><td>No MFA decision for MFA user</td></tr>
<tr><td>4</td><td>IBMMFA requested that RACROUTE REQUEST=VERIFY return the password-expired return code.</td></tr>
<tr><td>5</td><td>IBM MFA requested that RACROUTE REQUEST=VERIFY return the new-password-invalid return code.</td></tr>
<tr><td>6</td><td>IBM MFA requested that RACROUTE REQUEST=VERIFY return the password-invalid return code, but not to increment the password revoke count (partial success – needs more information).</td></tr>
<tr><td>7</td><td>~~Reserved for IBM's use~~ Relocate 443 is extended.</td></tr>
</table></td></tr>
</table>

Byte 2: Authenticator used:

Bit	Meaning when set
0	Password Evaluated
1	Password Successful
2	Password Phrase Evaluated
3	Password Phrase Successful
4	PassTicket Evaluated
5	PassTicket Successful
6	MFA authentication successful
7	MFA authentication unsuccessful

Byte 3-6: Authorization Reason Code 1

Bytes 7-10: Authorization Reason Code 2

Note: Below fields are only present when relocate 443 is extended.

Byte 11-14: Authorization Reason Code 3

Bytes 15-18: Authorization Reason Code 4

Byte 19: Flag byte 3: Authentication Details

Bit	Meaning when set
0	Password or Password Phrase expired
1	New Password or Password Phrase invalid
2	Identity Token (IDT) Evaluated
3	Identity Token (IDT) Successful
4	IBM MFA requested that RACROUTE REQUEST=VERIFY return the password-invalid return code, but not to increment the password revoke count (reauthentication requested).
5-7	Reserved

				Byte 20: Flag byte 4: Authentication Details <table><tr><th>Bit</th><th>Meaning when set</th></tr><tr><td>0-7</td><td>Reserved</td></tr></table> Bytes 21-28: Derived Application Name Byte 29-32: IDT Validation Reason Code Byte 33-36: IDT Error Reason Code Byte 37-40: Failing Service ID Byte 41-44: Failing Service Return Code Byte 45-48: Failing Service Reason Code

3.5.3 The format of the unloaded SMF type 80 data

2 The JOBINIT record extension

The JOBINIT record extension relocate section 443 is updated to reuse the former reserved field INIT_RESERVED_01 as INIT_RELO443_EXTENDED.

In addition, the below highlighted fields are added to the unloaded JOBINIT record extension based on the new extended relocate section 443.

Field Name	Type	Start	End	Comments
INIT_ACEE_VLF	Yes/ No	4540	4543	The ACEE was created from the VLF cache
INIT_MFA_USER	Yes/ No	4545	4548	The user has active MFA factors
INIT_MFA_FALLBACK	Yes/ No	4550	4553	The MFA user is allowed to fall back to password authentication when MFA is unavailable
INIT_MFA_UNAVAIL	Yes/ No	4555	4558	MFA was unavailable to make an authentication decision for the MFA user
INIT_MFA_PWD_EXPIRED	Yes/ No	4560	4563	IBM MFA requested that RACROUTE REQUEST=VERIFY return the

				password-expired return code
INIT_MFA_NPWD_INV	Yes/ No	4565	4568	IBM MFA requested that RACROUTE REQUEST=VERIFY return the new-password-invalid return code
INIT_MFA_PART_SUCC	Yes/ No	4570	4573	IBM MFA requested that RACROUTE REQUEST=VERIFY return the password-invalid return code, but not to increment the password revoke count (partial success – needs more information).
~~INIT_RESERVED_01~~ INIT_RELO443_EXTENDED	Yes/ No	4575	4578	~~Reserved for IBM's use~~ Relocate 443 is extended.
INIT_PASSWORD_EVAL	Yes/ No	4580	4583	The supplied password was evaluated
INIT_PASSWORD_SUCC	Yes/ No	4585	4588	The supplied password was evaluated successfully
INIT_PHRASE_EVAL	Yes/ No	4590	4593	The supplied password phrase was evaluated
INIT_PHRASE_SUCC	Yes/ No	4595	4598	The supplied password phrase was evaluated successfully
INIT_PASSTICKET_EVAL	Yes/ No	4600	4603	The supplied password was evaluated as a PassTicket
INIT_PASSTICKET_SUCC	Yes/ No	4605	4608	The supplied password was evaluated successfully as a PassTicket
INIT_MFA_SUCC	Yes/ No	4610	4613	The supplied password phrase/phrase was evaluated successfully as multifactor data
INIT_MFA_FAIL	Yes/ No	4615	4618	The supplied password/phrase was evaluated unsuccessfully as MFA data
INIT_AUTH_RSN1	Char	4620	4627	MFA Authentication return code. Expressed as hexadecimal number.
INIT_AUTH_RSN2	Char	4629	4636	MFA Authentication reason code. Expressed as hexadecimal number.
INIT_AUTH_RSN3	Char	4638	4645	PassTicket Authentication return code. Expressed as hexadecimal number.
INIT_AUTH_RSN4	Char	4647	4654	PassTicket Authentication reason code. Expressed as hexadecimal number.
INIT_PWD_PHR_EXPIRED	Yes/ No	4656	4659	The supplied password or password phrase was expired.

INIT_NPWD_NPHR_NONVAL	Yes/ No	4661	4664	The supplied new password or new password phrase was not valid.
INIT_IDT_EVAL	Yes/ No	4666	4669	The supplied Identity Token (IDT) was evaluated.
INIT_IDT_SUCC	Yes/ No	4671	4674	The supplied Identity Token (IDT) was evaluated successfully.
INIT_MFA_REAUTHENT	Yes/ No	4676	4679	IBM MFA requested that RACROUTE REQUEST=VERIFY return the password-invalid return code, but not to increment the password revoke count (reauthentication requested).
INIT_RESERVED_01	Yes/ No	4681	4684	Reserved for IBM's use
INIT_RESERVED_02	Yes/ No	4686	4689	Reserved for IBM's use
INIT_RESERVED_03	Yes/ No	4691	4694	Reserved for IBM's use
INIT_RESERVED_04	Yes/ No	4696	4699	Reserved for IBM's use
INIT_RESERVED_05	Yes/ No	4701	4704	Reserved for IBM's use
INIT_RESERVED_06	Yes/ No	4706	4709	Reserved for IBM's use
INIT_RESERVED_07	Yes/ No	4711	4714	Reserved for IBM's use
INIT_RESERVED_08	Yes/ No	4716	4719	Reserved for IBM's use
INIT_RESERVED_09	Yes/ No	4721	4724	Reserved for IBM's use
INIT_RESERVED_10	Yes/ No	4726	4729	Reserved for IBM's use
INIT_RESERVED_11	Yes/ No	4731	4734	Reserved for IBM's use
INIT_DERIVED_APPL_NAM	Char	4736	4743	Derived Application Name
INIT_IDT_VALIDTN_RSNC	Char	4745	4752	IDT Validation Reason Code
INIT_IDT_ERROR_RSNC	Char	4754	4761	IDT Error Reason Code
INIT_SERVICE_CODE	Char	4763	4770	Failing Service Identifier

INIT_SERVICE_RC	Char	4772	4779	Failing Service Return Code
INIT_SERVICE_RSNC	Char	4781	4788	Failing Service Reason Code

3.5.4 RACF database templates

The RACF database templates version string is updated to:
$/VERSION OA55926 00000223.00000050

The IDTPARMS segment is added to the GENERAL section.

```
$/SEGMENT  021 IDTPARMS
IDTPARMS  001 00 00 00000000 00 IDTPARMS - Start of segment fields
IDTTOKN   002 00 00 00000000 00 IDTPARMS - PKCS#11 Token Name
IDTSEQN   003 00 00 00000000 00 IDTPARMS - PKCS#11 Sequence Number
IDTCAT    004 00 00 00000000 00 IDTPARMS - PKCS#11 Category
IDTSALG   005 00 00 00000000 00 IDTPARMS - Signature Algorithm
IDTTIMEO  006 00 00 00000004 00 IDTPARMS - IDT Timeout
IDTANYAP  007 00 00 00000001 80 IDTPARMS - IDT Any Application
```

3.5.5 Class Descriptor Table

A new IDTDATA general resource class is added.

ICHERCDE macro keyword
CLASS=IDTDATA
POSIT=606
ID=1
PROFDEF=YES
MAXLNTH=246
CASE=UPPER
FIRST=NONATNUM
OTHER=ANY
OPER=NO
KEYQUAL=0
DFTRETC=4
DFTUACC=NONE

RACLIST=ALLOWED
RACLREQ=YES
SIGNAL=NO
GENERIC=ALLOWED
GENLIST=DISALLOWED
SLBLREQ=NO
RVRSMAC=NO
EQUALMAC=NO

3.6 z/OS Security Server RACF Data Areas

This information supplements the information in the *RACF Data Areas* chapter.

3.6.1 RCVT: RACF Communication Vector Table

The RACF Communication Vector Table adds a field to indicate that the IDT Function is available. Other products can check this bit to determine if the current version of RACF has IDT support added either in the base OS or via PTF.

Offset (dec)	Offset (Hex)	Type	Len	Name(Dim)	Description
...					
640	280	BITSTRING	1	RCVTFLG4	Function availability bits
	.1..			RCVTMFA3	MFA3 Functions (OA50930) are available.
	..1.			RCVTIDT	IDT Functions (OA55926) are available.
...					

3.6.2 RIPL: RACROUTE REQUEST=TOKENBLD/VERIFY/VERIFYX Parameter List (Request Section)

The PLV0001 PLIST adds a new field to indicate that the IDTA parameter was specified.

Offset (dec)	Offset (Hex)	Type	Len	Name(Dim)	Description
...					
144	(90)	CHARACTER	*	INIT_PLV0001	RELEASE PLV0001
144	(90)	ADDRESS	4	INIT_IDTA	IDTA ADDRESS
148	(94)	CHARACTER	*	INITEND_PLV0001	END OF PLV0001

3.6.3 RIXP: RACROUTE REQUEST=VERIFY/VERIFYX Exit Parameter List

Add a new field to indicate that the IDTA parameter was specified.

Offset (dec)	Offset (Hex)	Type	Len	Name(Dim)	Description
...					
204	(CC)	ADDRESS	4	RIX_IDTA	IDTA ADDRESS: points to an IDTA data area as mapped by IRRPIDTA.

3.7 z/OS Security Server RACF System Programmer's Guide

This information supplements the information in Chapter *RACF installation exits* in the *RACROUTE REQUEST=VERIFY(X)* Exits section.

3.7.1 Postprocessing exit (ICHRIX02)

Return Codes from the RACROUTE REQUEST=VERIFY(X) postprocessing exit
The existing description for return codes is amended to add the following note:

Note: If the IDTA keyword is specified and the IDTDATA class is active the ICHRIX02 return code 4 is not valid. In this case RACROUTE REQ=VERIFY will fail with the 8/70/6 return codes.

4 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.