RACF Identity Token (IDT) 2 Support:
APARs: RACF OA63462, SAF OA63463

Summary of Changes		
Version	**Date**	**Nature of Change**
V1.00	7/2023	Initial version

1 Introduction

RACROUTE Support for Identity Tokens:
In z/OS V2.4 and RACF APAR OA55926 and SAF APAR OA55927 support was introduced for RACF/SAF Identity Tokens (IDT). In the PTFs for these new APARs, the RACF support for Identity Tokens is enhanced to add support for several new use cases.

Generation of IDTs From an Existing ACEE Security Environment:
The existing RACF IDT support is limited to generating an IDT during RACROUTE authentication processing for users with traditional authentication mechanisms like a password. Some applications have a need to generate an IDT when an ACEE security environment already exists. This new support introduces the capability for applications to use the initACEE callable service to generate an IDT from an ACEE. The returned IDT can be used to authenticate the user using the existing IDT support in RACROUTE REQEST=VERIFY. The new initACEE function can generate an IDT for a protected user without traditional authentication mechanisms.

Authentication of a User with IDT Generated from an ACEE:
The existing RACF IDT support is limited to validating an IDT for users with traditional authentication mechanisms. This new support introduces the capability for RACROUTE REQEST=VERIFY to authenticate a user with an IDT which was generated from an ACEE. With this new support a user with the protected attribute can optionally be authenticated with an IDT.

Restriction: The ISPF panels are not updated for the new command operands with OA63462 and OA63463.

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 IDT2 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 post processing exits can alter the behavior of RACROUTE REQ=VERIFY authentication processing. When the IDTDATA class is activated, RACROUTE REQ=VERIFY will process the IDTA parameter for any application which has specified it.
With this new support a protected user can be authenticated with RACROUTE REQUEST=VERIFY,ENVIR=CREATE,PASSCHK=YES,IDTA= when an IDT is specified and the associated IDTDATA profile is configured to permit it with the PROTALLOWED(YES) keyword. The PROTALLOWED keyword defaults to NO.
Before allowing protected users to be authenticated with an IDT by setting the PROTALLOWED(YES) keyword in a IDTDATA class profile, the installation must ensure that any ICHRIX01 and ICHRIX02 exits are compatible with RACROUTE Identity Token processing a protected user with PASSCHK=YES.

2.4 Performance considerations

No identified performance considerations.

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 Security Administrator's Guide	SA23-2289
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

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 Security Administrator's Guide

This document supplements information for the following sections:

- Defining users

3.1.1 Defining users

…

Defining Protected user IDs

You can define a protected user ID by assigning the NOPASSWORD, NOPHRASE, and NOOIDCARD attributes through the ADDUSER or ALTUSER command. Protected user IDs are protected from being used to logon to the system and from being revoked through inactivity or unsuccessful attempts to access the system using incorrect passwords and password phrases. However, they can be revoked using the ALTUSER (userid) REVOKE command. If revoked, protected user IDs can be activated using the ALTUSER (userid) RESUME command.

A protected user ID cannot be used to enter the system by any method that uses, a supplied password, such as TSO logon, CICS signon, PassTicket authentication, z/OS UNIX rlogin, batch job submission when a password is specified using the PASSWORD parameter of the JOB statement, or by supplying a password phrase. Before assigning the PROTECTED attribute to a user ID, you should ensure that the user ID will not be used in any situation where specification of a password, PassTicket, or password phrase is required.

Applications can authenticate a protected user ID with an Identity Token (IDT) when the covering IDTDATA profile indicates PROTALLOWED(YES). See z/OS Security Server RACF Command Language Reference and z/OS Security Server RACROUTE Macro Reference for more details on the RACF IDT support.

You might want to assign protected user IDs to z/OS UNIX, and to the UNIX daemons, started procedures, applications, servers or subsystems associated with z/OS UNIX, to minimize their exposure to inadvertent or malicious misuse or revocation. Surrogate-submitted batch jobs can use protected user IDs. See “Using protected user IDs for batch jobs” for more information. Protected users can be associated with started procedures defined in the STARTED class (preferred method) or in the started procedures table (ICHRIN03). For more information, see “Assigning RACF user IDs to started procedures”.

The following example shows the ALTUSER command used to assign the PROTECTED attribute to an existing user ID.

```
ALTUSER SERVER8 NOPASSWORD NOPHRASE
```

A protected user ID will have the PROTECTED attribute displayed in the output of the LISTUSER command.

To remove the PROTECTED attribute from an existing user ID, use the ALTUSER command to assign a password:

```
ALTUSER SERVER8 PASSWORD(password)
```

3.2 z/OS Security Server RACF Command Language Reference

This document supplements information for the following RACF commands:

- RDEFINE
- RALTER
- RLIST

3.2.1 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) ]
    [ PROTALLOWED ( YES | NO ) ]
) ]
```

IDTPARMS

IDTPARMS

Specifies information for the IDTDATA class profile being changed.

…

PROTALLOWED (YES | NO)

Specifies whether an Identity Token (IDT) validated with this profile can be used to authenticate a protected user.

3.2.2 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]

[PROTALLOWED (YES | NO)]

)

NOIDTPARMS]

IDTPARMS | NOIDTPARMS

IDTPARMS

Specifies information for the IDTDATA class profile being changed.

…

PROTALLOWED (YES | NO)

Specifies whether an Identity Token (IDT) validated with this profile can be used to authenticate a protected user.

NOIDTPARMS

Deletes the IDTPARMS segment.

3.2.3 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
PROTECTED ALLOWED = NO
```

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 *Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY and initACEE* sections.

3.3.1 RACROUTE REQUEST=VERIFY

The parameters are explained as follows:

…

,PASSCHK=YES
,PASSCHK=NO
,PASSCHK=NOMFA

specifies whether the user's password, password phrase, MFA credentials, ~~or~~ OIDCARD or Identity Token (IDT) is to be verified.

YES

RACROUTE REQUEST=VERIFY verifies the user's password, password phrase, MFA credentials, ~~or~~ OIDCARD or IDT.

There are some circumstances where verification does not occur even though PASSCHK=YES is specified. Some examples are surrogate processing (see z/OS Security Server RACF Security Administrator's Guide) or when the START or the ENVRIN keywords are specified.

A User ID with the protected attribute can be authenticated with PASSCHK=YES with an IDT when the covering IDTDATA profile indicates PROTALLOWED(YES). See Appendix G. Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY and z/OS Security Server RACF Command Language Reference for more details on the RACF IDT support.

For a user subject to multi-factor authentication (MFA), RACF passes the contents of the PASSWRD=, NEWPASS=, PHRASE=, and NEWPHRASE= keywords to the MFA started task, where they are evaluated as MFA credentials. If the credentials are unable to be evaluated as MFA credentials (for example, if the MFA started task is unavailable), they are evaluated as RACF credentials if the user is allowed to fall back to password-based authentication.

NO

The user's password, password phrase, MFA credentials, ~~or~~ OIDCARD or IDT is not verified. And, if the logon is successful, no message is issued.
The IDTA keyword will be ignored and any supplied IDT will be ignored and no IDT will be generated.

NOMFA

Same as YES, except password and password phrase parameters are always verified as a password or password phrase, not as MFA credentials, even for users who have an active MFA factor.

Use of the PASSCHK=NOMFA parameter requires that RELEASE=1.9 or later be specified.

…

,USERID=userid addr

specifies the user identification of the user who has entered the system. The address points to a 1-byte length field, followed by the user ID, which can be up to eight characters.

If the USERID= keyword is omitted, "*" is the default.

When the IDTA is specified with a supplied IDT, the USERID parameter may be omitted. In this case, the "sub" claim value from the IDT is used as the effective USERID.

To prevent a protected user ID from being used to log on, RACROUTE REQUEST=VERIFY processing checks if the protected user ID indicator is on in the user profile. If the indicator is on, RACROUTE REQUEST=VERIFY fails unless keywords such as PASSCHK=NO or START=PROCNAME have been specified to indicate that no password is needed. If a password is expected to be specified for a RACROUTE, it fails as an incorrect password attempt. This denies users the ability to log on with a protected user ID using a password, PassTicket, or OID card. RACROUTE REQUEST=VERIFY processing returns the same RACROUTE return code, informational error message, and auditing as done for a normal system entry attempt with an incorrect password. However, the user profile is not updated, the revoke count is not incremented or reset, and the user is not revoked for exceeding the system limit for password attempts.
A User ID with the protected attribute can be authenticated with PASSCHK=YES with an IDT when the covering IDTDATA profile indicates PROTALLOWED(YES). See Appendix G. Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY and z/OS Security Server RACF Command Language Reference for more details on the RACF IDT support.

…

Appendix G. Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY and initACEE

This section is updated to indicate that initACEE can now also be used to generate an IDT from an ACEE. IDT validation continues to be performed with RACROUTE.

The following is guidance information for application programmers looking to utilize the IDTA parameter in RACROUTE REQUEST=VERIFY and initACEE. The Identity Token (IDT) has many different scenarios for authentication, however the following two are the most useful:

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 behavior 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.

Some applications are invoked in an environment that may not have performed the original user authentication and have a need to authenticate a user to another application interface. The initACEE SAF callable service can be used to generate an IDT for a user, including protected users, when an ACEE security environment already exists. Just like IDTs generated by RACROUTE during user authentication an IDT generated with initACEE from an ACEE can be used to authenticate a user by passing it into RACROUTE REQUEST=VERIFY,ENVIR=CREATE,PASSCHK=YES via the IDTA keyword.

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~~ and initACEE have support for an IDT in the format of a JSON Web Token (JWT). In the future ~~RACROUTE may support~~ other IDT formats may be supported.

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~~ are encoded 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 older 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.
- **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 Details:
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.
4. **“saf-acee”** – The IDT was created from an existing ACEE security environment.

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 when authenticating a user with RACROUTE, 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 and initACEE can create a signed or unsigned IDT. The security administrator may configure profiles in the IDTDATA class to control how ~~RACROUTE creates~~ the IDT is generated 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 an end user, it must turn on the IDTA_End_User_IDT flag. When the IDTA_End_User_IDT is on, RACROUTE and initACEE will not return an unsigned IDT ~~or~~ and RACROUTE will not 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 authentication processing.

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 following signature algorithms are supported:
- ‘HS256’ HMAC with SHA-256
- ‘HS384’ HMAC with SHA-384
- ‘HS512’ HMAC with SHA-512

Encrypted JWTs are not supported.

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 through 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 timestamp, 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 generate a new IDT to pass to RACROUTE authentication processing. ~~re-prompt the user for authentication credentials before calling RACROUTE again.~~

When RACROUTE and initACEE create 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 but the normal password change rules apply. When the IDT contains an “amr” claim value of “saf-pwd” it indicates that the user has authenticated with a password and new password may be specified. When the IDT contains an “amr” claim value of “saf-phr” it indicates the user has authenticated with a password phrase and a new password phrase may be specified. When the IDT contains an “amr” claim value of “saf-ptkt” it indicates the user has authenticated with a PassTicket and new password or new password phrase may be specified.

Protected Users:
The initACEE callable service can be used to generate an IDT for a protected user. A protected user can be authenticated with an IDT with RACROUTE REQ=VERIFY,ENVIR=CREATE,PASSCHK=YES when the covering IDTDATA profile indicates that protected users are allowed via the PROTALLOWED(YES) keyword in the IDTPARMS segment.
An IDT generated by initACEE will contain an AMR claim value of “saf-acee” which indicates the IDT was generated from an existing security environment. In order to authenticate a user with an IDT with the “saf-acee” AMR claim with RACROUTE REQ=VERIFY the target system must have the required support installed.

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 is not processed. The IDTDATA class must be active and RACLISTed before any profiles are used for the generation or validation of an IDT.

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

When there is no covering IDTDATA class profile, ~~RACROUTE will generate~~ an IDT will be generated 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 request will fail and the required size will be set in the IDTA_IDT_Len Identity Token Length field and the following error return codes will be set: • RACROUTE: 8/70/1 • initACEE: 8/12/5
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 is generated, the IDT length will be set on output. In some RACROUTE use cases, 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 will be set to the required size and fail with the fol-lowing return codes: • RACROUTE: 8/70/1 • initACEE: 8/12/5 The caller must reallocate the buffer with the larger size and reset to the IDTA_IDT_Len to zero or the length of a supplied IDT. When a there is an error processing a specified IDT ~~RACROUTE will set~~ the IDTA_IDT_Len will be set 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 gen-erate 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** – End User – 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 will not be returned and instead ~~set the~~ **IDTA_IDT_Gen_RC** will be set to **IDTA_IDT_GEN_RC_UNSIGN**. When this bit is set on during Identity Token validation, RA-CROUTE 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 ~~RACINIT~~ Call with an IDTA:

When an application initially calls RACROUTE REQEST=VERIFY or initACEE 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 to RACROUTE, 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 or initACEE.

Other fields: Initialize all other fields to binary zero.

Subsequent ~~RACINIT~~ RACROUTE Calls with an IDTA:
In some authentication scenarios, an application may need to call RACROUTE REQUEST=VERIFY multiple times. 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 when Validating an IDT with RACROUTE:
~~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.

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.

Error Handling when Generating an IDT:
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.

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.

3.4 z/OS Security Server RACF Callable Services

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

- The *initACEE* section is updated to add a new function code.
- The *R_admin* appendix is updated to add new field to the IDTPARMS segment.

3.4.1 initACEE (IRRSIA00): Initialize ACEE

Function

The initACEE service provides an interface for identity related functions:

1) Creating and managing RACF security contexts through the z/OS UNIX System Services pthread_security_np service, __login service, or by other MVS server address spaces that do not use z/OS UNIX services.
2) Registering and deregistering certificates through the z/OS UNIX System Services __security service.
3) Querying a certificate to determine if it is associated with a user ID.
4) Generating an Identity Token (IDT) from an ACEE security environment.

When initACEE calls RACINIT to create an ACEE for an MFA user, a new bit in the ACEE will be set to indicate that the user was authenticated with MFA. InitACEE will not copy an MFA authenticated ACEE in the managed cache. This will cause initACEE to call RACINIT and perform MFA authentication on subsequent initACEE calls for the same user. RACINIT may still use its VLF cache for MFA authenticated users when called by initACEE.

…

Format

```
CALL IRRSIA00 (Work_area,
               ALET, SAF_return_code,
               ALET, RACF_return_code,
               ALET, RACF_reason_code,
                     Function_code,
                     Attributes,
                     RACF_userid,
                     ACEE_ptr,
                     APPL_id,
                     Password,
                     Logstring,
                     Certificate,
                     ENVR_in,
                     ENVR_out,
                     Output_area,
                     X500 name,
                     Variable_list,
                     Security_label,
                     SERVAUTH_name,
                     Password_phrase,
                     IDID_area,
                     ACEE2_ALET,
                     ACEE2_ptr,
                     IDTA
                     )
```

…

Parameters

…

Function_code

The name of a 1-byte area containing the function code.

X'01'
Create an ACEE.

X'02'
Delete an ACEE.

X'03'
Purge all managed ACEEs.

X'04'
Register a certificate

X'05'
Deregister a certificate

X'06'
Query a certificate

X'07'
Generate an IDT from an ACEE

…

ACEE_ptr

The name of a 4-byte area that contains the ACEE address.
This parameter is not supported for function code X’07’ and ACEE2_PTR should be used instead.

APPL_id

The name of a 9-byte area that consists of a 1-byte length field followed by the name of the application to be used if verifying the user's authority to access the application. This saves the application from having to do a separate authorization check.
When using certificate mapping profiles, the application name is also used as part of the additional criteria in determining a user ID when a certificate is passed to initACEE.
When using the Generate an IDT from ACEE function the application name is used as one of the qualifiers to locate a profile in the IDTDATA class. For the Generate an IDT from ACEE function, the APPL_id parameter must be specified with a length between 1-8.
It must be specified in uppercase. If not specified, the length must equal zero.

…

ACEE2_ALET

The name of a 4-byte area containing the ALET for the ACEE pointed to named by the ACEE2_ptr parameter. The 4-byte area must be in the primary address space.
This parameter is only supported by function code X'07'.

ACEE2_PTR

The name of a 4-byte area that contains the ACEE address. This parameter may be qualified by the ALET specified in ACEE2_ALET.
This parameter is only supported by function code X'07'.

IDTA

The name of a fullword containing the address of an Identity Token Area for the generation of an Identity Token for the specified ACEE.
The IDT is returned in the buffer pointed to by the IDTA buffer pointer. The IDT length is set in the IDTA IDT length field. More IDTA parameter details are described in the z/OS Security Server RACROUTE Macro Reference publication in Appendix G.
The IDTA is mapped by SAF macro IRRPIDTA.
An IDT generated by initACEE will contain an AMR claim value of "saf-acee" which indicates the IDT was generated from an existing security environment. In order to authenticate a user with an IDT with the "saf-acee" AMR claim with RACROUTE REQ=VERIFY the target system must have the required support installed.

Table 6. Parameter usage

Parameter	**…**	**Generate IDT**
SAF_return_code	…	Output
RACF_return_code	…	Output
RACF_reason_code	…	Output
Function_code	…	Input
Attributes	…	n/a
RACF_userid	…	n/a
ACEE_ptr	…	n/a
APPL_id	…	Input
Password	…	n/a
Logstring	…	n/a
Certificate	…	n/a
ENVR_in	…	n/a
ENVR_out	…	n/a
Output_area	…	n/a
X500name	…	n/a
Variable_list	…	n/a
Security_label	…	n/a
SERVAUTH_name	…	n/a
Password_phrase	…	n/a
IDID_area	…	n/a
ACEE2_ALET	n/a	Input
ACEE2_ptr	n/a	Input

IDTA	n/a	Input/Ouput

…

Return and reason codes
IRRSIA00 returns the following values in the reason and return code parameters:
…

Table 17. initACEE Generate IDT return codes

SAF return code	RACF return code	RACF reason code	Explanation
0	0	0	The service was successful.
4	0	0	RACF is not installed.
8	8	4	Parameter list error occurred.
8	8	8	An internal error occurred during RACF processing.
8	8	12	Recovery environment could not be established.
8	8	16	IDTDATA class not active.
8	12	1	IDTA eyecatcher not “IDTA”.
8	12	2	IDTA version not valid.
8	12	3	IDTA length not valid.
8	12	4	IDTA IDT buffer pointer not valid.
8	12	5	IDT buffer length insufficient. The IDT length field has been updated to the required size.
8	12	6	IDTA IDT length is nonzero.
8	12	7	IDTA IDT type not valid.
8	16	3	Signed IDTs are required but a key is not configured.
8	16	4	ICSF is not available to generate signature.
8	16	5	ICSF error detected attempting to generate signature.

Usage notes

…

6. Audit records are written only in the following situations:
   a. An ACEE is to be created and a password has been specified that is not the user's current password, or a password phrase has been specified that is not the user's current password phrase.
   b. An ACEE is to be created and a PassTicket has been specified that does not evaluate.
   c. An ACEE is to be created and the user ID has been revoked.
   d. A certificate is to be registered, and the user is not authorized to the FACILITY class resource IRR.DIGTCERT.ADD.
   e. A certificate is to be deregistered and the user is not authorized to the FACILITY

class resource IRR.DIGTCERT.DELETE.

f. A certificate is successfully registered or deregistered, and SETROPTS AUDIT(USER) is in effect, or UAUDIT is in effect for the user, or the user has SPECIAL authority and SETROPTS SAUDIT is in effect.

g. An ACEE is to be created and a certificate has been specified that does not correspond to a RACF user ID.

h. An ACEE is to be created and a certificate has been specified that is not trusted.

i. An Identity Token is successfully generated, and SETROPTS AUDIT(USER) is in effect, or UAUDIT is in effect for the user, or the user has SPECIAL authority and SETROPTS SAUDIT is in effect.

j. An attempt to generate an Identity Token is unsuccessful with one of the 8/16/x return code combinations.

…

45. A parameter list error occurs for function codes X'01', X'02', X'03', or X'06’:

- When the ACEE2_ALET parameter is specified with non-zero value
- When the ACEE2_PTR parameter is specified with a non-zero address
- When the IDTA parameter is specified with a non-zero address

47. For function code X'07':

- When any parameters listed in table 6 as “n/a” are provided with non-zero lengths or values an 8/8/4 parameter list error is returned.
- When an IDTA field is not valid, one of the 8/12/x return codes is returned.
- More IDTA parameter details are described in the z/OS Security Server RACROUTE Macro Reference publication in Appendix G.

3.4.2 R_Admin (IRRSEQ00): RACF administration API

The R_admin reference appendix is updated:

· The table *IDTPARMS fields* is updated to add a new field.

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	
IDTPROTA	idtprota	'Y'	IDTPARMS (PROTALLOWED(YES))	Yes	Yes	Yes
		'N'	IDTPARMS (PROTALLOWED(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

Record formats produced by the database unload utility

The general resource IDTPARMS definition record (05K0) is updated to add new fields.

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”.
GRIDTP_PROTALLOWED	Char	358	361	Is the IDT allowed to authenticate a protected user? Valid values include “Yes” and “No”.

3.5.2 SMF records

Table of event codes and event code qualifiers
The "Table of event codes and event code qualifiers" is updated to add new event code qualifiers for event 67 and a new relocate for the initACEE event.

Event	Command	Code Qualifier	Description	Relocate type sections (possible SMF80DTP/SMF80DA2 values)
…				
67(43)	INITACEE	…		49, 53, 318, 319, 331, 332 374, 386, 392, 393, 394, 395, 424,425, 446, 449
		10(A)	No RACF user ID found for distributed identity	
		11(B)	Successfully generated IDT from ACEE	
		12(C)	Failed attempting to generate IDT from ACEE	
…				

Table of extended-length relocate section variable data:

The "Table of extended-length relocate section variable data" is updated:

- Relocate 443 is updated to add a new bit to indicate that authentication is from an Identity Token created from an existing security environment.
- Relocate 449 is added.

Data type (SMF80TP2) dec(hex)	Data length (SMF80DL 2)	Format	Audited by event code	Description (SMF80DA2)
443(1BB)	variable	mixed	1	Byte 1: Authentication information:
				Bit / **Meaning when set**
				0 / Authenticated from VLF
				1 / User has active MFA factor(s)
				2 / MFA user allowed to fall back

	when no MFA decision can be made
3	No MFA decision for MFA user
4	IBMMFA requested that RACROUTE REQUEST=VERIFY return the password-expired return code.
5	IBM MFA requested that RACROUTE REQUEST=VERIFY return the new-password-invalid return code.
6	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).
7	Relocate 443 is extended.

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	Legacy PassTicket Evaluated
6	Legacy PassTicket Successful
7	Enhanced PassTicket Type UPPER Evaluated

Byte 20: Flag byte 4: Authentication Details

Bit	Meaning when set
0	Enhanced PassTicket Type UPPER Successful
1	Enhanced PassTicket Type MIXED Evaluated
2	Enhanced PassTicket Type MIXED Successful
3	IDT from existing security environment.
4-7	Reserved

Bytes 21-28: Derived Application Name

Byte 29-32: IDT Validation Reason Code

Byte 33-36: IDT Generation Reason Code

				Byte 37-40: Failing Service ID Byte 41-44: Failing Service Return Code Byte 45-48: Failing Service Reason Code
…				
449 (1C1)	Variable	Mixed	67	Byte 1-8: User ID Byte 9-16: Application name Byte 17-20: IDT Build Reason Code Byte 21-24: Failing Service Identifier Byte 25-28: Failing Service Return Code Byte 29-32: Failing Service Reason Code

3.5.3 The format of the unloaded SMF type 80 data

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_RELO443_EXTENDED	Yes/ No	4575	4578	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	Authentication reason, part 1. Expressed as hexadecimal number.
INIT_AUTH_RSN2	Char	4629	4636	Authentication reason, part 2. Expressed as hexadecimal number.
INIT_AUTH_RSN3	Char	4638	4645	Authentication reason, part 3. Expressed as hexadecimal number.

INIT_AUTH_RSN4	Char	4647	4654	Authentication reason, part 4. Expressed as hexadecimal number.
INIT_PWD_PHR_EXPIRED	Yes/ No	4656	4656	The supplied password or password phrase was expired.
INIT_NPWD_NPHR_NONVAL	Yes/ No	4661	4661	The supplied new password or new password phrase was not valid.
INIT_IDT_EVAL	Yes/ No	4666	4666	The supplied Identity Token (IDT) was evaluated.
INIT_IDT_SUCC	Yes/ No	4671	4671	The supplied Identity Token (IDT) was evaluated successfully.
INIT_MFA_REAUTHENT	Yes/ No	4676	4676	IBM MFA requested that RACROUTE REQUEST=VERIFY return the password-invalid return code, but not to increment the password revoke count (reauthentication requested).
INIT_LPT_EVAL	Yes/ No	4681	4684	The supplied Password was evaluated as a legacy PassTicket.
INIT_LPT_SUCC	Yes/ No	4686	4689	The supplied Password was evaluated successfully as a legacy PassTicket.
INIT_EPT_UPPER_EVAL	Yes/ No	4691	4694	The supplied Password was evaluated as an enhanced PassTicket type UPPER.
INIT_EPT_UPPER_SUCC	Yes/ No	4696	4699	The supplied Password was evaluated successfully as an enhanced PassTicket type UPPER.
INIT_EPT_MIXED_EVAL	Yes/ No	4701	4704	The supplied Password was evaluated as an enhanced PassTicket type MIXED.
INIT_EPT_MIXED_SUCC	Yes/ No	4706	4709	The supplied Password was evaluated successfully as an enhanced PassTicket type MIXED.
~~INIT_RESERVED_07~~ INIT_IDT_FROM_SEC_ENV	Yes/ No	4711	4714	~~Reserved for IBM's use~~ IDT from existing security environment.
INIT_RESERVED_08	Yes/ No	4716	4716	Reserved for IBM's use
INIT_RESERVED_09	Yes/ No	4721	4721	Reserved for IBM's use
INIT_RESERVED_10	Yes/ No	4726	4726	Reserved for IBM's use

INIT_RESERVED_11	Yes/ No	4731	4731	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_BUILD_RSNC	Char	4754	4761	IDT Build 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

The InitACEE record extension
The initACEE record extension is updated to add a new fields for the generate IDT function from relocate 449.

Format of the InitACEE record extension (event code 67)

Field Name	**Type**	**Len**	**Start**	**End**	**Comments**
INTA_USER_NAME	Char	20	282	301	The name associated with the user ID.
…	…	…	…	…	…
INTA_CERT_FGRPRNT	Char	64	4496	4559	Certificate SHA64 fingerprint in printable hex
INTA_IDT_USER	Char	8	4561	4568	User ID from specified ACEE for generate IDT function
INTA_APPL	Char	8	4570	4577	Application name specified to initACEE for generate IDT function
INTA_IDT_BUILD_RSNC	Char	8	4579	4586	IDT Build Reason Code
INTA_SERVICE_CODE	Char	8	4588	4595	Failing Service Identifier
INTA_SERVICE_RC	Char	8	4597	4604	Failing Service Return Code
INTA_SERVICE_RSNC	Char	8	4606	4613	Failing Service Reason Code

The “Event qualifiers for InitACEE records” table is updated to add two new event code qualifiers.

Event qualifiers for InitACEE records

Event qualifier	**Event qualifier number**	**Event description**
SUCCSREG	00	Successful certificate registration.
…	…	…

DIDNOTDF	10	No RACF user ID found for distributed identity.
SUCCSIDT	11	Successful IDT generated from ACEE.
FAILIDT	12	Failed attempting to generate IDT from ACEE.

3.5.4 Appendix Event code qualifier descriptions

The Appendix “Event code qualifier descriptions” is updated to add two new event code qualifiers for the initACEE event.

…
67(43)
initACEE
…
10(A)
No RACF userID found for distributed identity
11(B)
Successful IDT generated from ACEE
12(C)
Failed attempting to generate IDT from ACEE

3.5.5 RACF database templates

The IDTPARMS segment in the GENERAL section is updated to add a new field.

```
$/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
IDTPROTA  008 00 00 00000001 80 IDTPARMS - IDT Protected allowed
```

3.6 z/OS Security Server RACF Data Areas

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

3.6.1 COMP: Common SAF/RACF Parameter List for z/OS UNIX System Services

The “INTA” section of the “Structure COMP” table is updated to add new parameters for the initACEE callable service.

Offset (dec)	**Offset (Hex)**	**Type**	**Len**	**Name(Dim)**	**Description**
...					
64	40	ADDRESS	4	INTA_IDID_AREA@	Address of a fullword containing the address of a variable length area containing distributed identity data (IDID)
		....		INTA_LAST_PARM_IDID	Variable length parameter. This could be the last parameter.
68	44		4	INTA_ACEE2_ALET@	Address of ALET for ACEE 2 parameter.
72	48		4	INTA_ACEE2P@	Address of a full word input area for the ACEE 2 address.
76	4C		4	INTA_IDTA@	Address of a fullword containing the address of a variable length area containing identity token area (IDTA).
		....		INTA_LAST_PARM_IDTA	Variable length parameter list. This could be the last parameter.
		....		INTA_LAST_PARM	Variable length parameter list. This is the last parameter.

The “INTA function code values” section of the "Constants for COMP” table is updated to add the new INTA_GENIDT function code.

Len	Type	Value	Name	Description
...				
1	DECIMAL	6	INTA_QUERY	Query a certificate for an associated user ID
1	DECIMAL	7	INTA_GENIDT	Generate an IDT from an ACEE
...				

3.6.2 RCVT: RACF Communication Vector Table

The RACF Communication Vector Table adds a field to indicate that the IDT V2 Functions are available. Other products can check this bit to determine if the current version of RACF has IDT V2 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..			RCVTIDT2	Identity Token 2 Functions (OA63462) are available.
...					

4 Trademarks