# Documentation Updates for APAR PH55271

# Updates for XL C/C++ Runtime Library Reference

This document contains updates to the information in XL C/C++ Runtime Library Reference (SC14-7314-XX).

## Chapter 2. Header files

### unistd.h — Implementation-specific functions

...

The unistd.h header file declares a number of implementation-specific functions:

| | | | |
|---|---|---|---|
| __atoe() | __atoe_l() | __authenticate() | __check_resource_auth_np() |
| __convert_id_np() | __etoa() | __etoa_l() | __isPosixOn() |
| __smf_record() | __wsinit() | | |

## Chapter 3. Library functions

### atoll() — Convert character string to signed long long

...

### __authenticate() — Authenticate the specified user's credentials

| Standards / Extensions | C or C++ | Dependencies |
|---|---|---|
| z/OS® UNIX | both | |

Format

```
#include <unistd.h>

int __authenticate(unsigned int Auth_cred_type,
                   int *User_name_length,
                   char *User_name, int Pass_length,
                   char *Pass, int New_pass_length,
                   char *New_pass, int *Idt_buffer_length,
                   char *Idt_buffer_ptr, int *Idt_length,
                   char **Msg_buffer_ptr,
                   int Appl_id_length, char *Appl_id,
                   unsigned int *Option_flags);
```

### General description

The __authenticate() function can authenticate a user using passwords, password phrases, PassTickets or Identity Tokens (IDTs). It can also optionally generate an IDT to be used for follow on authentications. The __authenticate() function only authenticates the user's credentials, it does not create a security context (ACEE) or modify the caller's process or thread identity.

### Auth_cred_type

The parameter identifies the type of credentials to be used to perform the authentication. It can be specified to the following values:

#### AUTH_USER_ID

The User_name parameter specified by the caller will be passed to RACF for the authentication.

#### AUTH_ID_TOKEN

The IDT from the location specified by the Idt_buffer_ptr parameter will be passed to RACF for authentication.

**Note:** Both AUTH_USER_ID and AUTH_ID_TOKEN can be specified. When both are specified, both the userid and IDT will be passed to RACF. If the userid associated with the IDT does not match the User_name parameter, the authentication will fail.

User_name_length

When AUTH_USER_ID is specified for the Auth_cred_type parameter, it specifies the length of the User_name parameter string.

Returned length of the User_name parameter string returned when authenticating with only an IDT. The name is obtained from a temporary ACEE created by RACF. The caller must specify the maximum userid length of 8 character to accommodate all possible userid sizes. The authenticate() service will modify the User_name_length parameter specified by the caller to indicate the length of the returned User_name parameter string.

User_name

Supplied userid when Auth_cred_type specifies AUTH_USER_ID.

Returned userid when Auth_cred_type specifies AUTH_ID_TOKEN and AUTH_USER_ID is not specified.

This parameter is an input/output parameter, the storage must be writable.

Pass_length

The parameter contains the length of the Pass parameter. This length must be between 1 and 8 characters for a password or PassTicket or between 9 and 100 characters for a password phrase. A length of zero indicates that Pass is to be ignored.

Pass

The parameter contains left-justified, the password, PassTicket or password phrase that is to be verified.

New_pass_length

The parameter contains the length of New_pass. This length must be between 1 and 8 characters for a password or between 9 and 100 characters for a password phrase. A length of zero indicates that New_pass is to be ignored.

New_pass

The parameter contains left-justified, the new password or password phrase.

Idt_buffer_length

The parameter contains supplied length of the buffer pointed to by Idt_buffer_ptr. If the supplied buffer is smaller than the IDT built by RACF, the syscall fails with -1, EINVAL, JrBuffTooSmall and the required length is returned in the Idt_length parameter. In this situation, the caller should allocate a larger buffer and retry the __authenticate() call.

### Idt_buffer_ptr

The parameter contains supplied address of the buffer that can be used for input or output of an IDT. When AUTH_ID_TOKEN is specified for the Auth_cred_type parameter, the buffer contain an IDT that the __authenticate() service will pass to RACF to be authenticated. When only AUTH_USER_ID is specified for the Auth_cred_type parameter and AUTH_BUILD_IDT is specified for the Option_flags parameter, If an IDT is successfully generated, the __authenticate() service will copy the newly generated IDT into the buffer. In both cases, if an IDT is copied into the buffer the AUTH_RETURNED_IDT flag in the Option_Flags parameter will be set by __authenticate() and returned to the caller.

### Idt_length

This parameter contains the address of IDT length.

When AUTH_ID_TOKEN is specified for the Auth_cred_type parameter, this parameter should supply length of the IDT whose location is specified by the Idt_buffer_ptr parameter. When AUTH_USER_ID is specified for the Auth_cred_type parameter and AUTH_BUILD_IDT is specified for the Option_flags parameter, the length must be zero.

When AUTH_ID_TOKEN is specified for the Auth_cred_type parameter and RACF refreshes the IDT while authenticating the IDT supplied by the caller, this parameter returns length of the IDT copied into the location specified by the Idt_buffer_ptr parameter. When AUTH_USER_ID is specified for the Auth_cred_type parameter and AUTH_BUILD_IDT is specified for the Option_flags parameter. The size of the newly created IDT generated by RACF will be returned.

The Idt_length parameter can be both supplied and returned for the same __authenticate() call depending on the conditions specified above being met or not.

### Msg_buffer_ptr

The parameter contains the address of a pointer of message buffer if any messages were returned by RACF. All message buffers for one __authenticate() call will be located in a

contiguous piece of storage and the caller is responsible for freeing the buffer storage using free(). The AUTH_MSGRTRN option in the Option_Flags parameter must be specified to have messages returned.

### Appl_id_length

The parameter contains the length of the Appl_id parameter. This length must be between 1 and 8 characters. A length of zero indicates the Appl_id parameter is to be ignored.

### Appl_id

The parameter contains left-justified, the APPLID that identifies the name of the application requesting authentication. If an Appl_id is not specific (Appl_id_length of zero) the application id will default to OMVSAPPL.

### Option_flags

The parameter contains the __authenticate() options. If no options are required, specify 0 for this paramter.

Valid values for this field include the following:

#### AUTH_BUILD_IDT

Request that an ID Token is built by RACF and returned to the caller. This option is valid when only AUTH_USER_ID is specified (AUTH_ID_TOKEN is not specified) for the Auth_cred_type parameter. The newly built ID Token is returned to the caller in the buffer pointed to by the Idt_buffer_ptr parameter and its length returned in the Idt_length parameter.

#### AUTH_RETURN_USERNAME

Request that the userid associated with the ID Token used for authentication be returned in the field specified by the User_name parameter. This option is valid when only AUTH_ID_TOKEN is specified (AUTH_USER_ID is not specified) for the Auth_cred_type parameter and the User_name_length parameter contains a value of 8. If those conditions are not met, the request will fail. The length of the returned User_name will be returned in the User_name_length parameter.

#### AUTH_MSGRTRN

This option controls the MSGRTRN parameter specified on the RACROUTE REQUEST=VERIFY call made by __authenticate().

When this option is not specified, MSGRTRN=NO (default) is specified for the resulting RACROUTE REQUEST=VERIFY. Messages will not be returned in a buffer and will instead be issued by RACF using TPUT.

When this option is specified, MSGRTRN=YES is specified for RACROUTE REQUEST=VERIFY. The address of the message buffer obtained by RACF will be returned in the Msg_buffer_ptr parameter.

AUTH_ RETURNED_IDT

This option is returned by the __authenticate() to indicate an IDT has been copied into the location specified by the Idt_buffer_ptr parameter. The length of that IDT is returned in the Idt_length parameter. This can occur when the caller request to build a new IDT or is authenticating with an IDT and RACF refreshes the IDT in response to changes of system or user settings. This option should not be specified by the caller and will cause the system call to fail with -1/EINVAL/JrBadOptnFlags.

## Returned value

If successful, __authenticate() returns a value of zero. If unsuccessful, it returns a value of -1 and sets errno to one of the following values:

EACCES

Permission is denied.

EINVAL

The parameter is incorrect.

ESRCH

No such process or thread exists.

EMVSSAFEXTRERR

SAF/RACF extract error.

EMVSSAF2ERR

SAF/RACF error.

EMVSEXPIRE

The password for the specified resource has expired.

EMVSPASSWORD

The new password or password phrase specified is not valid.

## Usage notes

1. If a profile is defined in the FACILITY class protecting the BPX.DAEMON resource, all programs that are loaded into the caller's address space must be controlled programs by the installed security product (such as RACF). If the __authenticate service detects that a load of a non-program control program was done, it fails with an errno of EMVSERR and an errnojr of JRENVDIRTY. For more information, see Establishing the correct level of security for daemons in *z/OS UNIX System Services Planning*.
2. To request RACF to return messages in a buffer the AUTH_MSGRTRN option in the Option_flags parameter must be specified in conjunction with Msg_buffer_ptr paramerter. If the AUTH_MSGRTRN option is not specified the Msg_buffer_ptr parameter is ignored and the __authenticate() service will not request RACF to return messages resulting from the request. When not returning messages RACF will issue the messages using TPUT. For more information about messages returned by RACF see the MSGSP and MSGRTRN parameters in Chapter 2., RACROUTE (stand form) in the *z/OS Security Server RACROUTE Macro Reference*
3. When a message buffer address is returned to the caller in the Msg_buff_ptr parameter, the caller is responsible to free the storage it points to. The format of the buffer is defined by RACF. The area consists of two fullwords followed by the message itself in write-to-operator (WTO) parameter list format. The first word is the length of the area including the two-fullword header; the second word points to the next message area, if there is one, or contains zero if no more message areas exist. See z/OS Security Server RACROUTE Macro Reference, Chapter 2., the MSGRTRN and MSGSP parameters for more information about messages returned by RACF.
4. The current minimum size of an IDT is 1024 bytes. This may change in the future as RACF adds support for more types of IDTs. See *z/OS Security Server RACROUTE Macro Reference, Appendix G. Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY* for more information about IDTs.
5. The __authenticate service only accepts and returns signed IDTs. See *z/OS Security Server RACROUTE Macro Reference, Appendix G. Activating and using the IDTA parameter in RACROUTE REQUEST=VERIFY* for more information about signed and unsigned IDTs.

6. The Idt_buffer_length and Idt_buffer_ptr parameters detail the length and location of the buffer supplied by the caller for storing IDTs. The Idt_length parameter details the length of an IDT located in the IDT buffer. The caller specifies the Idt_length when supplying an IDT for authentication. The system returns the Idt_length when either building a new IDT or for an IDT that has been refreshed by RACF when authenticating with an IDT. When the system returns a new or refreshed IDT the AUTH_RETURNED_IDT flag in the Option_Flags parameter will be set and returned to the caller to indicate a newly created or refreshed IDT has been returned.

## Related information

- “unistd.h — Implementation-specific functions”