

**NAME**

Heimdal Kerberos 5 keytab handling functions -

**Functions**

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_register** (krb5\_context context, const krb5\_kt\_ops \*ops)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_resolve** (krb5\_context context, const char \*name, krb5\_keytab \*id)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_default\_name** (krb5\_context context, char \*name, size\_t namesize)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_default\_modify\_name** (krb5\_context context, char \*name, size\_t namesize)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_default** (krb5\_context context, krb5\_keytab \*id)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_read\_service\_key** (krb5\_context context, krb5\_pointer keyprocarg, krb5\_principal principal, krb5\_kvno vno, krb5\_etype etype, krb5\_keyblock \*\*key)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_get\_type** (krb5\_context context, krb5\_keytab keytab, char \*prefix, size\_t prefixsize)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_get\_name** (krb5\_context context, krb5\_keytab keytab, char \*name, size\_t namesize)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_get\_full\_name** (krb5\_context context, krb5\_keytab keytab, char \*\*str)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_close** (krb5\_context context, krb5\_keytab id)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_destroy** (krb5\_context context, krb5\_keytab id)

KRB5\_LIB\_FUNCTION krb5\_boolean KRB5\_LIB\_CALL **krb5\_kt\_compare** (krb5\_context context, krb5\_keytab\_entry \*entry, krb5\_const\_principal principal, krb5\_kvno vno, krb5\_etype etype)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_get\_entry** (krb5\_context context, krb5\_keytab id, krb5\_const\_principal principal, krb5\_kvno kvno, krb5\_etype etype, krb5\_keytab\_entry \*entry)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_copy\_entry\_contents** (krb5\_context context, const krb5\_keytab\_entry \*in, krb5\_keytab\_entry \*out)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_free\_entry** (krb5\_context context, krb5\_keytab\_entry \*entry)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_start\_seq\_get** (krb5\_context context, krb5\_keytab id, krb5\_kt\_cursor \*cursor)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_next\_entry** (krb5\_context context, krb5\_keytab id, krb5\_keytab\_entry \*entry, krb5\_kt\_cursor \*cursor)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_end\_seq\_get** (krb5\_context context, krb5\_keytab id, krb5\_kt\_cursor \*cursor)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_add\_entry** (krb5\_context context, krb5\_keytab id, krb5\_keytab\_entry \*entry)

KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL **krb5\_kt\_remove\_entry** (krb5\_context context, krb5\_keytab id, krb5\_keytab\_entry \*entry)

KRB5\_LIB\_FUNCTION krb5\_boolean KRB5\_LIB\_CALL **krb5\_kt\_have\_content** (krb5\_context context, krb5\_keytab id)

## Detailed Description

### Function Documentation

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_add\_entry** (krb5\_context context, krb5\_keytab id, krb5\_keytab\_entry \* entry)

Add the entry in 'entry' to the keytab 'id'.

#### Parameters:

*context* a Keberos context.

*id* a keytab.

*entry* the entry to add

#### Returns:

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_close** (krb5\_context context, krb5\_keytab id)

Finish using the keytab in 'id'. All resources will be released, even on errors.

#### Parameters:

*context* a Keberos context.

*id* keytab to close.

#### Returns:

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_boolean KRB5\_LIB\_CALL krb5\_kt\_compare** (krb5\_context context, krb5\_keytab\_entry \* entry, krb5\_const\_principal principal, krb5\_kvno vno, krb5\_etype enctype)

Compare 'entry' against 'principal, vno, enctype'. Any of 'principal, vno, enctype' might be 0 which acts as a wildcard. Return TRUE if they compare the same, FALSE otherwise.

#### Parameters:

*context* a Keberos context.  
*entry* an entry to match with.  
*principal* principal to match, NULL matches all principals.  
*vno* key version to match, 0 matches all key version numbers.  
*enctype* encryption type to match, 0 matches all encryption types.

**Returns:**

Return TRUE or match, FALSE if not matched.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_copy\_entry\_contents**  
**(krb5\_context context, const krb5\_keytab\_entry \* in, krb5\_keytab\_entry \* out)**  
 Copy the contents of 'in' into 'out'.

**Parameters:**

*context* a Keberos context.  
*in* the keytab entry to copy.  
*out* the copy of the keytab entry, free with **krb5\_kt\_free\_entry()**.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_default** (**krb5\_context context,**  
**krb5\_keytab \* id**)  
 Set 'id' to the default keytab.

**Parameters:**

*context* a Keberos context.  
*id* the new default keytab.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_default\_modify\_name**  
**(krb5\_context context, char \* name, size\_t namesize)**  
 Copy the name of the default modify keytab into 'name'.

**Parameters:**

*context* a Keberos context.  
*name* buffer where the name will be written  
*namesize* length of name

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_default\_name (krb5\_context context, char \* name, size\_t namesize)**

copy the name of the default keytab into 'name'.

**Parameters:**

*context* a Keberos context.

*name* buffer where the name will be written

*namesize* length of name

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_destroy (krb5\_context context, krb5\_keytab id)**

Destroy (remove) the keytab in 'id'. All resources will be released, even on errors, does the equivalent of `krb5_kt_close()` on the resources.

**Parameters:**

*context* a Keberos context.

*id* keytab to destroy.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_end\_seq\_get (krb5\_context context, krb5\_keytab id, krb5\_kt\_cursor \* cursor)**

Release all resources associated with 'cursor'.

**Parameters:**

*context* a Keberos context.

*id* a keytab.

*cursor* the cursor to free.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_free\_entry (krb5\_context context,**

**krb5\_keytab\_entry \* entry)**

Free the contents of 'entry'.

**Parameters:**

*context* a Keberos context.

*entry* the entry to free

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_get\_entry (krb5\_context context, krb5\_keytab id, krb5\_const\_principal principal, krb5\_kvno kvno, krb5\_etype enctype, krb5\_keytab\_entry \* entry)**

Retrieve the keytab entry for 'principal, kvno, enctype' into 'entry' from the keytab 'id'. Matching is done like `krb5_kt_compare()`.

**Parameters:**

*context* a Keberos context.

*id* a keytab.

*principal* principal to match, NULL matches all principals.

*kvno* key version to match, 0 matches all key version numbers.

*enctype* encryption type to match, 0 matches all encryption types.

*entry* the returned entry, free with `krb5_kt_free_entry()`.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_get\_full\_name (krb5\_context context, krb5\_keytab keytab, char \*\* str)**

Retrieve the full name of the keytab 'keytab' and store the name in 'str'.

**Parameters:**

*context* a Keberos context.

*keytab* keytab to get name for.

*str* the name of the keytab name, use `krb5_xfree()` to free the string. On error, \*str is set to NULL.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_get\_name (krb5\_context context, krb5\_keytab keytab, char \* name, size\_t namesize)**

Retrieve the name of the keytab 'keytab' into 'name', 'namesize'

**Parameters:**

*context* a Keberos context.

*keytab* the keytab to get the name for.

*name* name buffer.

*namesize* size of name buffer.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_get\_type (krb5\_context context, krb5\_keytab keytab, char \* prefix, size\_t prefixsize)**

Return the type of the 'keytab' in the string 'prefix of length 'prefixsize'.

**Parameters:**

*context* a Keberos context.

*keytab* the keytab to get the prefix for

*prefix* prefix buffer

*prefixsize* length of prefix buffer

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_boolean KRB5\_LIB\_CALL krb5\_kt\_have\_content (krb5\_context context, krb5\_keytab id)**

Return true if the keytab exists and have entries

**Parameters:**

*context* a Keberos context.

*id* a keytab.

**Returns:**

Return an error code or 0, see `krb5_get_error_message()`.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_next\_entry (krb5\_context context, krb5\_keytab id, krb5\_keytab\_entry \* entry, krb5\_kt\_cursor \* cursor)**

Get the next entry from keytab, advance the cursor. On last entry the function will return

KRB5\_KT\_END.

**Parameters:**

*context* a Keberos context.

*id* a keytab.

*entry* the returned entry, free with **krb5\_kt\_free\_entry()**.

*cursor* the cursor of the iteration.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_read\_service\_key (krb5\_context context, krb5\_pointer keyprocarg, krb5\_principal principal, krb5\_kvno vno, krb5\_etype etype, krb5\_keyblock \*\* key)**

Read the key identified by '(principal, vno, etype)' from the keytab in 'keyprocarg' (the default if == NULL) into '\*key'.

**Parameters:**

*context* a Keberos context.

*keyprocarg*

*principal*

*vno*

*etype*

*key*

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_register (krb5\_context context, const krb5\_kt\_ops \* ops)**

Register a new keytab backend.

**Parameters:**

*context* a Keberos context.

*ops* a backend to register.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_remove\_entry (krb5\_context**

**context, krb5\_keytab id, krb5\_keytab\_entry \* entry)**

Remove an entry from the keytab, matching is done using **krb5\_kt\_compare()**.

**Parameters:**

*context* a Keberos context.

*id* a keytab.

*entry* the entry to remove

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_resolve (krb5\_context context, const char \* name, krb5\_keytab \* id)**

Resolve the keytab name (of the form 'type:residual') in 'name' into a keytab in 'id'.

**Parameters:**

*context* a Keberos context.

*name* name to resolve

*id* resulting keytab, free with **krb5\_kt\_close()**.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.

**KRB5\_LIB\_FUNCTION krb5\_error\_code KRB5\_LIB\_CALL krb5\_kt\_start\_seq\_get (krb5\_context context, krb5\_keytab id, krb5\_kt\_cursor \* cursor)**

Set 'cursor' to point at the beginning of 'id'.

**Parameters:**

*context* a Keberos context.

*id* a keytab.

*cursor* a newly allocated cursor, free with **krb5\_kt\_end\_seq\_get()**.

**Returns:**

Return an error code or 0, see **krb5\_get\_error\_message()**.