NAME

krb5_get_in_tkt, **krb5_get_in_cred**, **krb5_get_in_tkt_with_password**, **krb5_get_in_tkt_with_keytab**, **krb5_get_in_tkt_with_skey**, **krb5_free_kdc_rep**, **krb5_password_key_proc** - deprecated initial authentication functions

LIBRARY

Kerberos 5 Library (libkrb5, -lkrb5)

SYNOPSIS

#include <krb5.h>

krb5_error_code
krb5_get_in_tkt(*krb5_context context*, *krb5_flags options*, *const krb5_addresses *addrs*, *const krb5_enctype *etypes*, *const krb5_preauthtype *ptypes*, *krb5_key_proc key_proc*, *krb5_const_pointer keyseed*, *krb5_decrypt_proc decrypt_proc*, *krb5_const_pointer decryptarg*, *krb5_creds *creds*, *krb5_ccache ccache*, *krb5_kdc_rep *ret_as_reply*);

krb5_error_code
krb5_get_in_cred(*krb5_context context*, *krb5_flags options*, *const krb5_addresses *addrs*, *const krb5_enctype *etypes*, *const krb5_preauthtype *ptypes*, *const krb5_preauthdata *preauth*, *krb5_key_proc key_proc*, *krb5_const_pointer keyseed*, *krb5_decrypt_proc decrypt_proc*, *krb5_const_pointer decryptarg*, *krb5_creds *creds*, *krb5_kdc_rep *ret_as_reply*);

krb5_error_code
krb5_get_in_tkt_with_password(*krb5_context context*, *krb5_flags options*, *krb5_addresses *addrs*, *const krb5_enctype *etypes*, *const krb5_preauthtype *pre_auth_types*, *const char *password*, *krb5_ccache ccache*, *krb5_creds *creds*, *krb5_kdc_rep *ret_as_reply*);

krb5_error_code
krb5_get_in_tkt_with_keytab(*krb5_context context*, *krb5_flags options*, *krb5_addresses *addrs*, *const krb5_enctype *etypes*, *const krb5_preauthtype *pre_auth_types*, *krb5_keytab keytab*, *krb5_ccache ccache*, *krb5_creds *creds*, *krb5_kdc_rep *ret_as_reply*);

krb5_error_code
krb5_get_in_tkt_with_skey(*krb5_context context*, *krb5_flags options*, *krb5_addresses *addrs*, *const krb5_enctype *etypes*, *const krb5_preauthtype *pre_auth_types*, *const krb5_keyblock *key*, *krb5_ccache ccache*, *krb5_creds *creds*, *krb5_kdc_rep *ret_as_reply*);

krb5_error_code
krb5_free_kdc_rep(*krb5_context context*, *krb5_kdc_rep *rep*);

krb5_error_code
krb5_password_key_proc(*krb5_context context*, *krb5_enctype type*, *krb5_salt salt*,
krb5_const_pointer keyseed, *krb5_keyblock **key*);

DESCRIPTION

All the functions in this manual page are deprecated in the MIT implementation, and will soon be deprecated in Heimdal too, don't use them.

Getting initial credential ticket for a principal. **krb5_get_in_cred** is the function all other krb5_get_in function uses to fetch tickets. The other krb5_get_in function are more specialized and therefor somewhat easier to use.

If your need is only to verify a user and password, consider using krb5_verify_user(3) instead, it have a much simpler interface.

krb5_get_in_tkt and **krb5_get_in_cred** fetches initial credential, queries after key using the *key_proc* argument. The differences between the two function is that **krb5_get_in_tkt** stores the credential in a krb5_creds while **krb5_get_in_cred** stores the credential in a krb5_ccache.

krb5_get_in_tkt_with_password, **krb5_get_in_tkt_with_keytab**, and **krb5_get_in_tkt_with_skey** does the same work as **krb5_get_in_cred** but are more specialized.

krb5_get_in_tkt_with_password uses the clients password to authenticate. If the password argument is NULL the user user queried with the default password query function.

krb5_get_in_tkt_with_keytab searches the given keytab for a service entry for the client principal. If the keytab is NULL the default keytab is used.

krb5_get_in_tkt_with_skey uses a key to get the initial credential.

There are some common arguments to the krb5_get_in functions, these are:

options are the KDC_OPT flags.

etypes is a NULL terminated array of encryption types that the client approves.

addrs a list of the addresses that the initial ticket. If it is NULL the list will be generated by the library.

pre_auth_types a NULL terminated array of pre-authentication types. If *pre_auth_types* is NULL the function will try without pre-authentication and return those pre-authentication that the KDC returned.

ret_as_reply will (if not NULL) be filled in with the response of the KDC and should be free with **krb5_free_kdc_rep**().

key_proc is a pointer to a function that should return a key salted appropriately. Using NULL will use the default password query function.

decrypt_proc Using NULL will use the default decryption function.

decryptarg will be passed to the decryption function *decrypt_proc*.

creds creds should be filled in with the template for a credential that should be requested. The client and server elements of the creds structure must be filled in. Upon return of the function it will be contain the content of the requested credential (*krb5_get_in_cred*), or it will be freed with krb5_free_creds(3) (all the other krb5_get_in functions).

ccache will store the credential in the credential cache *ccache*. The credential cache will not be initialized, thats up the the caller.

krb5_password_key_proc is a library function that is suitable using as the *krb5_key_proc* argument to **krb5_get_in_cred** or **krb5_get_in_tkt**. *keyseed* should be a pointer to a NUL terminated string or NULL. **krb5_password_key_proc** will query the user for the pass on the console if the password isn't given as the argument *keyseed*.

krb5_free_kdc_rep() frees the content of *rep*.

SEE ALSO

krb5(3), krb5_verify_user(3), krb5.conf(5), kerberos(8)