NAME

gnutls_x509_trust_list_verify_crt2 - API function

SYNOPSIS

#include <gnutls/x509.h>

int gnutls_x509_trust_list_verify_crt2(gnutls_x509_trust_list_t *list***, gnutls_x509_crt_t *** *cert_list***, unsigned int** *cert_list_size***, gnutls_typed_vdata_st *** *data***, unsigned int** *elements***, unsigned int** *flags***, unsigned int *** *voutput***, gnutls_verify_output_function** *func***);**

ARGUMENTS

gnutls_x509_trust_list_t list
: The list

gnutls_x509_crt_t * cert_list
: is the certificate list to be verified

unsigned int cert_list_size
: is the certificate list size

gnutls_typed_vdata_st * data
: an array of typed data

unsigned int elements
: the number of data elements

unsigned int flags
: Flags that may be used to change the verification algorithm. Use OR of the gnutls_certificate_verify_flags enumerations.

unsigned int * voutput
: will hold the certificate verification output.

gnutls_verify_output_function func
: If non-null will be called on each chain element verification with the output.

DESCRIPTION

This function will attempt to verify the given certificate chain and return its status. The *voutput* parameter will hold an OR'ed sequence of **gnutls_certificate_status_t** flags.

When a certificate chain of *cert_list_size* with more than one certificates is provided, the verification status will apply to the first certificate in the chain that failed verification. The verification process starts from the end of the chain (from CA to end certificate). The first certificate in the chain must be the end-certificate while the rest of the members may be sorted or not.

Additionally a certificate verification profile can be specified from the ones in **gnutls_certificate_verification_profiles_t** by ORing the result of **GNUTLS_PROFILE_TO_VFLAGS()** to the verification flags.

Additional verification parameters are possible via the *data* types; the acceptable types are **GNUTLS_DT_DNS_HOSTNAME**, **GNUTLS_DT_IP_ADDRESS** and **GNUTLS_DT_KEY_PURPOSE_OID**. The former accepts as data a null-terminated hostname, and the latter a null-terminated object identifier (e.g., **GNUTLS_KP_TLS_WWW_SERVER**). If a DNS hostname is provided then this function will compare the hostname in the end certificate against the given. If names do not match the **GNUTLS_CERT_UNEXPECTED_OWNER** status flag will be set. In addition it will consider certificates provided with **gnutls_x509_trust_list_add_named_crt()**.

If a key purpose OID is provided and the end-certificate contains the extended key usage PKIX extension, it will be required to match the provided OID or be marked for any purpose, otherwise verification will fail with **GNUTLS_CERT_PURPOSE_MISMATCH** status.

RETURNS

On success, **GNUTLS_E_SUCCESS** (0) is returned, otherwise a negative error value. Note that verification failure will not result to an error code, only *voutput* will be updated.

SINCE

3.3.8

REPORTING BUGS

Report bugs to <bugs@gnutls.org>.
Home page: https://www.gnutls.org

COPYRIGHT

SEE ALSO

The full documentation for **gnutls** is maintained as a Texinfo manual. If the /usr/local/share/doc/gnutls/

directory does not contain the HTML form visit

https://www.gnutls.org/manual/