NAME

hx509 certificate functions -

Functions

int **hx509_cert_init** (hx509_context context, const Certificate *c, hx509_cert *cert)
int **hx509_cert_init_data** (hx509_context context, const void *ptr, size_t len, hx509_cert *cert)
void **hx509_cert_free** (hx509_cert cert)
hx509_cert **hx509_cert_ref** (hx509_cert cert)
void **hx509_verify_ctx_f_allow_default_trustanchors** (hx509_verify_ctx ctx, int boolean)
int **hx509_cert_find_subjectAltName_otherName** (hx509_context context, hx509_cert cert, const heim_oid *oid, hx509_octet_string_list *list)
int **hx509_cert_cmp** (hx509_cert p, hx509_cert q)
int **hx509_cert_get_issuer** (hx509_cert p, hx509_name *name)
int **hx509_cert_get_subject** (hx509_cert p, hx509_name *name)
int **hx509_cert_get_base_subject** (hx509_context context, hx509_cert c, hx509_name *name)
int **hx509_cert_get_serialnumber** (hx509_cert p, heim_integer *i)
time_t **hx509_cert_get_notBefore** (hx509_cert p)
time_t **hx509_cert_get_notAfter** (hx509_cert p)
int **hx509_cert_get_SPKI** (hx509_context context, hx509_cert p, SubjectPublicKeyInfo *spki)
int **hx509_cert_get_SPKI_AlgorithmIdentifier** (hx509_context context, hx509_cert p, AlgorithmIdentifier *alg)
int **hx509_cert_get_issuer_unique_id** (hx509_context context, hx509_cert p, heim_bit_string *issuer)
int **hx509_cert_get_subject_unique_id** (hx509_context context, hx509_cert p, heim_bit_string *subject)
int **hx509_verify_hostname** (hx509_context context, const hx509_cert cert, int flags, hx509_hostname_type type, const char *hostname, const struct sockaddr *sa, int sa_size)
hx509_cert_attribute **hx509_cert_get_attribute** (hx509_cert cert, const heim_oid *oid)
int **hx509_cert_set_friendly_name** (hx509_cert cert, const char *name)
const char * **hx509_cert_get_friendly_name** (hx509_cert cert)
int **hx509_query_alloc** (hx509_context context, hx509_query **q)
void **hx509_query_match_option** (hx509_query *q, hx509_query_option option)
int **hx509_query_match_issuer_serial** (hx509_query *q, const Name *issuer, const heim_integer *serialNumber)
int **hx509_query_match_friendly_name** (hx509_query *q, const char *name)
int **hx509_query_match_eku** (hx509_query *q, const heim_oid *eku)
int **hx509_query_match_cmp_func** (hx509_query *q, int(*func)(hx509_context, hx509_cert, void *), void *ctx)
void **hx509_query_free** (hx509_context context, hx509_query *q)
void **hx509_query_statistic_file** (hx509_context context, const char *fn)
void **hx509_query_unparse_stats** (hx509_context context, int printtype, FILE *out)
int **hx509_cert_check_eku** (hx509_context context, hx509_cert cert, const heim_oid *eku, int

allow_any_eku)
int **hx509_cert_binary** (hx509_context context, hx509_cert c, heim_octet_string *os)
int **hx509_print_cert** (hx509_context context, hx509_cert cert, FILE *out)

Detailed Description

See the **The basic certificate** for description and examples.

Function Documentation

int hx509_cert_binary (hx509_context context, hx509_cert c, heim_octet_string * os)

Encodes the hx509 certificate as a DER encode binary.

Parameters:

context A hx509 context.
c the certificate to encode.
os the encode certificate, set to NULL, 0 on case of error. Free the os->data with **hx509_xfree()**.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_check_eku (hx509_context context, hx509_cert cert, const heim_oid * eku, int allow_any_eku)

Check the extended key usage on the hx509 certificate.

Parameters:

context A hx509 context.
cert A hx509 context.
eku the EKU to check for
allow_any_eku if the any EKU is set, allow that to be a substitute.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_cmp (hx509_cert p, hx509_cert q)

Compare to hx509 certificate object, useful for sorting.

Parameters:

p a hx509 certificate object.
q a hx509 certificate object.

Returns:

0 the objects are the same, returns > 0 is p is 'larger' then q, < 0 if p is 'smaller' then q.

int hx509_cert_find_subjectAltName_otherName (hx509_context context, hx509_cert cert, const heim_oid * oid, hx509_octet_string_list * list)

Return a list of subjectAltNames specified by oid in the certificate. On error the

The returned list of octet string should be freed with **hx509_free_octet_string_list()**.

Parameters:

context A hx509 context.
cert a hx509 certificate object.
oid an oid to for SubjectAltName.
list list of matching SubjectAltName.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

void hx509_cert_free (hx509_cert cert)

Free reference to the hx509 certificate object, if the refcounter reaches 0, the object if freed. Its allowed to pass in NULL.

Parameters:

cert the cert to free.

hx509_cert_attribute hx509_cert_get_attribute (hx509_cert cert, const heim_oid * oid)

Get an external attribute for the certificate, examples are friendly name and id.

Parameters:

cert hx509 certificate object to search
oid an oid to search for.

Returns:

an hx509_cert_attribute, only valid as long as the certificate is referenced.

int hx509_cert_get_base_subject (hx509_context context, hx509_cert c, hx509_name * name)

Return the name of the base subject of the hx509 certificate. If the certiicate is a verified proxy certificate, the this function return the base certificate (root of the proxy chain). If the proxy certificate is not verified with the base certificate HX509_PROXY_CERTIFICATE_NOT_CANONICALIZED is returned.

Parameters:

context a hx509 context.
c a hx509 certificate object.
name a pointer to a hx509 name, should be freed by **hx509_name_free()**. See also **hx509_cert_get_subject()**.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

const char* hx509_cert_get_friendly_name (hx509_cert cert)

Get friendly name of the certificate.

Parameters:

cert cert to get the friendly name from.

Returns:

an friendly name or NULL if there is. The friendly name is only valid as long as the certificate is referenced.

int hx509_cert_get_issuer (hx509_cert p, hx509_name * name)

Return the name of the issuer of the hx509 certificate.

Parameters:

p a hx509 certificate object.
name a pointer to a hx509 name, should be freed by **hx509_name_free()**.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_get_issuer_unique_id (hx509_context context, hx509_cert p, heim_bit_string * issuer)

Get a copy of the Issuer Unique ID

Parameters:

context a hx509_context
p a hx509 certificate
issuer the issuer id returned, free with der_free_bit_string()

Returns:

An hx509 error code, see **hx509_get_error_string()**. The error code HX509_EXTENSION_NOT_FOUND is returned if the certificate doesn’t have a issuerUniqueID

time_t hx509_cert_get_notAfter (hx509_cert p)
Get notAfter time of the certificate.

Parameters:
p a hx509 certificate object.

Returns:
return not after time.

time_t hx509_cert_get_notBefore (hx509_cert p)
Get notBefore time of the certificate.

Parameters:
p a hx509 certificate object.

Returns:
return not before time

int hx509_cert_get_serialnumber (hx509_cert p, heim_integer * i)
Get serial number of the certificate.

Parameters:
p a hx509 certificate object.
i serial number, should be freed ith der_free_heim_integer().

Returns:
An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_get_SPKI (hx509_context context, hx509_cert p, SubjectPublicKeyInfo * spki)
Get the SubjectPublicKeyInfo structure from the hx509 certificate.

Parameters:
context a hx509 context.
p a hx509 certificate object.
spki SubjectPublicKeyInfo, should be freed with free_SubjectPublicKeyInfo().

Returns:
An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_get_SPKI_AlgorithmIdentifier (hx509_context context, hx509_cert p, AlgorithmIdentifier

*** alg)**
Get the AlgorithmIdentifier from the hx509 certificate.

Parameters:
context a hx509 context.
p a hx509 certificate object.
alg AlgorithmIdentifier, should be freed with free_AlgorithmIdentifier(). The algorithmidentifier is typicly rsaEncryption, or id-ecPublicKey, or some other public key mechanism.

Returns:
An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_get_subject (hx509_cert p, hx509_name * name)
Return the name of the subject of the hx509 certificate.

Parameters:
p a hx509 certificate object.
name a pointer to a hx509 name, should be freed by **hx509_name_free()**. See also **hx509_cert_get_base_subject()**.

Returns:
An hx509 error code, see **hx509_get_error_string()**.

int hx509_cert_get_subject_unique_id (hx509_context context, hx509_cert p, heim_bit_string * subject)
Get a copy of the Subect Unique ID

Parameters:
context a hx509_context
p a hx509 certificate
subject the subject id returned, free with der_free_bit_string()

Returns:
An hx509 error code, see **hx509_get_error_string()**. The error code HX509_EXTENSION_NOT_FOUND is returned if the certificate doesn't have a subjectUniqueID

int hx509_cert_init (hx509_context context, const Certificate * c, hx509_cert * cert)
Allocate and init an hx509 certificate object from the decoded certificate 'c'.

Parameters:

context A hx509 context.
c
cert

Returns:
Returns an hx509 error code.

int hx509_cert_init_data (hx509_context context, const void * ptr, size_t len, hx509_cert * cert)
Just like **hx509_cert_init()**, but instead of a decode certificate takes an pointer and length to a memory region that contains a DER/BER encoded certificate.

If the memory region doesn't contain just the certificate and nothing more the function will fail with HX509_EXTRA_DATA_AFTER_STRUCTURE.

Parameters:
context A hx509 context.
ptr pointer to memory region containing encoded certificate.
len length of memory region.
cert a return pointer to a hx509 certificate object, will contain NULL on error.

Returns:
An hx509 error code, see **hx509_get_error_string()**.

hx509_cert hx509_cert_ref (hx509_cert cert)
Add a reference to a hx509 certificate object.

Parameters:
cert a pointer to an hx509 certificate object.

Returns:
the same object as is passed in.

int hx509_cert_set_friendly_name (hx509_cert cert, const char * name)
Set the friendly name on the certificate.

Parameters:
cert The certificate to set the friendly name on
name Friendly name.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_print_cert (hx509_context context, hx509_cert cert, FILE * out)

Print a simple representation of a certificate

Parameters:

context A hx509 context, can be NULL
cert certificate to print
out the stdio output stream, if NULL, stdout is used

Returns:

An hx509 error code

int hx509_query_alloc (hx509_context context, hx509_query ** q)

Allocate an query controller. Free using **hx509_query_free()**.

Parameters:

context A hx509 context.
q return pointer to a hx509_query.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

void hx509_query_free (hx509_context context, hx509_query * q)

Free the query controller.

Parameters:

context A hx509 context.
q a pointer to the query controller.

int hx509_query_match_cmp_func (hx509_query * q, int(*)(hx509_context, hx509_cert, void *) func, void * ctx)

Set the query controller to match using a specific match function.

Parameters:

q a hx509 query controller.
func function to use for matching, if the argument is NULL, the match function is removed.
ctx context passed to the function.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_query_match_eku (hx509_query * q, const heim_oid * eku)

Set the query controller to require an one specific EKU (extended key usage). Any previous EKU matching is overwitten. If NULL is passed in as the eku, the EKU requirement is reset.

Parameters:

q a hx509 query controller.
eku an EKU to match on.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_query_match_friendly_name (hx509_query * q, const char * name)

Set the query controller to match on a friendly name

Parameters:

q a hx509 query controller.
name a friendly name to match on

Returns:

An hx509 error code, see **hx509_get_error_string()**.

int hx509_query_match_issuer_serial (hx509_query * q, const Name * issuer, const heim_integer * serialNumber)

Set the issuer and serial number of match in the query controller. The function make copies of the isser and serial number.

Parameters:

q a hx509 query controller
issuer issuer to search for
serialNumber the serialNumber of the issuer.

Returns:

An hx509 error code, see **hx509_get_error_string()**.

void hx509_query_match_option (hx509_query * q, hx509_query_option option)

Set match options for the hx509 query controller.

Parameters:

q query controller.
option options to control the query controller.

Returns:
An hx509 error code, see **hx509_get_error_string()**.

void hx509_query_statistic_file (hx509_context context, const char * fn)
Set a statistic file for the query statistics.

Parameters:
context A hx509 context.
fn statistics file name

void hx509_query_unparse_stats (hx509_context context, int printtype, FILE * out)
Unparse the statistics file and print the result on a FILE descriptor.

Parameters:
context A hx509 context.
printtype tyep to print
out the FILE to write the data on.

void hx509_verify_ctx_f_allow_default_trustanchors (hx509_verify_ctx ctx, int boolean)
Allow using the operating system builtin trust anchors if no other trust anchors are configured.

Parameters:
ctx a verification context
boolean if non zero, useing the operating systems builtin trust anchors.

Returns:
An hx509 error code, see **hx509_get_error_string()**.

int hx509_verify_hostname (hx509_context context, const hx509_cert cert, int flags, hx509_hostname_type type, const char * hostname, const struct sockaddr * sa, int sa_size)
Verify that the certificate is allowed to be used for the hostname and address.

Parameters:
context A hx509 context.
cert the certificate to match with
flags Flags to modify the behavior:

- HX509_VHN_F_ALLOW_NO_MATCH no match is ok

type type of hostname:

- HX509_HN_HOSTNAME for plain hostname.

- HX509_HN_DNSSRV for DNS SRV names.

hostname the hostname to check
sa address of the host
sa_size length of address

Returns:
An hx509 error code, see **hx509_get_error_string()**.