

**NAME**

hx509 CA functions -

**Functions**

int **hx509\_ca\_tbs\_init** (hx509\_context context, hx509\_ca\_tbs \*tbs)  
void **hx509\_ca\_tbs\_free** (hx509\_ca\_tbs \*tbs)  
int **hx509\_ca\_tbs\_set\_notBefore** (hx509\_context context, hx509\_ca\_tbs tbs, time\_t t)  
int **hx509\_ca\_tbs\_set\_notAfter** (hx509\_context context, hx509\_ca\_tbs tbs, time\_t t)  
int **hx509\_ca\_tbs\_set\_notAfter\_lifetime** (hx509\_context context, hx509\_ca\_tbs tbs, time\_t delta)  
struct units \* **hx509\_ca\_tbs\_template\_units** (void)  
int **hx509\_ca\_tbs\_set\_template** (hx509\_context context, hx509\_ca\_tbs tbs, int flags, hx509\_cert cert)  
int **hx509\_ca\_tbs\_set\_ca** (hx509\_context context, hx509\_ca\_tbs tbs, int pathLenConstraint)  
int **hx509\_ca\_tbs\_set\_proxy** (hx509\_context context, hx509\_ca\_tbs tbs, int pathLenConstraint)  
int **hx509\_ca\_tbs\_set\_domaincontroller** (hx509\_context context, hx509\_ca\_tbs tbs)  
int **hx509\_ca\_tbs\_set\_spki** (hx509\_context context, hx509\_ca\_tbs tbs, const SubjectPublicKeyInfo \*spki)  
int **hx509\_ca\_tbs\_set\_serialnumber** (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_integer \*serialNumber)  
int **hx509\_ca\_tbs\_add\_eku** (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_oid \*oid)  
int **hx509\_ca\_tbs\_add\_crl\_dp\_uri** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*uri, hx509\_name issuername)  
int **hx509\_ca\_tbs\_add\_san\_otherName** (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_oid \*oid, const heim\_octet\_string \*os)  
int **hx509\_ca\_tbs\_add\_san\_pkinit** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*principal)  
int **hx509\_ca\_tbs\_add\_san\_ms\_upn** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*principal)  
int **hx509\_ca\_tbs\_add\_san\_jid** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*jid)  
int **hx509\_ca\_tbs\_add\_san\_hostname** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*dnsname)  
int **hx509\_ca\_tbs\_add\_san\_rfc822name** (hx509\_context context, hx509\_ca\_tbs tbs, const char \*rfc822Name)  
int **hx509\_ca\_tbs\_set\_subject** (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_name subject)  
int **hx509\_ca\_tbs\_set\_unique** (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_bit\_string \*subjectUniqueID, const heim\_bit\_string \*issuerUniqueID)  
int **hx509\_ca\_tbs\_subject\_expand** (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_env env)  
int **hx509\_ca\_sign** (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_cert signer, hx509\_cert \*certificate)  
int **hx509\_ca\_sign\_self** (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_private\_key signer, hx509\_cert \*certificate)

**Detailed Description**

See the **Hx509 CA functions** for description and examples.

**Function Documentation**

**int hx509\_ca\_sign (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_cert signer, hx509\_cert \* certificate)**

Sign a to-be-signed certificate object with a issuer certificate.

The caller needs to at least have called the following functions on the to-be-signed certificate object:

⊕ **hx509\_ca\_tbs\_init()**

⊕ **hx509\_ca\_tbs\_set\_subject()**

⊕ **hx509\_ca\_tbs\_set\_spki()**

When done the to-be-signed certificate object should be freed with **hx509\_ca\_tbs\_free()**.

When creating self-signed certificate use **hx509\_ca\_sign\_self()** instead.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*signer* the CA certificate object to sign with (need private key).

*certificate* return certificate, free with **hx509\_cert\_free()**.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_sign\_self (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_private\_key signer, hx509\_cert \* certificate)**

Work just like **hx509\_ca\_sign()** but signs it-self.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*signer* private key to sign with.

*certificate* return certificate, free with **hx509\_cert\_free()**.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_crl\_dp\_uri (hx509\_context context, hx509\_ca\_tbs tbs, const char \* uri, hx509\_name issuername)**

Add CRL distribution point URI to the to-be-signed certificate object.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*uri* uri to the CRL.

*issuename* name of the issuer.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

issuename not supported

**int hx509\_ca\_tbs\_add\_eku (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_oid \* oid)**

An an extended key usage to the to-be-signed certificate object. Duplicates will detected and not added.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*oid* extended key usage to add.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_hostname (hx509\_context context, hx509\_ca\_tbs tbs, const char \* dnsname)**

Add a Subject Alternative Name hostname to to-be-signed certificate object. A domain match starts with ., an exact match does not.

Example of a an domain match: .domain.se matches the hostname host.domain.se.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*dnsname* a hostame.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_jid (hx509\_context context, hx509\_ca\_tbs tbs, const char \* jid)**

Add a Jabber/XMPP jid Subject Alternative Name to the to-be-signed certificate object. The jid is an

UTF8 string.

**Parameters:**

*context* A hx509 context.  
*tbs* object to be signed.  
*jid* string of an a jabber id in UTF8.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_ms\_upn (hx509\_context context, hx509\_ca\_tbs tbs, const char \* principal)**

Add Microsoft UPN Subject Alternative Name to the to-be-signed certificate object. The principal string is a UTF8 string.

**Parameters:**

*context* A hx509 context.  
*tbs* object to be signed.  
*principal* Microsoft UPN string.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_otherName (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_oid \* oid, const heim\_octet\_string \* os)**

Add Subject Alternative Name otherName to the to-be-signed certificate object.

**Parameters:**

*context* A hx509 context.  
*tbs* object to be signed.  
*oid* the oid of the OtherName.  
*os* data in the other name.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_pkinit (hx509\_context context, hx509\_ca\_tbs tbs, const char \* principal)**

Add Kerberos Subject Alternative Name to the to-be-signed certificate object. The principal string is a UTF8 string.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*principal* Kerberos principal to add to the certificate.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_add\_san\_rfc822name (hx509\_context context, hx509\_ca\_tbs tbs, const char \* rfc822Name)**

Add a Subject Alternative Name rfc822 (email address) to to-be-signed certificate object.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*rfc822Name* a string to a email address.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**void hx509\_ca\_tbs\_free (hx509\_ca\_tbs \* tbs)**

Free an To Be Signed object.

**Parameters:**

*tbs* object to free.

**int hx509\_ca\_tbs\_init (hx509\_context context, hx509\_ca\_tbs \* tbs)**

Allocate an to-be-signed certificate object that will be converted into an certificate.

**Parameters:**

*context* A hx509 context.

*tbs* returned to-be-signed certificate object, free with **hx509\_ca\_tbs\_free()**.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_ca (hx509\_context context, hx509\_ca\_tbs tbs, int pathLenConstraint)**

Make the to-be-signed certificate object a CA certificate. If the pathLenConstraint is negative path length constraint is used.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*pathLenConstraint* path length constraint, negative, no constraint.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_domaincontroller (hx509\_context context, hx509\_ca\_tbs tbs)**

Make the to-be-signed certificate object a windows domain controller certificate.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_notAfter (hx509\_context context, hx509\_ca\_tbs tbs, time\_t t)**

Set the absolute time when the certificate is valid to.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*t* time when the certificate will expire

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_notAfter\_lifetime (hx509\_context context, hx509\_ca\_tbs tbs, time\_t delta)**

Set the relative time when the certificate is going to expire.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*delta* seconds to the certificate is going to expire.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_notBefore (hx509\_context context, hx509\_ca\_tbs tbs, time\_t t)**

Set the absolute time when the certificate is valid from. If not set the current time will be used.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*t* time the certificated will start to be valid

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_proxy (hx509\_context context, hx509\_ca\_tbs tbs, int pathLenConstraint)**

Make the to-be-signed certificate object a proxy certificate. If the pathLenConstraint is negative path length constraint is used.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*pathLenConstraint* path length constraint, negative, no constraint.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_serialnumber (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_integer \* serialNumber)**

Set the serial number to use for to-be-signed certificate object.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*serialNumber* serial number to use for the to-be-signed certificate object.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_spki (hx509\_context context, hx509\_ca\_tbs tbs, const SubjectPublicKeyInfo \* spki)**

Set the subject public key info (SPKI) in the to-be-signed certificate object. SPKI is the public key and key related parameters in the certificate.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*spki* subject public key info to use for the to-be-signed certificate object.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_subject (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_name subject)**

Set the subject name of a to-be-signed certificate object.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*subject* the name to set a subject.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_template (hx509\_context context, hx509\_ca\_tbs tbs, int flags, hx509\_cert cert)**

Initialize the to-be-signed certificate object from a template certifiante.

**Parameters:**

*context* A hx509 context.

*tbs* object to be signed.

*flags* bit field selecting what to copy from the template certifiante.

*cert* template certificate.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_set\_unique (hx509\_context context, hx509\_ca\_tbs tbs, const heim\_bit\_string \* subjectUniqueID, const heim\_bit\_string \* issuerUniqueID)**

Set the issuerUniqueID and subjectUniqueID

These are only supposed to be used considered with version 2 certificates, replaced by the two extensions SubjectKeyIdentifier and IssuerKeyIdentifier. This function is to allow application using legacy protocol to issue them.

**Parameters:**

*context* A hx509 context.



*tbs* object to be signed.  
*issuerUniqueID* to be set  
*subjectUniqueID* to be set

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**int hx509\_ca\_tbs\_subject\_expand (hx509\_context context, hx509\_ca\_tbs tbs, hx509\_env env)**

Expand the the subject name in the to-be-signed certificate object using **hx509\_name\_expand()**.

**Parameters:**

*context* A hx509 context.  
*tbs* object to be signed.  
*env* environment variable to expand variables in the subject name, see **hx509\_env\_init()**.

**Returns:**

An hx509 error code, see **hx509\_get\_error\_string()**.

**struct units\* hx509\_ca\_tbs\_template\_units (void) [read]**

Make of template units, use to build flags argument to **hx509\_ca\_tbs\_set\_template()** with **parse\_units()**.

**Returns:**

an units structure.