

**NAME**

Heimdal NTLM library -

**Functions**

void **heim\_ntlm\_free\_buf** (struct **ntlm\_buf** \*p)  
void **heim\_ntlm\_free\_targetinfo** (struct **ntlm\_targetinfo** \*ti)  
int **heim\_ntlm\_encode\_targetinfo** (const struct **ntlm\_targetinfo** \*ti, int ucs2, struct **ntlm\_buf** \*data)  
int **heim\_ntlm\_decode\_targetinfo** (const struct **ntlm\_buf** \*data, int ucs2, struct **ntlm\_targetinfo** \*ti)  
void **heim\_ntlm\_free\_type1** (struct **ntlm\_type1** \*data)  
int **heim\_ntlm\_encode\_type1** (const struct **ntlm\_type1** \*type1, struct **ntlm\_buf** \*data)  
void **heim\_ntlm\_free\_type2** (struct **ntlm\_type2** \*data)  
int **heim\_ntlm\_encode\_type2** (const struct **ntlm\_type2** \*type2, struct **ntlm\_buf** \*data)  
void **heim\_ntlm\_free\_type3** (struct **ntlm\_type3** \*data)  
int **heim\_ntlm\_encode\_type3** (const struct **ntlm\_type3** \*type3, struct **ntlm\_buf** \*data)  
int **heim\_ntlm\_nt\_key** (const char \*password, struct **ntlm\_buf** \*key)  
int **heim\_ntlm\_calculate\_ntlm1** (void \*key, size\_t len, unsigned char challenge[8], struct **ntlm\_buf** \*answer)  
int **heim\_ntlm\_build\_ntlm1\_master** (void \*key, size\_t len, struct **ntlm\_buf** \*session, struct **ntlm\_buf** \*master)  
int **heim\_ntlm\_build\_ntlm2\_master** (void \*key, size\_t len, struct **ntlm\_buf** \*blob, struct **ntlm\_buf** \*session, struct **ntlm\_buf** \*master)  
int **heim\_ntlm\_keyex\_unwrap** (struct **ntlm\_buf** \*baseKey, struct **ntlm\_buf** \*encryptedSession, struct **ntlm\_buf** \*session)  
int **heim\_ntlm\_ntlmv2\_key** (const void \*key, size\_t len, const char \*username, const char \*target, unsigned char ntlmv2[16])  
int **heim\_ntlm\_calculate\_lm2** (const void \*key, size\_t len, const char \*username, const char \*target, const unsigned char serverchallenge[8], unsigned char ntlmv2[16], struct **ntlm\_buf** \*answer)  
int **heim\_ntlm\_calculate\_ntlm2** (const void \*key, size\_t len, const char \*username, const char \*target, const unsigned char serverchallenge[8], const struct **ntlm\_buf** \*infotarget, unsigned char ntlmv2[16], struct **ntlm\_buf** \*answer)  
int **heim\_ntlm\_verify\_ntlm2** (const void \*key, size\_t len, const char \*username, const char \*target, time\_t now, const unsigned char serverchallenge[8], const struct **ntlm\_buf** \*answer, struct **ntlm\_buf** \*infotarget, unsigned char ntlmv2[16])

**Detailed Description**

The NTLM core functions implement the string2key generation function, message encode and decode function, and the hash function functions.

**Function Documentation**

int **heim\_ntlm\_build\_ntlm1\_master** (void \* key, size\_t len, struct **ntlm\_buf** \* session, struct **ntlm\_buf** \* master)

Generates an NTLMv1 session random with associated session master key.

**Parameters:**

*key* the ntlm v1 key

*len* length of key

*session* generated session nonce, should be freed with **heim\_ntlm\_free\_buf()**.

*master* calculated session master key, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_build\_ntlm2\_master (void \* key, size\_t len, struct ntlm\_buf \* blob, struct ntlm\_buf \* session, struct ntlm\_buf \* master)**

Generates an NTLMv2 session random with associated session master key.

**Parameters:**

*key* the NTLMv2 key

*len* length of key

*blob* the NTLMv2 'blob'

*session* generated session nonce, should be freed with **heim\_ntlm\_free\_buf()**.

*master* calculated session master key, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_calculate\_lm2 (const void \* key, size\_t len, const char \* username, const char \* target, const unsigned char serverchallenge[8], unsigned char ntlmv2[16], struct ntlm\_buf \* answer)**

Calculate LMv2 response

**Parameters:**

*key* the ntlm key

*len* length of key

*username* name of the user, as sent in the message, assumed to be in UTF8.

*target* the name of the target, assumed to be in UTF8.

*serverchallenge* challenge as sent by the server in the type2 message.

*ntlmv2* calculated session key

*answer* ntlm response answer, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_calculate\_ntlm1 (void \* key, size\_t len, unsigned char challenge[8], struct ntlm\_buf \* answer)**

Calculate NTLMv1 response hash

**Parameters:**

*key* the ntlm v1 key

*len* length of key

*challenge* sent by the server

*answer* calculated answer, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_calculate\_ntlm2 (const void \* key, size\_t len, const char \* username, const char \* target, const unsigned char serverchallenge[8], const struct ntlm\_buf \* infotarget, unsigned char ntlmv2[16], struct ntlm\_buf \* answer)**

Calculate NTLMv2 response

**Parameters:**

*key* the ntlm key

*len* length of key

*username* name of the user, as sent in the message, assumed to be in UTF8.

*target* the name of the target, assumed to be in UTF8.

*serverchallenge* challenge as sent by the server in the type2 message.

*infotarget* infotarget as sent by the server in the type2 message.

*ntlmv2* calculated session key

*answer* ntlm response answer, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_decode\_targetinfo (const struct ntlm\_buf \* data, int ucs2, struct ntlm\_targetinfo \* ti)**

Decodes an NTLM targetinfo message

**Parameters:**

*data* input data buffer with the encode NTLM targetinfo message

*ucs2* if the strings should be encoded with ucs2 (selected by flag in message).

*ti* the decoded target info, should be freed with **heim\_ntlm\_free\_targetinfo()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_encode\_targetinfo (const struct ntlm\_targetinfo \* ti, int ucs2, struct ntlm\_buf \* data)**

Encodes a ntlm\_targetinfo message.

**Parameters:**

*ti* the ntlm\_targetinfo message to encode.

*ucs2* ignored

*data* is the return buffer with the encoded message, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_encode\_type1 (const struct ntlm\_type1 \* type1, struct ntlm\_buf \* data)**

Encodes an ntlm\_type1 message.

**Parameters:**

*type1* the ntlm\_type1 message to encode.

*data* is the return buffer with the encoded message, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_encode\_type2 (const struct ntlm\_type2 \* type2, struct ntlm\_buf \* data)**

Encodes an ntlm\_type2 message.

**Parameters:**

*type2* the ntlm\_type2 message to encode.

*data* is the return buffer with the encoded message, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_encode\_type3 (const struct ntlm\_type3 \* type3, struct ntlm\_buf \* data)**

Encodes an ntlm\_type3 message.

**Parameters:**

*type3* the ntlm\_type3 message to encode.

*data* is the return buffer with the encoded message, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**void heim\_ntlm\_free\_buf (struct ntlm\_buf \* p)**

heim\_ntlm\_free\_buf frees the ntlm buffer

**Parameters:**

*p* buffer to be freed

**void heim\_ntlm\_free\_targetinfo (struct ntlm\_targetinfo \* ti)**

Frees the ntlm\_targetinfo message

**Parameters:**

*ti* targetinfo to be freed

**void heim\_ntlm\_free\_type1 (struct ntlm\_type1 \* data)**

Frees the **ntlm\_type1** message

**Parameters:**

*data* message to be freed

**void heim\_ntlm\_free\_type2 (struct ntlm\_type2 \* data)**

Frees the **ntlm\_type2** message

**Parameters:**

*data* message to be freed

**void heim\_ntlm\_free\_type3 (struct ntlm\_type3 \* data)**

Frees the **ntlm\_type3** message

**Parameters:**

*data* message to be freed

**int heim\_ntlm\_keyex\_unwrap (struct ntlm\_buf \* baseKey, struct ntlm\_buf \* encryptedSession, struct ntlm\_buf \* session)**

Given a key and encrypted session, unwrap the session key

**Parameters:**

*baseKey* the sessionBaseKey

*encryptedSession* encrypted session, type3.session field.

*session* generated session nonce, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_nt\_key (const char \* password, struct ntlm\_buf \* key)**

Calculate the NTLM key, the password is assumed to be in UTF8.

**Parameters:**

*password* password to calculate the key for.

*key* calculated key, should be freed with **heim\_ntlm\_free\_buf()**.

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.

**int heim\_ntlm\_ntlmv2\_key (const void \* key, size\_t len, const char \* username, const char \* target, unsigned char ntlmv2[16])**

Generates an NTLMv2 session key.

**Parameters:**

*key* the ntlm key

*len* length of key

*username* name of the user, as sent in the message, assumed to be in UTF8.

*target* the name of the target, assumed to be in UTF8.

*ntlmv2* the ntlmv2 session key

**Returns:**

0 on success, or an error code on failure.

**int heim\_ntlm\_verify\_ntlm2 (const void \* key, size\_t len, const char \* username, const char \* target, time\_t now, const unsigned char serverchallenge[8], const struct ntlm\_buf \* answer, struct ntlm\_buf \* infotarget, unsigned char ntlmv2[16])**

Verify NTLMv2 response.

**Parameters:**

*key* the ntlm key

*len* length of key

*username* name of the user, as sent in the message, assumed to be in UTF8.

*target* the name of the target, assumed to be in UTF8.

*now* the time now (0 if the library should pick it up itself)

*serverchallenge* challenge as sent by the server in the type2 message.

*answer* ntlm response answer, should be freed with **heim\_ntlm\_free\_buf()**.

*infotarget* infotarget as sent by the server in the type2 message.

*ntlmv2* calculated session key

**Returns:**

In case of success 0 is return, an errors, a errno in what went wrong.