

NAME

SHA1, SHA1_Init, SHA1_Update, SHA1_Final, SHA224, SHA224_Init, SHA224_Update, SHA224_Final, SHA256, SHA256_Init, SHA256_Update, SHA256_Final, SHA384, SHA384_Init, SHA384_Update, SHA384_Final, SHA512, SHA512_Init, SHA512_Update, SHA512_Final - Secure Hash Algorithm

SYNOPSIS

```
#include <openssl/sha.h>

int SHA1_Init(SHA_CTX *c);
int SHA1_Update(SHA_CTX *c, const void *data, size_t len);
int SHA1_Final(unsigned char *md, SHA_CTX *c);
unsigned char *SHA1(const unsigned char *d, size_t n,
unsigned char *md);

int SHA224_Init(SHA256_CTX *c);
int SHA224_Update(SHA256_CTX *c, const void *data, size_t len);
int SHA224_Final(unsigned char *md, SHA256_CTX *c);
unsigned char *SHA224(const unsigned char *d, size_t n,
unsigned char *md);

int SHA256_Init(SHA256_CTX *c);
int SHA256_Update(SHA256_CTX *c, const void *data, size_t len);
int SHA256_Final(unsigned char *md, SHA256_CTX *c);
unsigned char *SHA256(const unsigned char *d, size_t n,
unsigned char *md);

int SHA384_Init(SHA512_CTX *c);
int SHA384_Update(SHA512_CTX *c, const void *data, size_t len);
int SHA384_Final(unsigned char *md, SHA512_CTX *c);
unsigned char *SHA384(const unsigned char *d, size_t n,
unsigned char *md);

int SHA512_Init(SHA512_CTX *c);
int SHA512_Update(SHA512_CTX *c, const void *data, size_t len);
int SHA512_Final(unsigned char *md, SHA512_CTX *c);
unsigned char *SHA512(const unsigned char *d, size_t n,
unsigned char *md);
```

DESCRIPTION

Applications should use the higher level functions [EVP_DigestInit\(3\)](#) etc. instead of calling the hash functions directly.

SHA-1 (Secure Hash Algorithm) is a cryptographic hash function with a 160 bit output.

SHA1() computes the SHA-1 message digest of the **n** bytes at **d** and places it in **md** (which must have space for `SHA_DIGEST_LENGTH == 20` bytes of output). If **md** is NULL, the digest is placed in a static array.

Note: setting **md** to NULL is **not thread safe**.

The following functions may be used if the message is not completely stored in memory:

SHA1_Init() initializes a **SHA_CTX** structure.

SHA1_Update() can be called repeatedly with chunks of the message to be hashed (**len** bytes at **data**).

SHA1_Final() places the message digest in **md**, which must have space for `SHA_DIGEST_LENGTH == 20` bytes of output, and erases the **SHA_CTX**.

The SHA224, SHA256, SHA384 and SHA512 families of functions operate in the same way as for the SHA1 functions. Note that SHA224 and SHA256 use a **SHA256_CTX** object instead of **SHA_CTX**. SHA384 and

SHA512 use **SHA512_CTX**. The buffer **md** must have space for the output from the SHA variant being used (defined by `SHA224_DIGEST_LENGTH`, `SHA256_DIGEST_LENGTH`, `SHA384_DIGEST_LENGTH` and `SHA512_DIGEST_LENGTH`). Also note that, as for the `SHA1()` function above, the `SHA224()`, `SHA256()`, `SHA384()` and `SHA512()` functions are not thread safe if **md** is NULL.

The predecessor of SHA-1, SHA, is also implemented, but it should be used only when backward compatibility is required.

RETURN VALUES

`SHA1()`, `SHA224()`, `SHA256()`, `SHA384()` and `SHA512()` return a pointer to the hash value.

`SHA1_Init()`, `SHA1_Update()` and `SHA1_Final()` and equivalent SHA224, SHA256, SHA384 and SHA512 functions return 1 for success, 0 otherwise.

CONFORMING TO

US Federal Information Processing Standard FIPS PUB 180-4 (Secure Hash Standard), ANSI X9.30

SEE ALSO

[EVP_DigestInit\(3\)](#)

COPYRIGHT

Copyright 2000-2016 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the OpenSSL license (the “License”). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.