NAME

SHA_Init, **SHA_Update**, **SHA_Final**, **SHA_End**, **SHA_File**, **SHA_FileChunk**, **SHA_Data**, **SHA1_Init**, **SHA1_Update**, **SHA1_Final**, **SHA1_End**, **SHA1_File**, **SHA1_FileChunk**, **SHA1_Data** - calculate the FIPS 160 and 160-1 ‘‘SHA’’ message digests

LIBRARY

Message Digest (MD4, MD5, etc.) Support Library (libmd, -lmd)

SYNOPSIS

#include <sys/types.h>
#include <sha.h>

void
SHA_Init(*SHA_CTX *context*);

void
SHA_Update(*SHA_CTX *context*, *const unsigned char *data*, *size_t len*);

void
SHA_Final(*unsigned char digest[20]*, *SHA_CTX *context*);

*char **
SHA_End(*SHA_CTX *context*, *char *buf*);

*char **
SHA_File(*const char *filename*, *char *buf*);

*char **
SHA_FileChunk(*const char *filename*, *char *buf*, *off_t offset*, *off_t length*);

*char **
SHA_Data(*const unsigned char *data*, *unsigned int len*, *char *buf*);

void
SHA1_Init(*SHA_CTX *context*);

void
SHA1_Update(*SHA_CTX *context*, *const unsigned char *data*, *size_t len*);

void

SHA1_Final(*unsigned char digest[20], SHA_CTX *context*);

*char **
SHA1_End(*SHA_CTX *context, char *buf*);

*char **
SHA1_File(*const char *filename, char *buf*);

*char **
SHA1_FileChunk(*const char *filename, char *buf, off_t offset, off_t length*);

*char **
SHA1_Data(*const unsigned char *data, unsigned int len, char *buf*);

DESCRIPTION

The SHA_ and SHA1_ functions calculate a 160-bit cryptographic checksum (digest) for any number of input bytes. A cryptographic checksum is a one-way hash function; that is, it is computationally impractical to find the input corresponding to a particular output. This net result is a "fingerprint" of the input-data, which does not disclose the actual input.

SHA (or SHA-0) is the original Secure Hash Algorithm specified in FIPS 160. It was quickly proven insecure, and has been superseded by SHA-1. SHA-0 is included for compatibility purposes only.

The **SHA1_Init**(), **SHA1_Update**(), and **SHA1_Final**() functions are the core functions. Allocate an *SHA_CTX*, initialize it with **SHA1_Init**(), run over the data with **SHA1_Update**(), and finally extract the result using **SHA1_Final**(), which will also erase the *SHA_CTX*.

SHA1_End() is a wrapper for **SHA1_Final**() which converts the return value to a 41-character (including the terminating '\0') ASCII string which represents the 160 bits in hexadecimal.

SHA1_File() calculates the digest of a file, and uses **SHA1_End**() to return the result. If the file cannot be opened, a null pointer is returned. **SHA1_FileChunk**() is similar to **SHA1_File**(), but it only calculates the digest over a byte-range of the file specified, starting at *offset* and spanning *length* bytes. If the *length* parameter is specified as 0, or more than the length of the remaining part of the file, **SHA1_FileChunk**() calculates the digest from *offset* to the end of file. **SHA1_Data**() calculates the digest of a chunk of data in memory, and uses **SHA1_End**() to return the result.

When using **SHA1_End**(), **SHA1_File**(), or **SHA1_Data**(), the *buf* argument can be a null pointer, in which case the returned string is allocated with malloc(3) and subsequently must be explicitly deallocated using free(3) after use. If the *buf* argument is non-null it must point to at least 41 characters

of buffer space.

ERRORS

The **SHA1_End**() function called with a null buf argument may fail and return NULL if:

[ENOMEM] Insufficient storage space is available.

The **SHA1_File**() and **SHA1_FileChunk**() may return NULL when underlying open(2), fstat(2), lseek(2), or SHA1_End(3) fail.

SEE ALSO

md4(3), md5(3), ripemd(3), sha256(3), sha512(3), skein(3)

HISTORY

These functions appeared in FreeBSD 4.0.

AUTHORS

The core hash routines were implemented by Eric Young based on the published FIPS standards.

BUGS

The SHA1 algorithm has been proven to be vulnerable to practical collision attacks and should not be relied upon to produce unique outputs, *nor should it be used as part of a new cryptographic signature scheme*.