sha-1

function_interface

`string valueSHA1(char *name);` //input filename and return the sha-1 string type

introduction

安全哈希算法（Secure Hash Algorithm）主要适用于数字签名标准 （Digital Signature Standard DSS）里面定义的数字签名算法（Digital Signature Algorithm DSA）。对于长度小于2^64位的消息，SHA1会产生一个160位的消息摘要。当接收到消息的时候，这个消息摘要可以用来验证数据的完整性。在传输的过程中，数据很可能会发生变化，那么这时候就会产生不同的消息摘要。

principle

在SHA1算法中，我们必须把原始消息（字符串，文件等）转换成位字符串。SHA1算法只接受位作为输入。假设我们对字符串“abc”产生消息摘要。首先，我们将它转换成位字符串如下：

01100001 01100010 01100011

————————————————

‘a’=97 ‘b’=98 ‘c’=99

这个位字符串的长度为24。下面我们需要5个步骤来计算消息摘要MAC。

补位

消息必须进行补位，以使其长度在对512取模以后的余数是448。也就是说，（补位后的消息长度）%512 = 448。即使长度已经满足对512取模后余数是448，补位也必须要进行。

补位是这样进行的：先补一个1，然后再补0，直到长度满足对512取模后余数是448。总而言之，补位是至少补一位，最多补512位。

补长度

所谓的补长度是将原始数据的长度补到已经进行了补位操作的消息后面。通常用一个64位的数据来表示原始消息的长度。如果消息长度不大于2^64，那么第一个字就是0。

如果原始的消息经过补位后长度超过了512，我们需要将它补成512的倍数。然后我们把整个消息分成一个一个512位的数据块，分别处理每一个数据块，从而得到消息摘要。

使用的常量

一系列的常量字K(0), K(1), ... , K(79)，如果以16进制给出。它们如下：

Kt = 0x5A827999 (0 <= t <= 19)

Kt = 0x6ED9EBA1 (20 <= t <= 39)

Kt = 0x8F1BBCDC (40 <= t <= 59)

Kt = 0xCA62C1D6 (60 <= t <= 79).

使用的函数

在SHA1中我们需要一系列的函数。每个函数ft (0 <= t <= 79)都操作32位字B，C，D并且产生32位字作为输出。ft(B,C,D)可以如下定义

ft(B,C,D) = (B AND C) or ((NOT B) AND D) (0 <= t <= 19)

ft(B,C,D) = B XOR C XOR D (20 <= t <= 39)

ft(B,C,D) = (B AND C) or (B AND D) or (C AND D) (40 <= t <= 59)

ft(B,C,D) = B XOR C XOR D (60 <= t <= 79)

计算消息摘要

必须使用进行了补位和补长度后的消息来计算消息摘要。计算需要两个缓冲区，每个都由5个32位的字组成，还需要一个80个32位字的缓冲区。第一个5个字的缓冲区被标识为A，B，C，D，E。第二个5个字的缓冲区被标识为H0, H1, H2, H3, H4

。80个字的缓冲区被标识为W0, W1,..., W79

另外还需要一个一个字的TEMP缓冲区。

为了产生消息摘要，在第3.2部分中定义的512位（16个字）的数据块M1, M2,..., Mn

会依次进行处理，处理每个数据块Mi 包含80个步骤。

在处理所有数据块之前，缓冲区{Hi} 被初始化为下面的值（16进制）

H0 = 0x67452301

H1 = 0xEFCDAB89

H2 = 0x98BADCFE

H3 = 0x10325476

H4 = 0xC3D2E1F0.

现在开始处理M1, M2, ... , Mn。为了处理 Mi,需要进行下面的步骤

(1). 将 Mi 分成 16 个字 W0, W1, ... , W15, W0 是最左边的字

(2). 对于 t = 16 到 79 令

W[t] = S1(W[t-3] XOR W[t-8] XOR W[t-14] XOR W[t-16]).

(3). 令 A = H0, B = H1, C = H2, D = H3, E = H4.

(4) 对于 t = 0 到 79，执行下面的循环

TEMP = S5(A) + ft(B,C,D) + E + Wt + Kt;

E = D; D = C; C = S30(B); B = A; A = TEMP;

(5). 令 H0 = H0 + A, H1 = H1 + B, H2 = H2 + C, H3 = H3 + D, H4 = H4 + E.

在处理完所有的 Mn, 后，消息摘要是一个160位的字符串，以下面的顺序标识

H0 H1 H2 H3 H4.

how to implete

在本函数中通过调用linux下自带的shell命令 `sha1sum filename` 会返回文件的sha1值，通过函数shellcommand会返回shell命令的标准输出，从而获得文件的sha1值。