Introduction to Cryptography

Implementation of SHA3

(Deadline: 5/24 23:59:59)

In this assignment, you are asked to implement SHA3, please follow the detail spec below to implement. Once your SHA3 is done, use it to do bitcoin mining and report your result.

■ I/O format

The Input
An arbitrary length of string with no escape sequences.
Usage: Your program must be run by using

./YourProgram [M = Input string] [L = output length in bit]

The Output
Print the binary string in hexadecimal on screen of “L-bit”. (For convenience, L will be a multiple of 4 in test cases.)
No any other escape symbol in the output. (Letter must be in Uppercase, any lowercase letter will be considered wrong answer)

■ Sample I/O

Sample Input 1
./SHA3 HelloWorld 512

Sample Output 1
8051E75A0819AFED889A9812C3713380684DAF07C4F073C63EAC7A8
63B02343DE0C20FECD248CB5031A6230F21602D8DE9972D1BA010D8
CE294784ED91AB38C8

Sample Input 2
./SHA3 ILoveCryptography 2000

Sample Output 2
Sample Input
3
./SHA3
Atreeconstructedbyhashingpaireddatatheleavesthenpairingandhashing theresu 800

Sample Output
D4EF2F8C28B17F67935B564750FDA6D2F5751BA8DD66D768D81FD4
B48C977392B5EB1551548BBAB844A9304C4A7944F66F0ADF6747B98
9FE2D0B260A968663403E8B5FE880D8DAD50A7F1A2F50197A77DA14
5BC4D05004D4E4877680051E988EF9E489

Sample Input
4
./SHA3
AjudgeinBrazilorderedonMondaythatphonecompaniesblockFacebooks
telecoupleto cooperateinapolicieasistanceJudgeMarcelMontalvaodeliveredtheorderbecauseWhatsAppwould notturnover information that could relatetoaninvesigationintonarcoticsactivityand aninterstategangThe
countdownat 2pmlocaltimeBloombergreportedthefineto top honecompanies for not blocking the app wouldbe about 143000 perday Rounghalf of Brazilss population of 200million use the messagesasaneconomical alternativeto other pricier forms of communications the shut dow nwill have a big effect on many Brazilians This is the sec ond time in six month s a judge has suspended the service in Brazil though a court overturned the previous decision in December 3000
Requirement

1. Write your code in C or C++

2. Your assignment must obey the following requirements:
   (a). Name a folder as “0xxxxxx” (your student ID).
   (b). The folder must contain a Makefile and your source code.
      (Put the source code in this folder directly, not to package them into another folder)
   (c). Zip your folder as “0xxxxxx.zip”

※ A valid assignment content may look like the following figure

(These are important since we use shell script to grade your assignment, so if you didn’t follow the rule, you may not pass the judgement and your score will be deducted)
3. Upload your zip file to e3 system before deadline.

   Late submission in 1 week: your score = 0.7 * original score
   Late submission in 2 weeks: your score = 0.5 * original score
   Late submission over 2 weeks: your score = 0

4. DO NOT Plagiarize! We’ll use e3 judgement system to check your code, the one who is plagiarized and the one who plagiarizes will both get 0 on this assignment.

5. We will judge your code on CS linux workstation, make sure your code performs well on the platform.

■ Grading policy

(a). Pass 4 sample test cases: 80%

   (You’ll get 0 in this part if you hard-coded the sample)

(b). Bitcoin mining : 20%

   If your SHA-3 passes all the test cases, this part can be counted:
   Use your identity number(remove the first letter) + student ID as an initial string (for example: 1234567890212345 ) to your SHA-3, then continue to hash until the output value contains 6 continuous 0’s in front of the output. (ex: 000000363EBAF....), that is, 24 continuous bit 0.
   You can simply increase your initial value each time as the new input or any other ways you like for mining.
   (ex: 1234567890212345 -> 1234567890212346 -> 1234567890212347 -> 1234567890212348...)
   When the desired output is found, print screen your result and record the corresponding input and your mining time (how much time, and the number of times you spent until you find) as a report and upload your report in PDF format in your zip file.

   In summary, your report must contain the followings:
1. Print screen of the result
2. The input value you use to hash out
3. How much time (secs or mins or hours ...) you spent
4. How many times you hashed ( ex: 16777217 hashes )

Note that you CANNOT run this mining on CS station, or you may get suspended, if you want to do this part, just run it on your own PC or NB.

Spec of SHA-3

**Input:**
A message with arbitrary length, you have to partition it into blocks of 576 bits.

**Padding algorithm:**
Implement **Multirate Padding** here (append **10*1 as padding**), if the message length mod 576 equals 0, a padding block of 576 bits is also added.

**Sponge construction:**
- **Bitrate** \( r \): 576 bits
- **Capacity** \( c \): 1024 bits
- **State variable** \( s \): length \( b = r + c = 1600 \) bits, which is initialized to all zeros.

**Absorbing Phase:**
- **Input block** \( X_i \):
  
  \( P_i || 0^* \), which means \( P_i \) is padded with zeroes to extend its length to 1600 bits for \( i = 0 \) to \( k-1 \), where \( k \) is the number of the total blocks

**Squeezing Phase:**
- **Output:** Your program has to be able to generate outputs of any length of bits. (The number of bits will always be a multiple of 4 in test cases.)

**Notification**
We’ve provided a simple skeleton C++ source code of the construction of SHA3, you may just implement the /**TODO**/ part given in the source code, in summary, you have to implement the following functions:

1. The **Padding** algorithm in the `Sponge.cpp`
2. The **Absorbing phase** in the `Sponge.cpp`
3. The **Squeezing phase** in the `Sponge.cpp`
4. The **Internal function** of SHA3 in the `InternalFun.cpp`

If you’re not satisfied with TA’s code skeleton, you’re of course welcomed to implement the whole program by yourself, just make sure your code follows the requirements mentioned above.

We’ll open a discussion topic on e3 system. If there’s any difficulty during your implementation, feel free to ask questions on the discussion forum, we’ll help you with your problems, thanks!