One Way Hash

Lucas Larsson

April 2022

1 Generating Message Digest and MAC

Question 1. Describe your observations. What differences do you see between the algorithms?

Answer 1. The algorithms generate different hashes for the same input. Two main differences are that the hashes are not even little similar considering they have the same input, and the second is that they are of different lengths of 128, 160 and 256 bits respectively for MD5, SHA-1 and SHA-256.

.

Question 2. Write down the digests generated using the three algorithms.
Answer 2.

```
openssl dgst -md5 mail.txt
    MD5(mail.txt)=  5a14ecef080eb96ddb98df80a825e1ea

openssl dgst -sha1 mail.txt
    SHA1(mail.txt)= 3c3eafc8cf5967dd3fad26fa76bd123f8d77528b

openssl dgst -sha256 mail.txt
    SHA2-256(mail.txt)= 49022d321df0bb6c26d155e6454720a
    c65629ac06e77675f7666b1cc299a5900
```

2 Keyed Hash and HMAC

Question 3. Do we have to use a key with a fixed size in HMAC? If so, what is the key size?If not, why?

Answer 3. No, the HMAC key can be of any length, this is due to the Hash functions, an explanation from Wikipedia: "cryptographic hash functions [map] data of arbitrary size to a bit string of a fixed size (a hash)". So the key that you input, your password, can be of any size. Upon further analysis this is due to the functions using the "Modules" operator which is great for this specific use of mapping arbitrary lengths to a smaller set of bits.

Question 4. Now use the string "IV1013-key" as the secret key and write down the keyed hashes generated using the three algorithms.

Answer 4.

```
# Hash function with a key using HMAC.
        hmac = hash(
            (key xor opad) +
            hash((key xor ipad) xor message)
        )
```

```
openssl dgst -md5 -hmac "IV1013-key" mail.txt
    HMAC-MD5(mail.txt)= d1c9b22e0227a34d62a5465f4b15a484
```

```
openssl dgst -sha1 -hmac "IV1013-key" mail.txt
    HMAC-SHA1(mail.txt)= 7d0b9edb9085063ec5e713b25205507af3ed4f2e
```

```
openssl dgst -sha256 -hmac "IV1013-key" mail.txt
    HMAC-SHA2-256(mail.txt)= 798d3a1207af77193d700982e5d309802bcde
    f43b1ed81c8c6248442885b0623
```

3 The Randomness of One-way Hash

To understand the properties of one-way hash functions, we would like to do the following exercise for MD5 and SHA256:

1. Generate the hash value H1 for the file created in Section 2.1.
2. Flip the first bit of the input file (e.g. 01100001 → 11100001). You can achieve this modification using a binary editor such as ghex or Bless.
3. Generate the hash value H2 for the modified file.
4. How similar are H1 and H2?

Question 5. Describe your observations. Count how many bits are the same between H1 and H2 for MD5 and SHA256 (writing a short program to count the same bits might help you). In the report, specify how many bits are the same.

Answer 5.

```
for "lulars@kth.se"
```

```
openssl dgst -md5 mail.txt
    MD5(mail.txt)=  5a14ecef080eb96ddb98df80a825e1ea
```

```
openssl dgst -sha256 mail.txt
    SHA2-256(mail.txt)= 49022d321df0bb6c26d155e6454720ac656
    29ac06e77675f7666b1cc299a5900
```

```
for modified "lulars@kth.se" just flipping the first bit from 0 -> 1

    | openssl dgst -md5 mail.txt
        MD5(6.txt)= 71c3ff95086adc39644d70462342c048

    | openssl dgst -sha256 mail.txt
        SHA2-256(mail.txt)= b41c4aed8a83a17d92623b10a4cdf0dad
        a9c4bff9ac694325868d26614a351f1
```

A5... Not similar at all, I wrote a Java program to check the similar bits which can be found on my git repository: here

A print-out of the program:

```
 MD5
[64] different bits of total [128]

SHA-256
[147] different bits of total [256]
```

4 Collision Resistance

Question 6. Investigate how many trials it will take to break the weak collision property using the brute- force method. Below is a list of five messages. For each message, report how many trials it took before you could find a message with the same hash.

Answer 6.

- IV1013 security 26.370.363
- Security is fun 896.704
- Yes, indeed 24.145.654
- Secure IV1013 682.046
- No way 3.921.472

One Thing to notice here is that these are the tries I got when running the algorithm, and then when comparing to other students working on the same task I noticed that I had new line characters in some input strings, after removing them I rerun the program and got totally different statistics on the number of trails, the reason is obvious after having solved Question 5, one bit can have a huge difference, just an example is trying to crack the hash for the string "Security is fun" went from almost 58 million tries to less than 1 million.

```
     Version 1 :

Digest for the message "IV1013 security", using SHA-256 is:
9c24cd7aa776ea2182fb53657ad98f955d0d68a6ae4f65fafee25b800d9f6085
Cracking the Password.............

It took [26370363] trials to generate an identical digest

The brute-forced digest is:    0x9c24cd
------------------------------------------------------------
Digest for the message "Security is fun", using SHA-256 is:
41e772983176d9f6cfe73eb75d6d7cbee865fe99d6b0ef2c353a196469c28661
Cracking the Password............................

It took [57351479] trials to generate an identical digest

The brute-forced digest is:    0x41e772
------------------------------------------------------------
Digest for the message "Yes, indeed", using SHA-256 is:
25d378775a5542250a748c8b61cd83c8e882d2cfee851e14b484f0a17c4406d2
Cracking the Password......

It took [12771851] trials to generate an identical digest

The brute-forced digest is:    0x25d378
------------------------------------------------------------
Digest for the message "Secure IV1013
", using SHA-256 is:
0364e53861d71e82239fcbbd977ca0e04172bc455037e35b0177fe1a183620e0
Cracking the Password........

It took [17255705] trials to generate an identical digest

The brute-forced digest is:    0x0364e5
------------------------------------------------------------
Digest for the message "No way
", using SHA-256 is:
95bd6e0b8e223268fc03ba2a5cded2f70ae79b5e523d2a3951a9cc9b81e5f582
Cracking the Password..........

It took [21321165] trials to generate an identical digest

The brute-forced digest is:    0x95bd6e
------------------------------------------------------------

Process finished with exit code 0
```

```
------------------------------------------------------------------------
Version 2
    Digest for the message "IV1013 security", using SHA-256 is:
    9c24cd7aa776ea2182fb53657ad98f955d0d68a6ae4f65fafee25b800d9f6085
    Cracking the Password.............

    It took [26370363] trials to generate an identical digest

    The brute-forced digest is:    0x9c24cd
    ------------------------------------------------------------
    Digest for the message "Security is fun", using SHA-256 is:
    ef9fa63795079c192599ea691ab84624ae7ddbd01f893832ce9efbd811534b75
    Cracking the Password

    It took [896704] trials to generate an identical digest

    The brute-forced digest is:    0xef9fa6
    ------------------------------------------------------------
    Digest for the message "Yes, indeed", using SHA-256 is:
    0a1634cb0209131d28b3332898c4cc284ebf3795f6a0f8750a173ff45ec201f6
    Cracking the Password.............

    It took [24145654] trials to generate an identical digest

    The brute-forced digest is:    0x0a1634
    ------------------------------------------------------------
    Digest for the message "Secure IV1013", using SHA-256 is:
    4cee4916951f8851394770c655e179aafe517614e3ab4ad2e37316973a91c39b
    Cracking the Password

    It took [682046] trials to generate an identical digest

    The brute-forced digest is:    0x4cee49
    ------------------------------------------------------------
    Digest for the message "No way", using SHA-256 is:
    cb82f5c1f4f70f68c92466ef2ec5fe7fa921dc97328ae97a1f96a3fe676cffa0
    Cracking the Password.

    It took [3921472] trials to generate an identical digest

    The brute-forced digest is:    0xcb82f5
    ------------------------------------------------------------

    Process finished with exit code 0
```