Khoa Khoa học & Kỹ thuật máy tính

Trường ĐH Bách Khoa TP.HCM

Cryptography and Network Security

Lab 4

Hash Functions

INTRODUCTION

This Lab will introduce students to hash functions and how they provide for message integrity by using CrypTool software. Students will be asked to use hashing to detect if an ecrypted message has been tampered with. Students will also need to show that this integrity check can be bypassed by tampering with both the ciphertext and the hashcode.

Install CrypTool 1 (version: 1.4.31 Beta 06 - English) by going to:
https://www.cryptool.org/en/ct1-downloads

EXPERIENCE

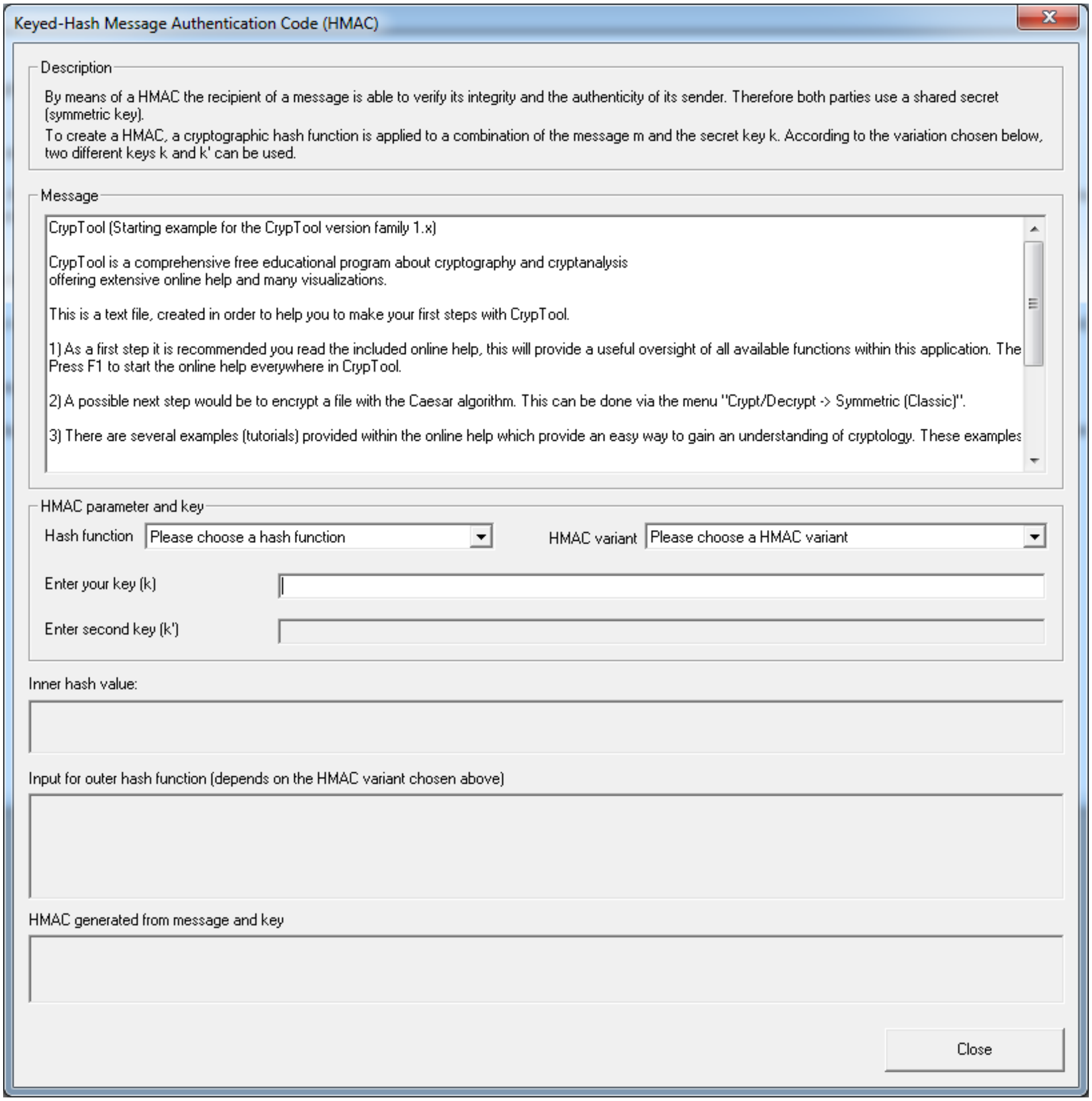
Lab on hash generation and sensitivity of hash functions to plaintext modifications

Keyed-Hash Message Authentication Code (HMAC) ensures integrity of a message and authentication of the message. It requires a common key for sender and recipient.

1. Open the file **CrypTool-en.txt** under:

   C:\Program Files (x86)\CrypTool\examples.

2. Select from Menu: “**Indiv. Procedures” \“Hash” \“Generation of HMACs**”.



Keyed-Hash Message Authentication Code (HMAC)

Description

By means of a HMAC the recipient of a message is able to verify its integrity and the authenticity of its sender. Therefore both parties use a shared secret (symmetric key).
To create a HMAC, a cryptographic hash function is applied to a combination of the message m and the secret key k. According to the variation chosen below, two different keys k and k' can be used.

Message

CrypTool (Starting example for the CrypTool version family 1.x)

CrypTool is a comprehensive free educational program about cryptography and cryptanalysis
offering extensive online help and many visualizations.

This is a text file, created in order to help you to make your first steps with CrypTool.

1) As a first step it is recommended you read the included online help, this will provide a useful oversight of all available functions within this application. The
Press F1 to start the online help everywhere in CrypTool.

2) A possible next step would be to encrypt a file with the Caesar algorithm. This can be done via the menu "Crypt/Decrypt -> Symmetric (Classic)".

3) There are several examples (tutorials) provided within the online help which provide an easy way to gain an understanding of cryptology. These examples

HMAC parameter and key

Hash function: Please choose a hash function

HMAC variant: Please choose a HMAC variant

Enter your key (k)

Enter second key (k')

Inner hash value:

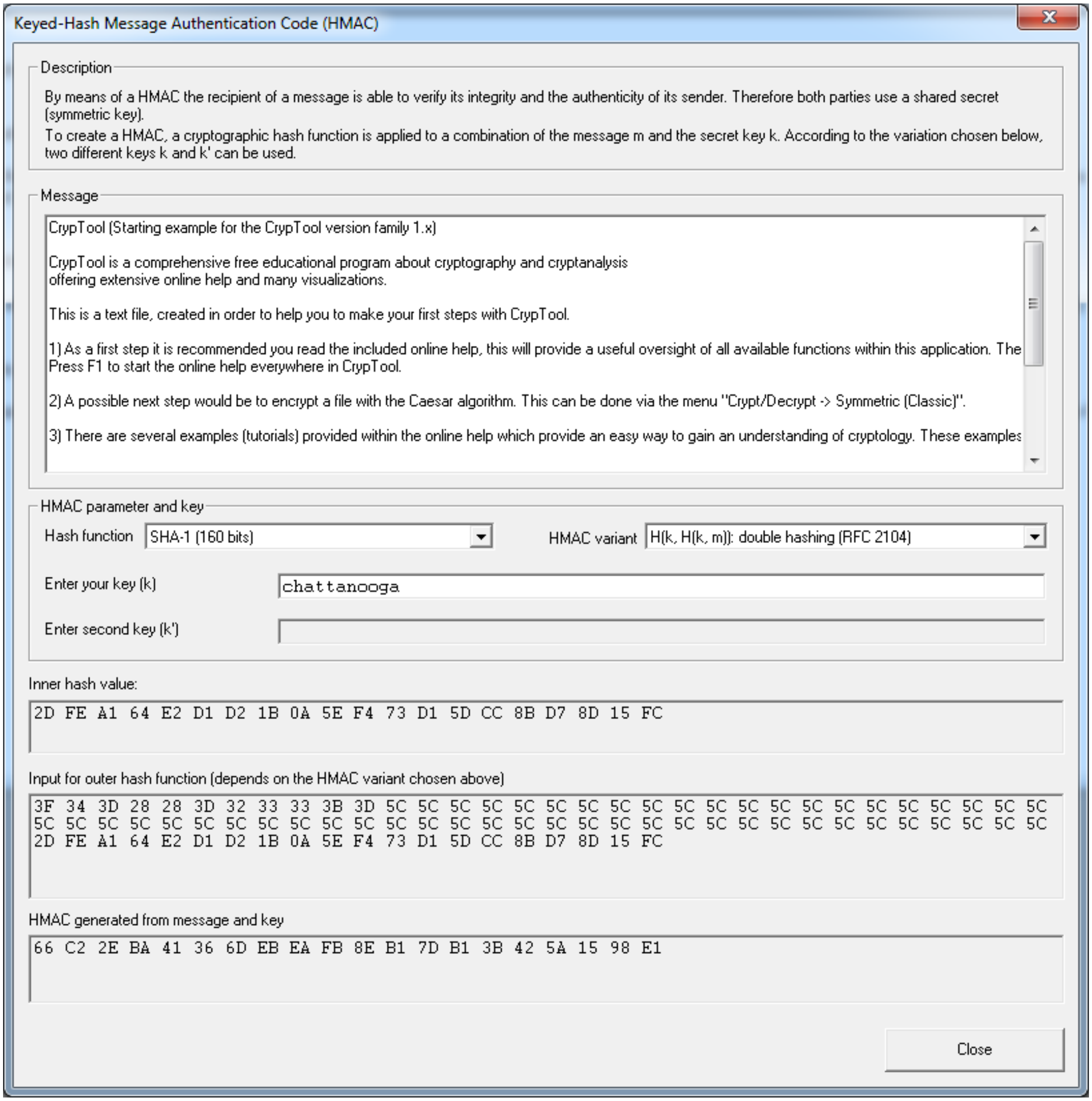
Input for outer hash function (depends on the HMAC variant chosen above)

HMAC generated from message and key

3. Select SHA-1 as hash function and double hashing as HMAC variants.

4. Enter your key “chattanooga”. The HMAC code generated from the message and the key is:

66 C2 2E BA 41 36 6D EB EA FB 8E B1 7D B1 3B 42 5A 15 98 E1



Keyed-Hash Message Authentication Code (HMAC)

Description

By means of a HMAC the recipient of a message is able to verify its integrity and the authenticity of its sender. Therefore both parties use a shared secret (symmetric key).
To create a HMAC, a cryptographic hash function is applied to a combination of the message m and the secret key k. According to the variation chosen below, two different keys k and k' can be used.

Message

CrypTool (Starting example for the CrypTool version family 1.x)

CrypTool is a comprehensive free educational program about cryptography and cryptanalysis
offering extensive online help and many visualizations.

This is a text file, created in order to help you to make your first steps with CrypTool.

1) As a first step it is recommended you read the included online help, this will provide a useful oversight of all available functions within this application. The
Press F1 to start the online help everywhere in CrypTool.

2) A possible next step would be to encrypt a file with the Caesar algorithm. This can be done via the menu "Crypt/Decrypt -> Symmetric (Classic)".

3) There are several examples (tutorials) provided within the online help which provide an easy way to gain an understanding of cryptology. These examples

HMAC parameter and key

Hash function: SHA-1 (160 bits)

HMAC variant: H(k, H(k, m)): double hashing (RFC 2104)

Enter your key (k): chattanooga

Enter second key (k'):

Inner hash value:

```
2D FE A1 64 E2 D1 D2 1B 0A 5E F4 73 D1 5D CC 8B D7 8D 15 FC
```

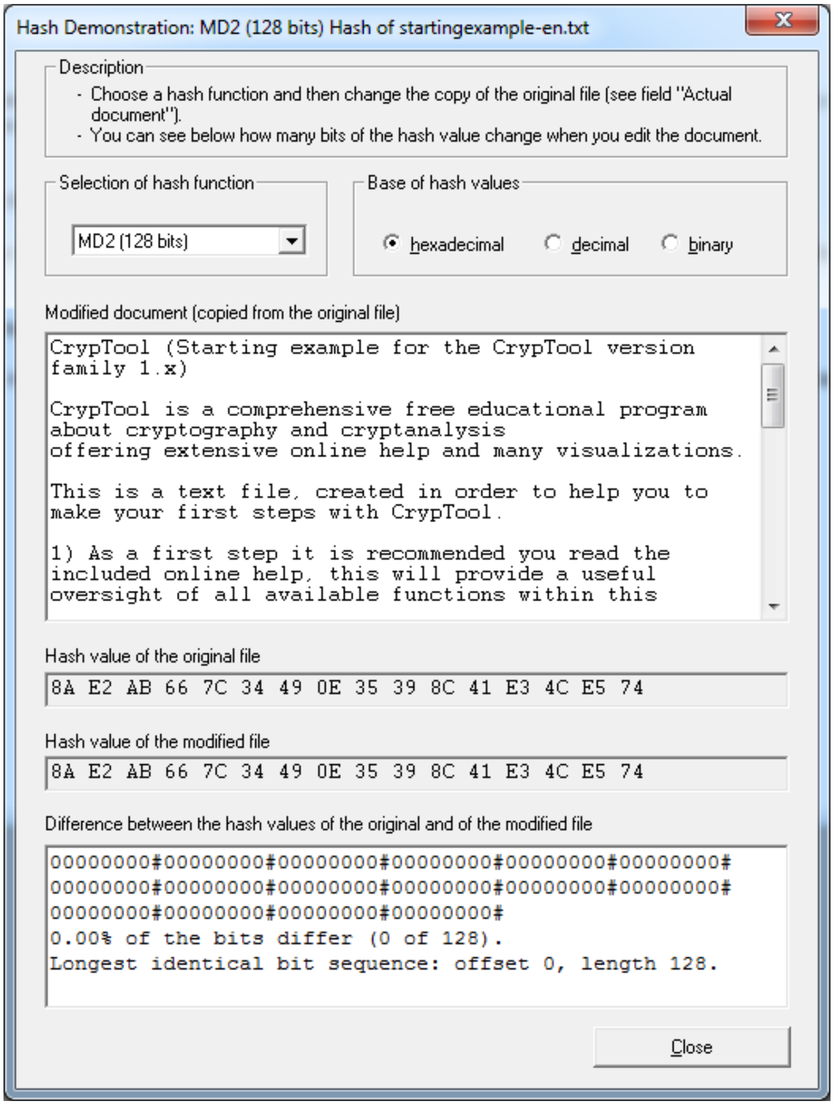
Input for outer hash function (depends on the HMAC variant chosen above)

```
3F 34 3D 28 28 3D 32 33 33 3B 3D 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C
5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C 5C
2D FE A1 64 E2 D1 D2 1B 0A 5E F4 73 D1 5D CC 8B D7 8D 15 FC
```

HMAC generated from message and key

```
66 C2 2E BA 41 36 6D EB EA FB 8E B1 7D B1 3B 42 5A 15 98 E1
```

5. Select from menu “**Indiv. Procedures” \“Hash” \“Hash Demonstration**”.



Hash Demonstration: MD2 (128 bits) Hash of startingexample-en.txt

Description

- Choose a hash function and then change the copy of the original file (see field "Actual document").
- You can see below how many bits of the hash value change when you edit the document.

Selection of hash function: MD2 (128 bits)

Base of hash values: ● hexadecimal ○ decimal ○ binary

Modified document (copied from the original file)

```
CrypTool (Starting example for the CrypTool version
family 1.x)

CrypTool is a comprehensive free educational program
about cryptography and cryptanalysis
offering extensive online help and many visualizations.

This is a text file, created in order to help you to
make your first steps with CrypTool.

1) As a first step it is recommended you read the
included online help, this will provide a useful
oversight of all available functions within this
```

Hash value of the original file

```
8A E2 AB 66 7C 34 49 0E 35 39 8C 41 E3 4C E5 74
```

Hash value of the modified file

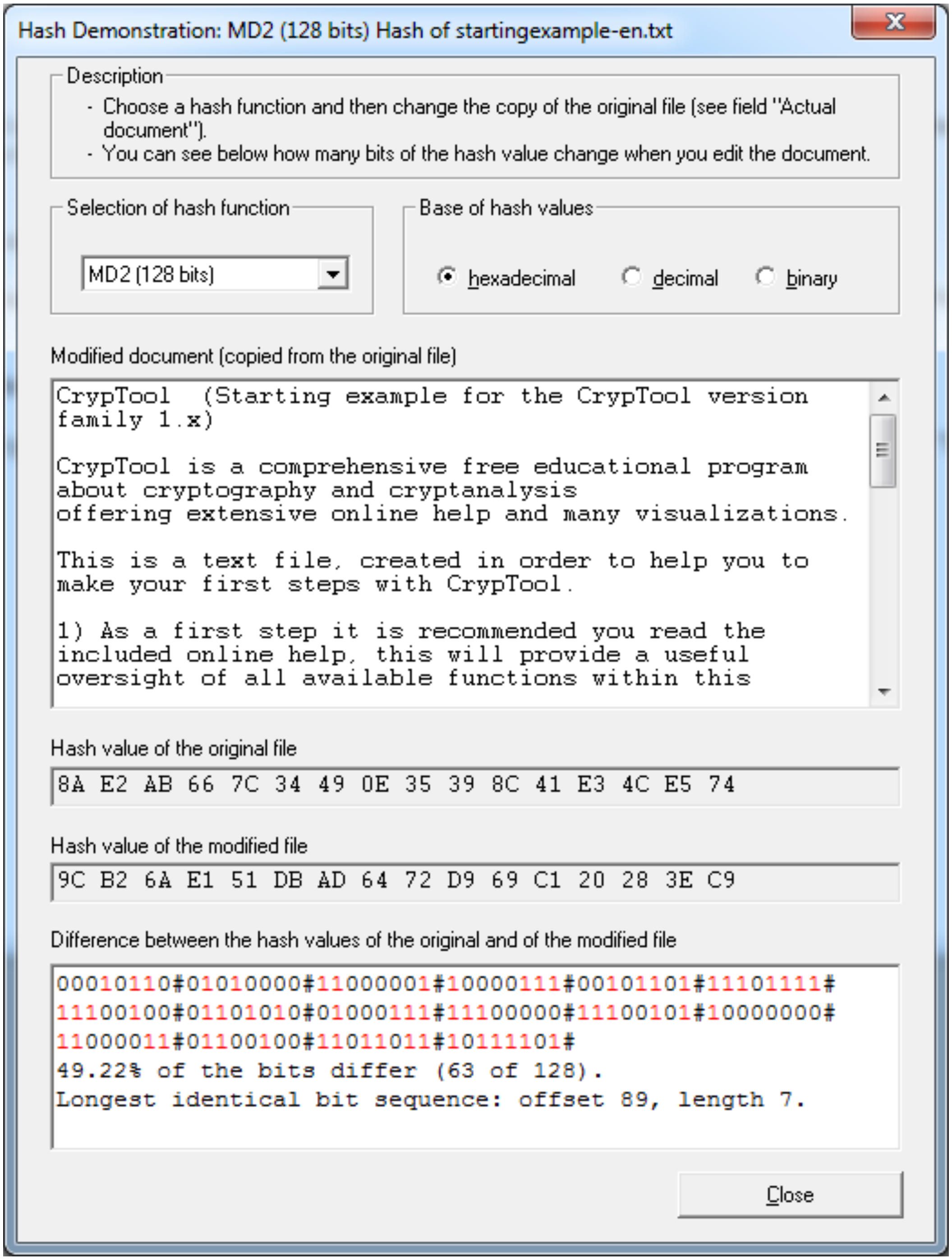
```
8A E2 AB 66 7C 34 49 0E 35 39 8C 41 E3 4C E5 74
```

Difference between the hash values of the original and of the modified file

```
00000000#00000000#00000000#00000000#00000000#00000000#
00000000#00000000#00000000#00000000#00000000#00000000#
00000000#00000000#00000000#00000000#
0.00% of the bits differ (0 of 128).
Longest identical bit sequence: offset 0, length 128.
```

6. Select a hash function from Selection of hash function.

7. add a space after CrypTool in plaintext. We will see **49.22%** bits differ (63 of 128). A good hash function should react highly sensitively to even the smallest change in the plaintext –“Avalanche effect” (small change, big impact).



Hash Demonstration: MD2 (128 bits) Hash of startingexample-en.txt

Description

- Choose a hash function and then change the copy of the original file (see field "Actual document").
- You can see below how many bits of the hash value change when you edit the document.

Selection of hash function: MD2 (128 bits)

Base of hash values: ● hexadecimal ○ decimal ○ binary

Modified document (copied from the original file)

```
CrypTool  (Starting example for the CrypTool version
family 1.x)

CrypTool is a comprehensive free educational program
about cryptography and cryptanalysis
offering extensive online help and many visualizations.

This is a text file, created in order to help you to
make your first steps with CrypTool.

1) As a first step it is recommended you read the
included online help, this will provide a useful
oversight of all available functions within this
```

Hash value of the original file

```
8A E2 AB 66 7C 34 49 0E 35 39 8C 41 E3 4C E5 74
```

Hash value of the modified file

```
9C B2 6A E1 51 DB AD 64 72 D9 69 C1 20 28 3E C9
```

Difference between the hash values of the original and of the modified file

```
00010110#01010000#11000001#10000111#00101101#11101111#
11100100#01101010#01000111#11100000#11100101#10000000#
11000011#01100100#11011011#10111101#
49.22% of the bits differ (63 of 128).
Longest identical bit sequence: offset 89, length 7.
```

HOMEWORK

Exercise 1. Use an example to show that Hash function can help to protect integrity of your message. You can encrypt your plaintext message, tamper the cipher text and use hash function to check whether the decrypted messaged is changed.

Exercise 2. Hash function H(·) generates a representative compact fingerprint (a hash value) of a given piece of information. As such, H(·) has to satisfy the following properties: H(·) is applicable to messages of arbitrary (finite) size, it generates a hash value of a fixed size, and it can be easily (quickly) generated for any message.

a. Consider all possible 200 bit long inputs to hash function H(·). Assume that H(·) outputs 160 bit hash values. How many input values, on average, hash to each possible output value?

(Hint: Model H(·) as follows. For a given n bit hash value H(x), the probability that the hash value H(y) of a randomly chosen message (preimage) y, equals H(x), is approx- imately 2^{-n}.)

b. Consider the following hash value obtained by hashing with SHA-1 a single letter of English alphabet: C6 3A E6 DD 4F C9 F9 DD A6 69 70 E8 27 D1 3F 7C 73 FE 84 1C. Find the corresponding letter. Describe your approach. Use CrypTool to accomplish this task.

c. Assume that you succeed in the previous task (you recover the hashed letter). Does you success imply that that SHA-1 hash function does not satisfy one-way property? Explain your answer.

d. Create a new document in CrypTool by clicking on the icon “New”. Write some text in the new document. In the main menu, under “Indiv. Procedures” submenu select “Hash → Hash Demonstration...” to open “Hash demo” window. Modify text that ap- pears in “Actual document” window and observe what happens with the corresponding hash value. Explain your observation.