Université Libre de Bruxelles
Faculté des Sciences

2024-2025

INFO-F-405: Introduction to cryptography

3. Hashing

Definitions and requirements

Exercise 1

For a given hash function, does second preimage resistance imply collision resistance, or vice-versa, and why?

Exercise 2

Choose a standard cryptographic hash function (or XOF) arbitrarily. Write a small program that computes the digest of a given string truncated to n bytes. For a given value n, compute the digest of arbitrary (but distinct) input strings, and look for a collision at the output.

1. Start with $n = 1$, so $\ell = 8$ bits of output. After how many iterations t do you get a collision?
2. Increase n until this starts to take too much time. Do you recognize the law that gives you $t(n)$?

Exercise 3

Use the previously written piece of code and do the following test. Choose an arbitrary message m and compute $H = \text{hash}(m)$. Then, flip the first bit of m in order to obtain m'. Can you predict the value of $H' = \text{hash}(m')$ from that of H without explicitly computing it?

Repeat the experiment with different values of m and/or flipping bits at different positions. Can you see a predictable relationship between the corresponding H and H'? What is the average number of differences (i.e., the Hamming distance) between H and H'?

Exercise 4

A friend of yours designed his/her own hash function CATSHA320, with 320 bits of output. It is an iterated hash function that cuts the input message into blocks of 192 bits, processes them sequentially and has a chaining value of 224 bits. Without knowing more about this hash function, what is its expected collision resistance (in number of attempts), and why?

Same question for CATSHA320++ with the same specifications but a block size of 384 bits and a chaining value of 480 bits.

Zooming from Merkle-Damgård into SHA-1

Exercise 5

[1] The core of some hash functions is the compression function. Let assume that we build a compression function on top of the block cipher

$$\text{output block} = \text{AES}_{\text{key}}(\text{input block})$$

in one of the following ways:

- $f_1(x, y) = \text{AES}_x(y) \oplus x$
- $f_2(x, y) = \text{AES}_x(x) \oplus y$

We ask you to find collisions for f_1 and for f_2. In other words you should find values a_1, b_1 and a_2, b_2 such that $f_1(a_1, b_1) = f_1(a_2, b_2)$ and also find values c_1, d_1 and c_2, d_2 such that $f_2(c_1, d_1) = f_2(c_2, d_2)$.

Exercise 6

Many well-known hash functions, such as MD5, SHA-1 and the SHA-2 family, use a block cipher in the Davies-Meyer construction. However, these are all specific, dedicated, block ciphers. What about the AES? So let us build SHAES by first building a compression function from the AES-256 in the Davies-Meyer construction, then putting the compression function in the Merkle-Damgård construction.

1. What is the block size of SHAES?

[1] This exercise was inspired by an exercise from the Security course given by Dan Boney.

2. What is the chaining value size of SHAES?

3. Is SHAES a valid solution in practice?

Exercise 7

Some designs, like the hash function SHA-1 and the SHA-2 family, use a combination of modular addition (in $\mathbb{Z}_{2^n}$) and bitwise addition or XOR (in $\mathbb{Z}_2^n$), plus rotations (or circular shifts). In the litterature, this combination is often referred to "ARX" for addition-rotation-XOR. The idea is that the modular addition is non-linear from the point of view of $\mathbb{Z}_2^n$, and vice-versa, the XOR is non-linear in $\mathbb{Z}_{2^n}$.

Formally, an element $x = (x_0, x_1, \ldots, x_{n-1}) \in \mathbb{Z}_2^n$ is mapped to the integer $X = \sum_i x_i 2^i$ in $\mathbb{Z}_{2^n}$. Let us define $X = \mathsf{integer}(x)$ as the function that converts an element of $\mathbb{Z}_2^n$ to an element of $\mathbb{Z}_{2^n}$, and let $x = \mathsf{vector}(X)$ be its inverse.

1. Show that $f(x, y) = \mathsf{vector}(\mathsf{integer}(x) + \mathsf{integer}(y))$, i.e., the modular addition of x and y interpreted as integers, is non-linear in $\mathbb{Z}_2^n$. *Hint:* It is sufficient to give a counter-example. Also, you may set $n = 2$ for simplicity.

2. Show that $F(X, Y) = \mathsf{integer}(\mathsf{vector}(X) \oplus \mathsf{vector}(Y))$, i.e., the XOR of X and Y interpreted as n-bit vectors, is non-linear in $\mathbb{Z}_{2^n}$.

Modern generic security

Exercise 8

Let $\mathcal{RO}(x)$ be a random oracle that outputs an infinite sequence of uniformly and independently distributed bits for each input $x \in \{0,1\}^*$. If the random oracle is queried twice with the same input, it always returns the same sequence. We denote with $\mathcal{RO}(x, \ell)$ the output of $\mathcal{RO}(x)$ truncated after the first ℓ output bits, so $\mathcal{RO}(x, \ell) \in \{0,1\}^\ell$.

1. *Preimage attack.* Given some value $y \in \{0,1\}^\ell$, an adversary would like to find a preimage, i.e., an input x such that $\mathcal{RO}(x, \ell) = y$. What is the probability of success after t attempts? (We can approximate assuming that $t \ll 2^\ell$.)

2. *Multi-target preimage attack.* Given a set of m distinct values $\{y_1, \ldots, y_m\}$, with $y_i \in \{0,1\}^\ell$, an adversary would be happy to find a preimage of any one of these images. What is the probability of success after t attempts?

Let $h(x)$ be a concrete extendable output function (XOF) that outputs a potentially infinite sequence of bits, and we similarly denote $h(x, \ell)$ its truncation to ℓ output bits. As of today, the only known attack against $h(x)$ has a probability of success $t/2^{c/2}$, for some security parameter c, and where t is the time spent by the adversary expressed in the number of attempts. This means that, except for this probability $\epsilon(t)$, we can model $h(x)$ as a random oracle.

3. What is the range of output sizes ℓ and parameter values c such that $h(x)$ offers 128 bits of preimage resistance?

4. What is the range of output sizes ℓ and parameter values c such that $h(x)$ offers 128 bits of multi-target preimage resistance with $m = 2^{32}$?

You may use the approximation $\log(a + b) \approx \max(\log a, \log b)$.

Inside SHA-3

Exercise 9

Let π be a cryptographically secure permutation that maps $b = 1600$-bit strings onto b-bit strings. We assume that π and its inverse π^{-1} are both equally efficiently implementable. We build a hash function by applying the sponge construction on top of π. The hash function has a fixed output length of $n = 256$ bits. However, for the sake of this question, we set the capacity to $c = 0$.

1. Please explain how to obtain a collision with this hash function.

2. Given two prefixes x and y of b bits each, show how to obtain a collision between one input that must start with x and the other one with y.

3. Given an output value z of n bits, describe how to find a preimage with this hash function.

4. Now let us assume that this hash function is used exclusively in an application that hashes input strings made of ASCII-encoded text, for which each byte has its most significant bit always set to 0. More precisely, we restrict the input strings m such that every 8th bit is 0, and for the sake of simplicity this is the only restriction we consider (i.e., we do not care whether m is actually valid ASCII-encoded text or not). Please explain how to obtain a collision with this hash function such that the colliding inputs have this restriction. What is the complexity of your attack?

5. Given an output value z of n bits, describe how to find a preimage with this hash function such that the preimage has the aforementioned restriction. What is the complexity of your attack?

Exercise 10

Consider FIPS 202 and extendable output functions.

1. What is (approximately) the performance ratio between SHAKE128 and SHAKE256? And between SHAKE128 and SHA3-512?
2. For SHAKE128, what is the output size you need to choose to have 128-bit security for collision, preimage and second preimage resistance?
3. Same question, but only for preimage and second preimage resistance (no collision resistance required)?
4. Let us fix an ouput size ℓ. Given a set of m distinct values $\{y_1, \ldots, y_m\}$, with $y_i \in \{0,1\}^\ell$, an adversary would be happy to find a preimage of any one of these images. What is (approximately) the probability of success after t random attempts against a random oracle?
5. Say that $m = 2^{40}$. Can we use SHAKE128 to resist against multi-target attacks with a security level of 128 bits? If so, what would be the required output size ℓ?

Others

Exercise 11

How (not?) to build message authentication codes from hash functions.

Let K be a k-bit secret key (e.g., $k = 128$ bits).

1. Let us assume that we build a MAC function as follows:

$$\mathrm{MAC}_K(\text{message}) = \mathrm{hash}(\text{message}) \oplus K,$$

where $\mathrm{hash}(\cdot)$ is a secure k-bit hash function and $\oplus$ denotes the bitwise mod-2 addition (or XOR). Is this MAC function secure? If so, please justify briefly. If not, please describe an attack in the light of the EU-CMA game.

2. Let us consider another MAC function. We now assume that the MAC function is constructed as follows:

$$\mathrm{MAC}_K(\text{message}) = \mathrm{hash}(K_1 \| \text{message}) \oplus K,$$

where $\mathrm{hash}(\cdot)$ is a secure k-bit hash function, $\|$ denotes the concatenation and $K = K_1 \| K_2$ with $|K_1| = |K_2| = k/2$. What is the security strength of this MAC function, and why?