# Introduction to cryptology
# (GBIN8U16)
# Final Examination

2018-05-03

### Instructions

The duration of this examination is three hours. All exercises are independent, and they may be solved in any order. Answers to the questions must be detailed and complete to get maximum credit. The full scale is not determined yet: it may not be necessary to answer all questions in order to obtain a perfect mark.

### Exercise 1: Hash function security notions

In the following questions, $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ is a cryptographic hash function.

**Q. 1:** An attacker is given a message $m \in \{0,1\}^*$; his/her goal is to find a message $m' \neq m$ s.t. $\mathcal{H}(m) = \mathcal{H}(m')$.

1. What is the name given to this type of attack?
2. Assuming that $\mathcal{H}$ behaves like a random function, how many solutions exist to the problem (that is, what is the size of the set $\{m' \in \{0,1\}^* | \, \mathcal{H}(m) = \mathcal{H}(m')\}$?
3. Assuming that $\mathcal{H}$ behaves like a random function, how many evaluations of $\mathcal{H}$ are sufficient to find one solution with probability close to 1? How many are necessary, with high probability?
4. Give an example of hash function $\mathcal{H}$ (that does not necessarily behave as a random function) for which there is no solution for this attack for the message `0`.

**Q. 2:** An attacker wants to find $m$ and $m' \neq m$ s.t. $\mathcal{H}(m) = \mathcal{H}(m')$.

1. What is the name given to this type of attack?
2. Assuming that $\mathcal{H}$ behaves like a random function, how many evaluations of $\mathcal{H}$ are sufficient to find one solution with probability close to 1? How many are necessary, with high probability?

**Q. 3:** Given the current state of technology, which of the following values of $n$ offer sufficient protection against the above attacks?

1. 64, when one only wishes to be protected from the attack of **Q. 1**.
2. 64, when one only wishes to be protected from the attack of **Q. 2**.
3. 128, when one only wishes to be protected from the attack of **Q. 2**.

4. 256, when one only wishes to be protected from the attack of **Q. 1**.

5. 384, when one wishes to be protected from both attacks.

### Exercise 2: A random sequence

**Q. 1:** Let $\mathcal{S}$ be a set of size $N$; let $(u_n)_{n\in\mathbb{N}}$ be a sequence whose elements are drawn independently and uniformly at random from $\mathcal{S}$, i.e. for all $i$, $u_i \xleftarrow{\$} \mathcal{S}$. Suppose that you do not initially know $\mathcal{S}$, nor $N$.

1. Give an algorithm that takes as input a finite number of elements of $(u_n)$ and that returns an approximation of $N$.

2. What is the time and memory complexity of your algorithm?

### Exercise 3: Diffie-Hellman assumptions

In the following questions, $\mathbb{G} = \langle g \rangle$ is a finite group of order $N$ (i.e. with $N$ elements $g^0, \ldots, g^{N-1}$).

**Q. 1:** Let $h = g^a$, $0 \leq a < N$ be an element of $\mathbb{G}$.

1. What is the discrete logarithm of $h$ with respect to the generator $g$?

2. Assuming that $h$ is a generator of $\mathbb{G}$ (i.e. $\langle h \rangle = \mathbb{G}$), what is the discrete logarithm of $h$ with respect to the generator $h$?

The *discrete logarithm* (DLOG) assumption for $\mathbb{G}$ states that given $g$, $h = g^a$, with $a \xleftarrow{\$} [0, N[$, it is hard to compute the discrete logarithm of $h$ in base $g$.

**Q. 2:** The *computational Diffie-Hellman* (CDH) assumption states that given $g$, $g^a$, $g^b$, with $a \xleftarrow{\$} [0, N[$, $b \xleftarrow{\$} [0, N[$, it is hard to find $g^{ab}$. An attacker is said to *break* CDH if she/he is able to find $g^{ab}$.

1. Show that if one can compute discrete logarithms in base $g$ with complexity $L$ and an exponentiation in base $g$ with complexity $E$, then one can break CDH with complexity $\leq L + E$.

**Q. 3:** The *decisional Diffie-Hellman* (DDH) assumption states that given $g$, $g^a$, $g^b$, $g^x$, with $a \xleftarrow{\$} [0, N[$, $b \xleftarrow{\$} [0, N[$, it is hard to distinguish the two cases $x = ab$ or $x \xleftarrow{\$} [0, N[$. An attacker is said to *break* DDH if she/he is able to decide the value of $x$ with probability $\approx 1$.

1. Show that if one can break CDH with complexity $C$, one can break DDH with complexity $C$.

**Q. 4:** An assumption $A$ is said to be *stronger* than an assumption $B$ if breaking $B$ implies breaking $A$, but breaking $A$ does not necessarily imply breaking $B$.

1. Order the DDH, CDH and DLOG assumptions from weakest to strongest.

## Exercise 4: Baby-step/Giant-step over an interval

In the following questions, $\mathbb{G} = \langle g \rangle$ is a finite group of order $N$ (i.e. with $N$ elements $g^0, \ldots, g^{N-1}$).

We recall that the baby-step/giant-step algorithm may be used to solve a discrete-logarithm problem in $\mathbb{G}$ with respect to the generator $g$. By defining $\nu$ as $\left\lceil \sqrt{N} \right\rceil$, one precomputes the list $L_1 = [g^{i\nu};\ 0 \leq i \leq \nu]$, and solves the problem $g^a$ by computing $L_2 = [g^a g^i;\ 0 \leq i \leq \nu]$. A collision between the two lists gives the value of the discrete logarithm.

**Q. 1:**

1. What is the time and memory complexity of the baby-step/giant-step algorithm?

**Q. 2:** Suppose now that you must compute the discrete logarithm of $g^a$ and that you know that $a \in I = [\alpha, \beta] \subset [0, N[$, and that $\#I = \beta - \alpha + 1 \ll N$.

1. Explain in words how to modify the above algorithm to solve this specific discrete logarithm more efficiently. What is the resulting time and memory complexity?

2. Give a full pseudo-code of this modified algorithm.

## Exercise 5: Symmetric modes of operation

In the following questions, $\mathcal{E} : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. We suppose that $\mathcal{E}$ is a "good" cipher, in the sense that for every key $k$, $\mathcal{E}(k, \cdot)$ behaves like a random permutation.

**Q. 1:** In order to encrypt a message $m$ of more than $n$ bits with $\mathcal{E}$, one proposes to use the following mode: pad $m$ so that its length is equal to $l \times n$ for some $l$; write the resulting message as the concatenation $m_1 || \ldots || m_l$, with all the blocks $m_i$s being $n$-bit long; for all $i$, encrypt the block $m_i$ with the key $k$ and initialization vector $c_0$ as $c_i = \mathcal{E}(k, m_i \oplus c_{i-1})$.

1. What is the name of this mode?

2. Give the decryption procedure, that from $c_0 || \ldots || c_l$ and $k$ returns $m1 || \ldots || m_l$.

**Q. 2:** We recall (briefly) that a good mode of operation must be such that distinguishing the encryption of two messages $m$ and $m'$ of equal length is hard, while being given prior access to chosen-plaintext encryptions.

1. Is the mode of the previous question good if $c_0$ is set to a constant?

2. Is the mode of the previous question good if $c_0$ is implemented as a randomly initialized global counter? That is, the value of $c_0$ used to encrypt the $i^{\text{th}}$ message is set to $\mathsf{IV} + i \mod 2^n$, where the initial value of the counter $\mathsf{IV}$ is chosen uniformly at random (i.e. $\mathsf{IV} \xleftarrow{\$} \{0,1\}^n$).

3. Is the mode of the previous question good if $c_0$ is implemented as the encryption (with a key independent from the encryption key of the mode itself) of a global counter initialised to zero? That is, the value of $c_0$ used to encrypt the $i^{\text{th}}$ message is set to $\mathcal{E}(k', (i \mod 2^n))$, with $k' \xleftarrow{\$} \{0,1\}^\kappa$ a secret key.

**Q. 3:** One proposes a variant of the above mode, where the encryption of $m_1||\dots||m_l$ with the key $k$ and initialization vector $x_0$ is defined for all $i$ as $c_i = m_i \oplus x_i$; $x_i = \mathcal{E}(k, x_{i-1})$.

1. Give the decryption procedure for this mode.

2. Based on your knowledge of mode of operations, explain why this is a good mode if $x$ is implemented as a global variable initialized to zero for the first message and not reset between different messages. (For instance, this means that if one starts by encrypting the two two-block messages $m_1||m_2$ and $m'_1||m'_2$, one has $c'_2 = m'_2 \oplus \mathcal{E}^4(k, 0)$, with $\mathcal{E}^4(k, 0) = \mathcal{E}(k, \mathcal{E}(k, \mathcal{E}(k, \mathcal{E}(k, 0))))$.)

## Exercise 6: RSA-CRT

In the following questions, $N = pq$ for prime numbers $p$ and $q$; $e, d \in (\mathbb{Z}/\varphi(N)\mathbb{Z})^\times \backslash \{1\}$ such that $ed \equiv 1 \mod \varphi(N)$; $\mathcal{P} : \mathbb{Z}/N\mathbb{Z} \to \mathbb{Z}/N\mathbb{Z}$ is defined by $m \mapsto m^e \mod N$, and its inverse $\mathcal{P}^{-1}$ is defined by $m \mapsto m^d \mod N$.

**Q. 1:** Let $C_p = q \times (q^{-1} \mod p)$; $C_q = p \times (p^{-1} \mod q)$.

1. Compute the following: $C_p \mod p$; $C_p \mod q$; $C_q \mod p$; $C_q \mod q$.

**Q. 2:** Let $0 \leq x < N$ be such that $x \equiv x_p \mod p$; $x \equiv x_q \mod q$.

1. Using (implicitly) the Chinese Remainder Theorem (CRT), give the value of $x$ in function of $C_p$, $x_p$, $C_q$, $x_q$ and $N$.

**Q. 3:** A user wishes to implement $\mathcal{P}^{-1}$ by computing the exponentiation to $d$ using the CRT.

1. Explain (briefly) why if $\mathcal{P}$ and $\mathcal{P}^{-1}$ are used within an RSA cryptosystem, a CRT implementation may only be used by someone knowing the private key

2. Give the details of such an implementation, that first reduces the input mod $p$ and mod $q$, performs the exponentiation over these residues, and recombines the result using the CRT, and justify its correctness.

**Q. 4:** We now want to show that if a *single fault* occurs during the CRT computation of $m^d \mod N$, the faulty result may be used to factor $N$. Let $u = m^d \mod N$ be the expected result of the computation and $v$ be a faulty result such that $v \equiv u \mod p$, $v \not\equiv u \mod q$.

1. Give an expression for $a := u^e \mod N$ in function of $m$, $C_p$, $C_q$ and $N$.

2. Give an expression for $b := v^e \mod N$ in function of $m$, $C_p$, $C_q$, $N$, and an unknown quantity $x$.

3. Show that $\gcd((a - b), N)$ reveals a non-trivial factor of $N$.

**Q. 5:**

1. Conclude on the importance of protecting RSA-CRT implementations against faulty computations.