# Cryptography and Security — Midterm Exam

Serge Vaudenay

15.10.2025

- duration: 1h45
- **no documents** allowed, except one 2-sided sheet of handwritten notes
- a pocket calculator is allowed
- communication **devices are not allowed**
- the exam invigilators will **not answer** any technical question during the exam
- readability and style of writing will be part of the grade
- answers should **not be written with a pencil**

## 1 Nonce-Based Multi-Use Perfect Secrecy

We define a nonce-based encryption scheme as follows: given a plaintext $X$, a key $K$, and an additional input called *nonce* $N$, the ciphertext $Y$ is obtained by $Y = \mathsf{Enc}(K, N, X)$. Decryption takes input $K$, $N$, $Y$ and returns $X = \mathsf{Dec}(K, N, Y)$. It is forbidden to use the same nonce $N$ more than once to encrypt. Furthermore, we do not assume that the nonce is private (hence, the adversary has access to $Y$ and $N$). The probability distributions of $K$, $N$ and $X$ are supposed to be independent. We assume that $X$, $Y$, and $N$ belong to the same domain $\mathbf{F}$, which is a finite field. We assume that the distribution of $K$ is determined by the encryption system. We say that the encryption system is *1-time secure* if for any $x, x', y, n \in \mathbf{F}$, we have

$$\Pr[\mathsf{Enc}(K, n, x) = y] = \Pr[\mathsf{Enc}(K, n, x') = y]$$

**Q.1** In this question only, we forget about the nonce $N$ (i.e. $Y = \mathsf{Enc}(K, X)$). Recall the definition of perfect secrecy and prove that the above notion (without $n$) is equivalent when the support of $X$ is $\mathbf{F}$.
HINT: Show that 1-time security implies $\forall x, x', y \quad \Pr[Y = y|X = x] = \Pr[Y = y|X = x']$.

**Q.2** Propose an efficient nonce-based encryption system which is 1-time secure.

**Q.3** We say that the encryption system is 2-time secure if for any $x_1, x_2, x_1', x_2', y_1, y_2, n_1, n_2 \in \mathbf{F}$ such that $n_1 \neq n_2$, we have

$$\Pr[\mathsf{Enc}(K, n_1, x_1) = y_1, \mathsf{Enc}(K, n_2, x_2) = y_2] = \Pr[\mathsf{Enc}(K, n_1, x_1') = y_1, \mathsf{Enc}(K, n_2, x_2') = y_2]$$

Prove that 2-time security implies that the key space is at least as large as $\mathbf{F}^2$.
HINT: Use $\mathsf{Enc}'(K, (n_1, n_2), (x_1, x_2)) = (\mathsf{Enc}(K, n_1, x_1), \mathsf{Enc}(K, n_2, x_2))$.

**Q.4** Propose a 2-time secure encryption system. (Prove that it is secure.)

**Q.5** Propose a definition for $d$-time security and propose a nonce-based system which achieves it.

## 2 Multiexponentiation

In this exercise, we consider algorithms to compute $g^e$ in an Abelian group $G$ (which uses multiplicative notations). Given an integer $e$, we denote by $e[i] = \lfloor \frac{e}{2^i} \rfloor \bmod 2$ the $i$th bit of $e$ (e.g. $e[0]$ is the least significant bit). We denote by $e[j \cdots i]$ the number obtained by concatenating the bits from the $j$th to the $i$th. (For instance, for $e = 23 = 10111_2$, we have $e[4 \cdots 0] = 10111_2 = 23$ and $e[3 \cdots 1] = 011_2 = 3$.) We recall the square-and-multiply algorithm from left to right.

$\mathsf{Exp}(g, e)$:

```
1: for i = ℓ − 1 down to 0 do
2:     if i = ℓ − 1 then
3:         x ← 1
4:     else
5:         x ← x^2
6:     end if
7:     if e[i] = 1 then
8:         x ← x × g
9:     end if
10: end for
11: return x
```

**Q.1** Assuming that $e$ is a random string of $\ell$ bits, what are the average number of squarings and the average number of multiplications in the square-and-multiply algorithm?

**Q.2** We now use the so-called $2^w$-ary method: we first make a precomputation of all $g^b$ for $b = 0, \ldots, 2^w - 1$, we split $e$ in blocks of $w$ bits, then we scan all the blocks from left to right.

**Q.2a** Fully specify the precomputation algorithm with input $g$.

**Q.2b** What is the number of squarings and of multiplications?

**Q.2c** Fully specify the algorithm to compute $g^e$ using the results of the precomputation.

**Q.2d** What is the expected number of squarings and of multiplications in this phase?

**Q.2e** Assuming that $\ell$ is divisible by $w$, compute the sum of the total number of squarings and of multiplications in the two phases and compare with the normal square-and-multiply algorithm for $w = 1, 2, 3, 4$ and say for which $\ell$ which algorithm is better.

**Q.3** We now want to compute $g_1^{e_1} \cdots g_k^{e_k}$, where all $e_i$ are written with $\ell$ bits.

**Q.3a** What is the expected number of squarings and multiplications if we use the naïve method based on the square-and-multiply algorithm?

**Q.3b** Inspired by the $2^w$-ary method, propose an algorithm based on the precomputation of $T\{b_1, \ldots, b_k\} = g_1^{b_1} \cdots g_k^{b_k}$ for all the $2^{kw}$ possible combinations of blocks. (We assume that $g_1, \ldots, g_k$ are available for precomputation.)

**Q.3c** What is the expected number of squarings and multiplications in each phase? Make a comparison with the naïve algorithm for $k = 2$ as it was done for the $k = 1$ case.