

Cryptography and Security — Midterm Exam

Serge Vaudenay

5.12.2018

- 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 Design Challenges with Bad Requirements

This exercise proposes two totally independent design challenges. Students with the best answer will get the full points. Others will have partial points. (To compare the answers, we check if all requirements are satisfied and we then look at the complexity of the algorithms.)

- Q.1** (Design challenger 1.) You must design a block cipher $(\{0, 1\}^k, \{0, 1\}^n, \text{Enc}, \text{Dec})$ (following the definition from the course in which k is the key length and n is the block length) which is secure against key recovery under chosen plaintext and ciphertext attacks. More precisely, it must be $(q, t, 2^{-k})$ -secure for any q and t , i.e. whatever the number of queries q and the time complexity t , the probability of success should not be better than the probability of success 2^{-k} of an attack which guesses the key at random.

- Q.2** (Design challenger 2.) Given some parameters k and n , you must design two algorithms Enc and Dec with the following interface:
- Enc takes two inputs $K \in \{0, 1\}^k$ and $\text{pt} \in \{0, 1\}^n$ and produce one output ct .
 - Dec takes two inputs $K \in \{0, 1\}^k$ and ct and produce one output pt' .
- The obtained pair (Enc, Dec) must be secure against decryption under chosen plaintext attack. More precisely, it must be $(q, t, 2^{-n})$ -secure for any q and t , i.e. whatever the number of queries q and the time complexity t , the probability of success should not be better than the probability of success 2^{-n} of an attack which guesses pt at random.

2 RSA with Carmichael Numbers

We recall that by definition, a Carmichael number is an integer n which is the product of several (at least two) pairwise different prime numbers p_i such that $p_i - 1$ divides $n - 1$. We know that a positive integer n is a Carmichael number if and only if it is not prime and for any $b \in \mathbf{Z}_n^*$, we have $b^{n-1} \bmod n = 1$.

In what follows, we consider two Carmichael numbers p and q , and $n = pq$. We denote $\delta = (p - 1) \times (q - 1)$.

Q.1 Assuming that p and q are coprime, formally prove that $x^\delta \bmod n = 1$ for all $x \in \mathbf{Z}_n^*$?

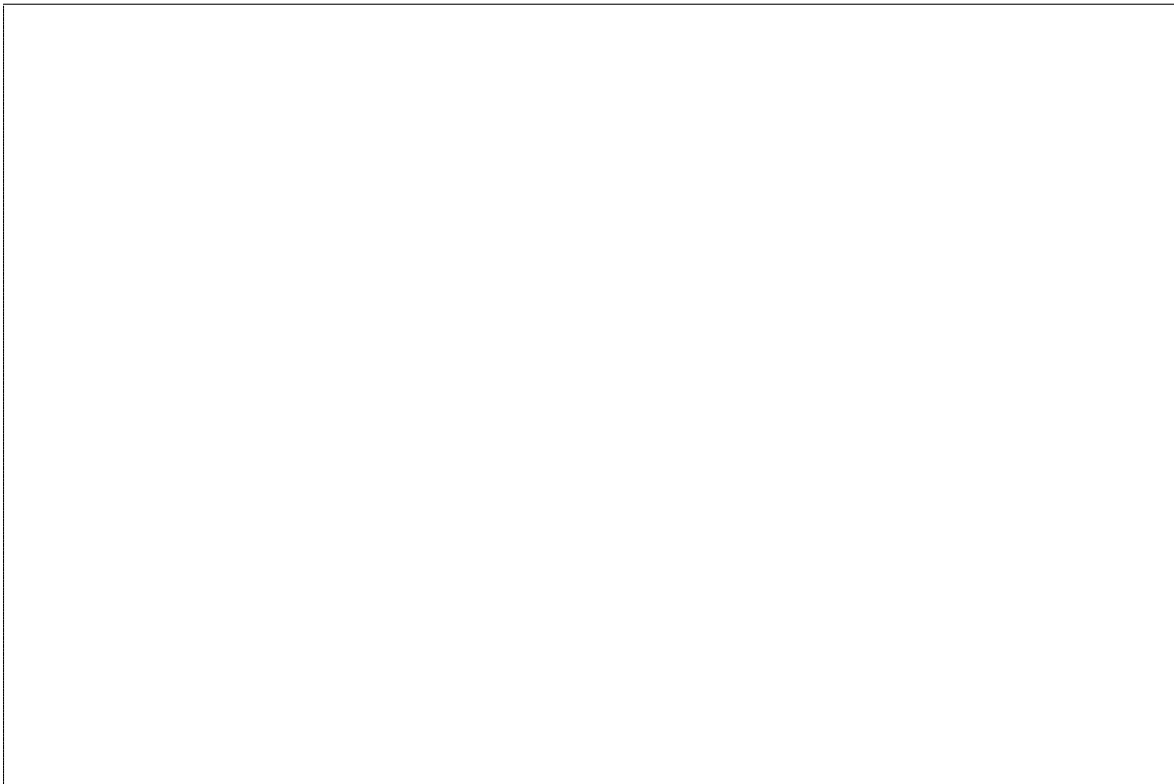
In what follows, we assume that p and q are coprime.

Q.2 If e and d are integers such that $ed \bmod \delta = 1$, show that for all $x \in \mathbf{Z}_n^*$, we have $x^{ed} \bmod n = x$.

Q.3 With the same notations, prove that $x^{ed} \bmod n = x$ for all $x \in \mathbf{Z}_n$.



Q.4 Is it a good idea to use such variant of RSA?



3 Invalid Curve Attack over ECDH

The ECDH protocol uses an elliptic curve defined by some domain parameters $D = (q, a, b, P, n)$, where q and n are prime, $a, b \in \mathbf{Z}_q$, and P is a point of order n on the elliptic curve over \mathbf{Z}_q defined by the equation $y^2 = x^3 + ax + b$. In ECDH, each participant U generates his ephemeral secret key $\text{sk}_U \in \mathbf{Z}_n^*$ and its ephemeral public key $\text{pk}_U = \text{sk}_U \times P$. Two participants A and B end up with a secret $\text{sk}_A \times \text{pk}_B = \text{sk}_B \times \text{pk}_A$ from which they extract some key DHKey. We focus on Bluetooth using the P256 curve. In Bluetooth, two devices which want to communicate without knowing each other first run the ECDH protocol to share a secret DHKey, then authenticate the x -coordinates of exchanged public keys by using an alternate communication channel. So, the objective of an adversary could be to interfere with the communication without modifying the x -coordinates so that one or both participants end up with some DHKey which is known by the adversary. The goal of this exercise is to mount such attack for the Bluetooth implementation of ECDH.

In Bluetooth, ECDH is run as follows: each participant U interacting with his counterpart \bar{U} does what follows:

- U sets up the parameters D of P256.
- U picks $\text{sk}_U \in \mathbf{Z}_n^*$ at random and computes $\text{pk}_U = \text{sk}_U \times P$. We denote by (x_U, y_U) the coordinates of pk_U . The computation is done with the double-and-add algorithm.
- U sends (x_U, y_U) to \bar{U} .
- U receives $(x_{\bar{U}}, y_{\bar{U}})$ from \bar{U} and verifies that it is not the point at infinity.
- U computes $(K_x, K_y) = \text{sk}_U \times (x_{\bar{U}}, y_{\bar{U}})$. This computation is done with the double-and-add algorithm.
- U throws away K_y and computes a KDF function on K_x to obtain DHKey.
- After that, U authenticates $x_U, x_{\bar{U}}$ and some other values by some ad-hoc means.

We recall how point addition and point doubling is done. We denote $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$, $R = (x_R, y_R)$. For $P \neq Q$ and $x_P = x_Q$, $P + Q$ is the point at infinity. Otherwise, $R = P + Q$ is computed by first computing

$$\lambda = \begin{cases} \frac{y_P - y_Q}{x_P - x_Q} \bmod q & \text{if } P \neq Q \\ \frac{3x_P^2 + a}{2y_P} \bmod q & \text{if } P = Q \end{cases} \quad x_R = \lambda^2 - 2x_P \bmod q \quad y_R = -y_P - \lambda(x_R - x_P) \bmod q$$

Q.1 Based on what you have learned on the Diffie-Hellman protocol, tell what is missing and how we should have implemented it.

Q.2 Given an attack in which the adversary replaces $y_{\bar{U}}$ by some value $y'_{\bar{U}}$ such that $(x_{\bar{U}}, y'_{\bar{U}})$ lies on a curve of equation $y^2 = x^3 + ax + b'$ and $(x_{\bar{U}}, y'_{\bar{U}})$ has order 2 on this curve. Give the method to compute b' and $y'_{\bar{U}}$ from D and $x_{\bar{U}}$.

Q.3 Prove that the algorithm mapping an input point Q to $\text{sk}_U \times Q$ on the curve $y^2 = x^3 + ax + b$ and on the curve $y^2 = x^3 + ax + b'$ by using the double-and-add algorithm are exactly the same algorithm. Deduce that by applying the previous attack, the adversary can easily compute DHKey which is obtained by U .

Q.4 Applying the previous attack in both directions, prove that both participants obtain the same DHKey with probability $\frac{1}{4}$.