Advanced Cryptography — Final Exam
Solution

Serge Vaudenay

30.6.2021

- duration: 3h
- any document allowed
- 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

The exam grade follows a linear scale in which each question has the same weight.

1 Encryption Security with a Ciphertext Checking Oracle

We consider the following One-Way under Validity Checking Attack (OW-VCA) game. The advantage of the adversary is the probability it returns 1.

Game $\Gamma^{\mathcal{A}}(1^s)$:
1: $\mathsf{Gen}(1^s) \to \mathsf{pk}, \mathsf{sk}$
2: pick $\mathsf{pt}^* \in \mathcal{M}_s$ at random
3: $\mathsf{Enc}(\mathsf{pk}, \mathsf{pt}^*) \to \mathsf{ct}^*$
4: $\mathcal{A}^{\mathsf{VCO}}(\mathsf{pk}, \mathsf{ct}^*) \to z$
5: **return** $1_{z=\mathsf{pt}^*}$

Oracle $\mathsf{VCO}(\mathsf{ct})$
6: $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$
7: **return** $1_{x \neq \perp}$

Where s is the security parameter, $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a public-key cryptosystem, $\mathcal{M}_s$ is the plaintext domain, and $\perp$ is the special output of Dec indicating that decryption failed.

Q.1 Is PKCS#1 v1.5 secure with respect to this notion?

We have seen in the course that there is an algorithm $\mathcal{A}$ such that given a target ciphertext $\mathsf{ct}^* = y$, makes some queries $\mathsf{pt} = s^e y \bmod N$ to the VCO oracle for some well chosen s and eventually decrypts ct^*. This is the Bleichenbacher attack. So, PKCS#1 v1.5 is not secure in this model.
Giving a referene to the attack seen in the course was enough to get the points.

Q.2 Propose a definition of KR-VCA security whose goal is key recovery.

Game $\Gamma^{\mathcal{A}}(1^s)$*:*
1: $\mathsf{Gen}(1^s) \to \mathsf{pk}, \mathsf{sk}$
2: $\mathcal{A}^{\mathsf{VCO}}(\mathsf{pk}) \to z$
3: ***return*** $1_{z=\mathsf{sk}}$

Oracle $\mathsf{VCO}(\mathsf{ct})$
4: $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$
5: ***return*** $1_{x \neq \perp}$

The advantage of the adversary is the probability the game returns 1.
One common mistake was to generate a "target message", to encrypt it, and to give it to the adversary. This was useless as the adversary could do it by himself.

Q.3 We recall the Regev cryptosystem over the plaintext domain $\mathcal{M} = \{0, 1\}$.

Gen selects a prime number p, integers m and n, a parameter $\sigma \ll \frac{p}{m}$. Then, it selects a secret $\mathsf{sk} \in \mathbf{Z}_p^n$ and a public key $\mathsf{pk} = (A, b)$ satisfying $b = A \times \mathsf{sk} + e \bmod p$, where $A \in \mathbf{Z}_p^{m \times n}$ is a $m \times n$ matrix and $e \in \mathbf{Z}_p^m$ is an error vector which is selected as follows: for each component i, we sample a real number with normal distribution with mean 0 and standard deviation σ and take e_i as its nearest integer.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{pt})$ picks a vector $v \in \{0, 1\}^m$ at random, $c_1 = v^t \times A \bmod p$, $c_2 = \mathsf{pt} \times \left\lfloor \frac{p}{2} \right\rfloor + v^t b \bmod p$, and returns $\mathsf{ct} = (c_1, c_2)$.

$\mathsf{Dec}(\mathsf{sk}, (c_1, c_2))$ computes $d = c_2 - c_1 \times \mathsf{sk} \bmod p$ then pt' such that $d - \mathsf{pt}' \times \left\lfloor \frac{p}{2} \right\rfloor$ is congruent to an integer in the $[-\frac{p}{4}, +\frac{p}{4}]$ interval modulo p.

Prove that the cryptosystem is correct.

If we follow the protocol, then

$$c_2 = \mathsf{pt} \left\lfloor \frac{p}{2} \right\rfloor + v^t A \times \mathsf{sk} + v^t e \bmod p$$

so $d = \mathsf{pt} \left\lfloor \frac{p}{2} \right\rfloor + v^t e \bmod p$. $v^t e$ *is the sum of at most* m *integers rounded from some normally distributed number of mean 0 and standard deviation* σ*. Hence, the order of magnitude of* $v^t e$ *is at most* $m\sigma$*. Since* $\sigma \ll \frac{p}{m}$*, this is small compared to* p*. Hence, we can assume that with high probability,* $v^t e$ *is congruent to an integer in the* $[-\frac{p}{4}, +\frac{p}{4}]$ *interval modulo* p*. In that case, decryption yields* pt*.*
To get the points, it was necessary to justify that $v^t e$ *is small. A penalty could occur if a student said correctness was perfect.*

Q.4 Make a successful KR-CCA attack on the Regev cryptosystem.

The adversary first encrypts a random plaintext pt *into* (c_1, c_2) *Then, it queries* $(c_1, c_2 + \delta)$ *to the decryption oracle with a chosen* δ*. The adversary makes a cut-and-choose algorithm on the value of* δ *to find the one such that* $(c_1, c_2 + \delta)$ *decrypts to the right plaintext but* $(c_1, c_2 + \delta + 1)$ *does not. This roughly requires* $\log_2 p$ *queries. It deduces that*

$$c_2 + \delta - c_1 \times \mathsf{sk} = \left\lfloor \frac{p}{4} \right\rfloor \pmod{p}$$

With this, the adversary deduces sk*.*
An alternate solution (which was not useful in the next question) was to set $v_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ *on the* i*-th coordinate and to call the decryption oracle on* $\mathsf{ct} = (v_i, c_2)$ *to obtain* $1_{c_2 - \mathsf{sk}_i \in \left[-\frac{p}{4}, \frac{p}{4}\right]}$*. By trying several* c_2 *(by cut-and-choose), we get* sk_i*.*

Q.5 We define a cryptosystem over a domain $\mathcal{M}_s$ as follows: Gen is like in the Regev cryptosystem, Enc first computes $x = (\mathsf{pt}, H(\mathsf{pt}))$ using a hash function, then encrypt each of the n bits of x using the Regev cryptosystem to obtain $\mathsf{ct} = \mathsf{ct}_1, \ldots, \mathsf{ct}_n$. Dec decrypts the n ciphertexts to obtain n bits x' which are parsed into $x' = (\mathsf{pt}', h')$. If $h' = H(\mathsf{pt}')$, then pt' is returned. Otherwise, $\perp$ is returned.
Prove that this cryptosystem is not KR-VCA secure.

We apply the previous attack on the first ciphertext component. When $x'_1 = x_1$*, the* VCO *oracle returns 1. Otherwise, it returns* $\perp$*. The adversary can deduce if* $x'_1 = x_1$ *or* $x'_1 = 1 - x_1$*. This is enough to run the attack.*

2 Optimal Resistance to Linear Cryptanalysis Modulo 2

This exercise is inspired from Baignères-Junod-Vaudenay, How Far Can We Go Beyond Linear Cryptanalysis? *, ASIACRYPT 2004, LNCS vol. 3329, Springer.*

Let n be an integer. We consider $X_1, \ldots, X_n$ i.i.d. random variables which are uniform over $\mathbf{Z}_4$. We consider Y independent from $X_1, \ldots, X_n$ and uniformly distributed in $\{0,1\}$. We let $X_{n+1} = Y + X_1 + \cdots + X_n \bmod 4$. Finally, $X = (X_1, \ldots, X_{n+1}) \in \mathbf{Z}_4^{n+1}$. We write X as a bitstring of length $2n+2$ by concatenating the binary representation of the X_i over two bits. We denote the bits $X[1], \ldots, X[2n+2]$. Hence, $X_1 = 2X[1] + X[2]$, $X_2 = 2X[3] + X[4]$, etc. We recall that for a random variable B, we have $\mathsf{LP}(B) = \left(E((-1)^B)\right)^2$.

The goal of the exercise is to show that although for every balanced linear function $x \mapsto a \cdot x$ from $\mathbf{Z}_2^{2n+2}$ to $\mathbf{Z}_2$, the LP bias is very small, there exists a balanced Boolean function $x \mapsto f(x)$ whose LP bias is huge.

Q.1 Let B be the most significant bit of $X_{n+1} - X_1 - \cdots - X_n \bmod 4$.
Compute $\mathsf{LP}(B)$.

By definition of X_{n+1}, we have $X_{n+1} - X_1 - \cdots - X_n \bmod 4 = Y$. Since $Y \in \{0,1\}$, we have $B = 0$ all the time. Hence, $\mathsf{LP}(B) = 1$.
A frequent mistake was to say that B was the most significant bit of Y which was Y itself as it had only one bit. The question meant "the most significant bit in the 2-bit representation". If we understand it as "the most significant bit of the number it represents", this bit is always 1 unless the number is 0 (which would have indeed be equivalent to Y in this case). It was clearly not the correct meaning in this question.

Q.2 Let a be a nonzero binary mask over $2n+2$ bits such that $a[2n+1] = 0$.
Prove that $\mathsf{LP}(a \cdot X) = 0$.

We have $X[2n+2] = Y + X[2] + \cdots + X[2n] \bmod 2$. We assume that $a[2n+1] = 0$. Hence,

$$\begin{aligned} a \cdot X = {} & (a[1]X[1] + a[3]X[3] + \cdots + a[2n-1]X[2n-1]) + \\ & ((a[2] + a[2n+2])X[2] + \cdots + (a[2n] + a[2n+2])X[2n]) + \\ & a[2n+2]Y \pmod 2 \end{aligned}$$

Since $X[1], \ldots, X[2n], Y$ are i.i.d. and uniform, $a \cdot X \bmod 2$ is an unbiased random bit so $\mathsf{LP}(a \cdot X) = 0$.
A common mistake was to use the piling up lemma without caring for the independence of the random bits.

Q.3 Let a be a binary mask over $2n+2$ bits such that $a[2n+1] = 1$ and $a[i] = 0$ for some odd index i.

Prove that $\mathsf{LP}(a \cdot X) = 0$.

HINT: $X[2n+1] = \sum_j X[2j-1] + \sum_{j<j'} X[2j]X[2j'] + \sum_j X[2j]Y$ where j and j' go from 1 to n.

The hint comes from the property of the addition $X_1 + \cdots + X_n + Y$: we add all most significant bits of $X_1, \ldots, X_n$ and the carry which is the sum of the bi-products of the least significant bits.
We continue the computation in the previous question by adding $a[2n+1]X[2n+1]$ and observing that $a[i]X[i] = 0$ for some odd i. The term $a[2n+1]X[2n+1]$ can be written $f(X[i], \ldots, X[i-1], X[i+1], \ldots, X[2n], Y) + X[i]$ for some Boolean function f. Since $X[i]$ is uniform and independent from the rest in the sum, then $a \cdot X$ is unbiased and $\mathsf{LP}(a \cdot X) = 0$.

Q.4 Let a be a binary mask over $2n+2$ bits such that $a[i] = 1$ for every odd index i.

Prove that $\mathsf{LP}(a \cdot X) = 2^{-n-1}$ for n odd.

HINT: For every n, $\left(\sum_{w=0}^{n-1} \binom{n}{w} (-1)^{\frac{w(w-1)}{2}}\right)^2 = 2^n \left(1 + \sin \frac{n\pi}{2}\right)$.

With the computations of the previous questions, we have

$$\begin{aligned} a \cdot X = &((a[2] + a[2n+2])X[2] + \cdots + (a[2n] + a[2n+2])X[2n]) + \\ &a[2n+2]Y + \\ &\sum_{j<j'} X[2j]X[2j'] + \sum_{j} X[2j]Y \pmod 2 \end{aligned}$$

By making changes of variable $x_i = X[2i] + \mathsf{cte}$ *for* $i = 1, \ldots, n$ *and* $x_{n+1} = Y$, *we obtain* $a \cdot X = \sum_{j<j'} x_j x_{j'} + \mathsf{cte}$ *with* j *and* j' *from 1 to* $n+1$. *The constant plays no role in the bias. We let* $\Sigma = \sum_{j<j'} x_j x_{j'}$.
If w *is the sum of all* x_j *(i.e., the number of 1's), then the number of nonzero bi-products is* $\Sigma = \frac{w(w-1)}{2}$. *The value modulo 2 depends of the value of* $w \bmod 4$. *If* $w \bmod 4 \in \{0, 1\}$, Σ *is even. Otherwise,* Σ *is odd. Thus,*

$$\mathsf{LP}(a \cdot X) = 2^{-2n-2} \left(\sum_{w=0}^{n} \binom{n+1}{w} (-1)^{\frac{w(w-1)}{2}} \right)^2$$

After painful calculus, we obtain the formula of the hint and deduce

$$\mathsf{LP}(a \cdot X) = 2^{-n-1} \left(1 + \sin \frac{(n+1)\pi}{2} \right)$$

For n *odd,* $\sin \frac{(n+1)\pi}{2} = 0$ *so* $\mathsf{LP}(a \cdot X) = 2^{-n-1}$.
Considering that every $\mathbf{Z}_2$*-linear bit of* X *has an LP bounded by* 2^{-n}, *having a bit* B *with LP equal to 1 is surprising. We deduce that having all* $\mathbf{Z}_2$*-linear bits with low LP does not proving any insurance on the nonexistence of a strongly biased bit.*
To prove the hint, we compute

$$\Sigma_u = \sum_{w=0}^{n-1} \binom{n}{w} i^{uw} = (1 + i^u)^n$$

We have

$$\frac{(1-i)\Sigma_1 + (1+i)\Sigma_3}{2} = \sum_{w=0}^{n-1} \binom{n}{w} \frac{(1-i)i^w + (1+i)i^{-w}}{2} = \sum_{w=0}^{n-1} \binom{n}{w} (-1)^{\frac{w(w-1)}{2}}$$

Hence,

$$\sum_{w=0}^{n-1} \binom{n}{w} (-1)^{\frac{w(w-1)}{2}} = \frac{(1-i)(1+i)^n + (1+i)(1-i)^n}{2} = \mathsf{Re}\left((1-i)(1+i)^n\right)$$

Since $(1-i)(1+i)^n = 2^{\frac{n+1}{2}} e^{(n-1)i\frac{\pi}{4}}$ *and* $\cos^2(n-1)\frac{\pi}{4} = \frac{1+\cos(n-1)\frac{\pi}{2}}{2} = \frac{1+\sin\frac{n\pi}{2}}{2}$, *we obtain the result.*

3 MPC-in-the-Head

Let R be a relation over bitstrings x and w defining an NP language. We assume a multi-party computation (MPC) with two participants A and B such that

- A and B have as public common input x;
- A and B have respective private inputs w_A and w_B;
- A and B have as final common output $R(x, w_A \oplus w_B)$;
- a malicious participant learns nothing about the private input of honest participants.

We let $\mathcal{U}(x, w_U; r_U)$ be the protocol run by $U \in \{A, B\}$ and $\mathsf{Run}(x, \mathcal{A}(w_A; r_A), \mathcal{B}(w_B; r_B))$ be the interaction. We will use a commitment scheme Commit.

We define a Σ protocol over the challenge set $\{A, B\}$ as follows.

- $\mathcal{P}(x, w)$ first flips w_A, r_A, r_B, sets $w_B = w_A \oplus w$, then simulates the interaction $\mathsf{Run}(x, \mathcal{A}(w_A; r_A), \mathcal{B}(w_B; r_B))$. It computes the transcript t (i.e. x and the list of exchanged messages) of the protocol.
- It flips k_A and k_B and computes $c_A = \mathsf{Commit}(w_A, r_A; k_A)$ and $c_B = \mathsf{Commit}(w_B, r_B; k_B)$.
- The message $a = (t, c_A, c_B)$ is sent to $\mathcal{V}$.
- $\mathcal{V}$ flips a challenge $e \in \{A, B\}$ and sends it to $\mathcal{P}$.
- $\mathcal{P}$ sends $z = (w_e, r_e, k_e)$.
- $\mathcal{V}$ makes a final verification.

Q.1 Describe the final verification of $\mathcal{V}$ and prove that the Σ protocol is correct.

The verifier must verify two things: that $c_e = \mathsf{Commit}(w_e, r_e; k_e)$ and that running $\mathcal{E}(\mathsf{View}_e)$ for the view reconstructed from (w_e, r_e, t) gives the messages from E in t and finally outputs 1.
When we run the MPC protocol, the algorithm run on the view outputs $R(x, w_A \oplus w_B) = R(x, w) = 1$ so this is correct.

Q.2 Define an extractor and prove it is correct.

If we have the response to two challenges, we have opening of every commitment. So we have two matching views which correspond to an execution of the MPC protocol which outputs $R(x, w_A \oplus w_B) = 1$. We can thus compute a valid witness $w_A \oplus w_B$.
A common mistake was to prove the correct extraction only with the w_A and w_B computed by the honest prover.

Q.3 How would we define a simulator? (An informal argument is fine for this question.)

We should first formalize what it means that "a malicious participant learns nothing about the private input of honest participants". We take the example of a malicious A^*. We denote by $\mathcal{A}^*$ the algorithm which says how A^* acts. This must be formalized by saying there is a simulator $S_B^{\mathcal{A}^*}$ such that the view of $\mathcal{A}^*$ in the interaction $\mathsf{Run}(x, \mathcal{A}^*(w_A; r_A), \mathcal{B}(w_B; r_B))$ is indistinguishable from $S_B^{\mathcal{A}^*}(w_A, r_A)$. Then, for the challenge $e = A$, we define $\mathcal{A}^* = \mathcal{A}$ and we compute the transcript t from $S_B^{\mathcal{A}^*}(w_A, r_A)$ on random w_A and r_A. We can then commit to (w_A, r_A) with a random k_A. With a perfectly hiding commitment, we take c_B at random and we can form $a = (t, c_A, c_B)$ and $z = (w_A, r_A, k_A)$.
For the challenge $e = B$, the simulation is similar.
A common mistake was to simulate by running the MPC protocol with random w_A and w_B. The revealed values does not allow to get both w_A and w_B but the obtained views are unfortunately perfectly distinguishable from the correct views. Indeed, the MPC returns $R(x, w_A \oplus w_B)$ in clear. We cannot even restart until R returns 1 as it is hard to generate witnesses at random. Trying to modify messages in the MPC goes back to the problem of simulating a valid MPC transcript.