Cryptography and Security — Midterm Exam
Solution

Serge Vaudenay

9.12.2016

- 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

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

1 An Attempt to Fix Double Encryption

We consider a block cipher C over n-bit blocks with a key of n bits. We define $\mathsf{Enc}_{K_1,K_2,K_3}(x) = C_{K_3}(C_{K_1}(x) \oplus K_2)$ where $\oplus$ is the bitwise XOR operation. This defines a new block cipher with n-bit blocks and $3n$-bit keys. We consider key recovery known plaintext attacks against Enc using r pairs (x_i, y_i) such that $y_i = \mathsf{Enc}_{K_1,K_2,K_3}(x_i)$ for $i = 1, \ldots, r$.

Throughout this exercise, we measure the time complexity in terms of number of C or C^{-1} operations.

Q.1 In this question, we assume that K_2 is fixed and equal to 0.

Q.1a Show that the equation $y_i = \mathsf{Enc}_{K_1,K_2,K_3}(x_i)$ can be written in the form $f_i(K_1) = g_i(K_3)$ for some functions f_i and g_i.

Clearly,

$$\begin{aligned} & y_i = \mathsf{Enc}_{K_1,K_2,K_3}(x_i) \\ \Longleftrightarrow\ & y_i = C_{K_3}(C_{K_1}(x_i) \oplus K_2) \\ \Longleftrightarrow\ & y_i = C_{K_3}(C_{K_1}(x_i)) \\ \Longleftrightarrow\ & C^{-1}_{K_3}(y_i) = C_{K_1}(x_i) \end{aligned}$$

which is of form $f_i(K_1) = g_i(K_3)$ *with* $f_i(K_1) = C_{K_1}(x_i)$ *and* $g_i(K_3) = C^{-1}_{K_3}(y_i)$.

Q.1b Using the previous question, describe an attack method with time complexity of order of magnitude 2^n. (Justify the complexity.)

We have the meet-in-the-middle attack:

initialize the hash table T to empty
for *all* K_1 **do**
    compute $z = f_1(K_1)$
    if $T[z]$ *undefined* **then**
        add list (K_1) *in a hash table with key* z: $T[z] \leftarrow (K_1)$
    else
        insert K_1 *in the list* $T[z]$: $T[z] \leftarrow (K_1, T[z])$
    end if
end for
for *all* K_3 **do**
    compute $z = g_1(K_3)$
    if $T[z]$ *defined* **then**
        for *all* K_1 *in list* $T[z]$ **do**
            set $i \leftarrow 2$
            while $i \leq r$ *and* $f_i(K_1) = g_i(K_3)$ **do**
                set $i \leftarrow i + 1$
                if $i > r$ **then**
                    yield (K_1, K_3) *as output*
                end if
            end while
        end for
    end if
end for

It may yield several outputs but it must include the correct one. Spurious outputs are ruled out by increasing r as it will check for more i for $i = 2, \ldots, r$.
With $r = \mathcal{O}(1)$, the attack has time complexity $\mathcal{O}(2^n)$. Even with larger r, we can see that the probability the while loop iterates more than a constant time is very small. So, the number of K_3 for which we need many iteration is small. The complexity remains $\mathcal{O}(2^n)$.
The optimal value for r is analyzed in the next question.

Q.1c Analyze the probability of success (the probability that it produces the correct solution and only the correct one). Propose (and justify) a minimal value for r to produce a good result.

The attack gives the right solution with probability 1, but may give spurious (K_1, K_3) solutions. Each wrong (K_1, K_3) pair is solution to the system $f_i(K_1) = g_i(K_3)$ for $i = 1, \ldots, r$ with probability 2^{-rn}. We have $2^{2n} - 1$ possible bad solutions. The probability to have no spurious solution is thus $(1 - 2^{-rn})^{2^{2n}-1} \approx e^{-2^{(2-r)n}}$. For $r = 1$, this probability is e^{-2^n} which is close to 0. For $r = 2$, this probability is $e^{-1} \approx 37\%$. For $r = 3$, this probability is $e^{-2^{-n}} \approx 1 - 2^{-n}$ which is very close to 1. So, $r = 3$ is enough to recover the right solution and only this one with probability very close to 1.

Q.2 We now assume that K_2 is part of the secret with n bits of entropy.

Q.2a Show that the attack of the previous question can be directly adapted to obtain an attack of complexity 2^{2n}.

We set $g_i(K_2, K_3) = C^{-1}_{K_3}(y_i) \oplus K_2$ and have a loop over all (K_2, K_3) instead of K_3. We obtain a time complexity of 2^{2n}.

Q.2b Show that two equations $y_i = \mathsf{Enc}_{K_1,K_2,K_3}(x_i)$ and $y_j = \mathsf{Enc}_{K_1,K_2,K_3}(x_j)$ imply an equation which can be written in the form $f_{i,j}(K_1) = g_{i,j}(K_3)$ for some functions $f_{i,j}$ and $g_{i,j}$.

$$
\begin{aligned}
& y_i = \mathsf{Enc}_{K_1,K_2,K_3}(x_i) \quad \text{and} \quad y_j = \mathsf{Enc}_{K_1,K_2,K_3}(x_j) \\
\iff & y_i = C_{K_3}(C_{K_1}(x_i) \oplus K_2) \quad \text{and} \quad y_j = C_{K_3}(C_{K_1}(x_j) \oplus K_2) \\
\iff & C^{-1}_{K_3}(y_i) = C_{K_1}(x_i) \oplus K_2 \quad \text{and} \quad C^{-1}_{K_3}(y_j) = C_{K_1}(x_j) \oplus K_2 \\
\implies & C^{-1}_{K_3}(y_i) \oplus C^{-1}_{K_3}(y_j) = C_{K_1}(x_i) \oplus C_{K_1}(x_j)
\end{aligned}
$$

which is of form $f_{i,j}(K_1) = g_{i,j}(K_3)$ with $f_{i,j}(K_1) = C_{K_1}(x_i) \oplus C_{K_1}(x_j)$ and $g_{i,j}(K_3) = C^{-1}_{K_3}(y_i) \oplus C^{-1}_{K_3}(y_j)$.

Q.2c Deduce an attack method of complexity 2^n and make the analysis like in Q.1c.

Using the previous question, we can use the equations $f_{1,2}(K_1) = g_{1,2}(K_3)$ to obtain the correct (K_1, K_3) and some spurious ones. For each found solution we can compute $K_2 = C^{-1}_{K_3}(y_1) \oplus C_{K_1}(x_1)$. Then, we can check if (K_1, K_2, K_3) is consistent with additional equations $\mathsf{Enc}_{K_1,K_2,K_3}(x_i) = y_i$ for $i = 3, \ldots, r$.
The obtained attack has time complexity 2^n.
The probability to have no spurious solution is now $(1 - 2^{-rn})^{2^{3n}-1} \approx e^{-2^{(3-r)n}}$ and we need $r = 4$ to have a probability close to 1 to get only the right key.

2 The Hill Cipher

Let d be an integer. We define the Hill cipher with security parameter d as follows. The message space is $\mathbf{Z}_{26}^d$. Messages are strings of d alphabetical characters encoded into $\mathbf{Z}_{26}$. The key space is the set of invertible $d \times d$ matrices over $\mathbf{Z}_{26}$. Given a key K and a message X, the encryption of X under K is $\mathsf{Enc}_K(X) = K \times X$ with operations modulo 26.

Q.1 Explain how the decryption works.

> *As the square matrix K is invertible, we can invert it and we obtain $\mathsf{Dec}_K(Y) = K^{-1}Y$.*

Q.2 Propose a chosen plaintext key recovery attack with complexity $\mathcal{O}(d^2)$ using d chosen plaintexts. (Justify the complexity.)
HINT: assume that read/write of a $\mathbf{Z}_{26}$ element costs $\mathcal{O}(1)$ complexity.

> *We let $X_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ where the 1 is at position i. The vector $Y_i = K \times X_i$ is the ith column of K. So, using these d chosen plaintexts, by collecting the ciphertexts we fully reconstruct K. This works with complexity $\mathcal{O}(d^2)$ (the time to read K).*

Q.3 Given d known plaintext/ciphertext pairs (X_i, Y_i) for $i = 1, \ldots, d$, propose a key recovery attack of complexity $\mathcal{O}(d^4)$ when $d \to +\infty$ and prove the complexity.
WARNING: d^4 is lower than d^7!
HINT: assume that the X_i vectors are linearly independent!

> *We consider all terms in the first row of K as d unknowns. Looking at the first term of Y_i, we obtain*
>
> $$(Y_i)_1 = \sum_{j=1}^{d} K_{1,j}(X_i)_j$$
>
> *which is a linear equation. So, with d known plaintext/ciphertext pairs, we obtain d linear equations in d unknowns. If the X_i are linearly independent, then the system is regular so we can solve it by inverting a $d \times d$ matrix. In other cases, the system is likely to have a high rank, so we have a small number of solutions that we can enumerate. Later, we can isolate the right one with additional samples. Inverting a matrix can be performed with Gauss elimination in cubic time. So, we have an attack of complexity $\mathcal{O}(d^3)$ to recover the first row of K. We do this for each row and obtain a complexity of $\mathcal{O}(d \times d^3)$. This is better than a straightforward attack looking at the d^2 unknowns directly which would work in $\mathcal{O}(d \times d^6)$.*

3 Attribute-Based Encryption

> *The following exercise is inspired from* Fuzzy Identity-Based Encryption *by Sahai, and Waters, published in the proceedings of EUROCRYPT'05 pp. 457–473, LNCS vol. 3494, Springer 2005.*

We use an *attribute-based* encryption scheme. It allows to encrypt a message respective to a set of attributes att' so that only people having privileges for at least d of these attributes can decrypt the ciphertext. People receive a secret sk corresponding to the list of attributes att that they have. Decryption works only when $\#(\mathsf{att} \cap \mathsf{att}') \geq d$. For instance, an attribute age could represent people over 25, an attribute $\mathsf{licence}$ could represent people owning a driving licence. To rent a car, customers should get an ignition key M which is encrypted for people being over 25 and with a driving licence, so with $\mathsf{att}' = \{\mathsf{age}, \mathsf{licence}\}$. Only people with att including these two privileges should be able to decrypt it and take a car. So, we would set $d = 2$. To use this scheme, an authority generates the master secret msk and the master public key mpk using Setup. Then, it gives attributes att to users and gives them a secret key sk to allow them to decrypt some ciphertexts. Finally, an encryption function using mpk and a set of attributes att' can encrypt messages.

We consider (multiplicative) groups G_1 and G_2 of prime order p and a bilinear map

$$e : G_1 \times G_1 \to G_2$$

We recall that it means that we have

$$e(u^a v^b, w) = e(u, w)^a e(v, w)^b \quad \text{and} \quad e(u, v^a w^b) = e(u, v)^a e(u, w)^b$$

for all $u, v, w \in G_1$ and $a, b \in \mathbf{Z}$. We let g be a generator of G_1. We assume that $e(g, g)$ is a generator of G_2. We consider the following algorithms.

$\mathsf{Setup}(d, n) \to (\mathsf{msk}, \mathsf{mpk})$
1. pick $t_1, \ldots, t_n \in \mathbf{Z}_p^*$ and $y \in \mathbf{Z}_p$ at random
2. set $T_i = g^{t_i}$, $i = 1, \ldots, n$ and $Y = e(g, g)^y$
3. set $\mathsf{mpk} = (d, T_1, \ldots, T_n, Y)$ and $\mathsf{msk} = (t_1, \ldots, t_n, y)$

$\mathsf{Gen}(\mathsf{msk}, \mathsf{att}) \to \mathsf{sk}$ $\quad$ $\{\mathsf{msk} = (t_1, \ldots, t_n, y)$, $\mathsf{att} \subseteq \{1, \ldots, n\}$ non empty$\}$
1. pick some random polynomial $q \in \mathbf{Z}_p[x]$ of degree at most $d - 1$ such that $q(0) = y$ in $\mathbf{Z}_p$
2. set $D_i = g^{\frac{q(i)}{t_i}}$ for $i \in \mathsf{att}$
3. set $\mathsf{sk} = (D_i)_{i \in \mathsf{att}}$ {the list of all D_i for $i \in \mathsf{att}$}

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{att}', M) \to \mathsf{ct}$ $\quad$ $\{\mathsf{mpk} = (d, T_1, \ldots, T_n, Y)$, $\mathsf{att}' \subseteq \{1, \ldots, n\}$ non empty, $M \in G_2\}$
1. pick $s \in \mathbf{Z}_p$ at random
2. set $E' = MY^s$ and $E_i = T_i^s$ for $i \in \mathsf{att}'$
3. set $\mathsf{ct} = (E', (E_i)_{i \in \mathsf{att}'})$ {E' and the list of all E_i for $i \in \mathsf{att}'$}

Q.1 Let $i \neq j$ be two attributes. Show that there exist some $\lambda_{i,j}, \mu_{i,j} \in \mathbf{Z}_p$ such that

$$\forall a, b \in \mathbf{Z}_p \qquad \lambda_{i,j}(ai + b) + \mu_{i,j}(aj + b) = b \pmod{p}$$

> *We let $\lambda_{i,j} = \frac{j}{j-i}$ and $\mu_{i,j} = -\frac{i}{j-i}$ and the property easily follows.*

Q.2 In this question, we assume that $d = 2$.
Specify a decryption algorithm $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}, \mathsf{ct}) \to M'$ such that for all M, att, $i, j \in \mathsf{att}$ such that $i \neq j$, when we run

1: $\mathsf{Setup}(d, n) \to (\mathsf{msk}, \mathsf{mpk})$
2: $\mathsf{Gen}(\mathsf{msk}, \mathsf{att}) \to \mathsf{sk}$
3: $\mathsf{Enc}(\mathsf{mpk}, \{i, j\}, M) \to \mathsf{ct}$
4: $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}, \mathsf{ct}) \to M'$

then we always have $M' = M$.

We have

$$\begin{aligned}\frac{E'}{e(D_i, E_i)^{\lambda_{i,j}} e(D_j, E_j)^{\mu_{i,j}}} &= \frac{E'}{e\left(g^{\frac{q(i)}{t_i}}, g^{t_i s}\right)^{\lambda_{i,j}} e\left(g^{\frac{q(j)}{t_j}}, g^{t_j s}\right)^{\mu_{i,j}}} \\ &= \frac{E'}{e(g, g)^{q(i)\lambda_{i,j} s + q(j)\mu_{i,j} s}} \\ &= \frac{MY^s}{e(g, g)^{ys}} \\ &= M\end{aligned}$$

So the decryption can work like this.

Q.3 More generally, let $I = \{i_1, \ldots, i_d\} \subseteq \{1, \ldots, n\}$ be a subset of size d. Show that there exists a function $\lambda_I : I \to \mathbf{Z}_p$ such that

$$\forall q \in \mathbf{Z}_p[x] \qquad \mathsf{deg}(q) \leq d - 1 \Longrightarrow \lambda_I(i_1)q(i_1) + \cdots + \lambda_I(i_d)q(i_d) = q(0) \pmod p$$

(q is a polynomial of degree up to $d - 1$).

We can easily show that the solution exists by observing that the linear system

$$\lambda_I(i_1)i_1^j + \cdots + \lambda_I(i_d)i_d^j = 1_{j=0}$$

for $j = 0, \ldots, d - 1$ *is non-singular.*
We can also use the Lagrange interpolation polynomials. Let

$$L_{I,i_j}(x) = \prod_{k=1,\ldots,j-1,j+1,\ldots,d} \frac{x - i_k}{i_j - i_k}$$

We have $L_{I,i_j}(i_{j'}) = 1_{j=j'}$ *for all* $j' = 1, \ldots, d$. *So,* $L_{I,i_1}(x)q(i_i) + \cdots + L_{I,i_d}(x)q(i_d)$ *have the same values as* q *on* I. *Since both are polynomials of degree up to* $d-1$ *and both agree on at least* d *points, they must be the same polynomial. So, they match on* $x = 0$ *which yields* $L_{I,i_1}(0)q(i_i) + \cdots + L_{I,i_d}(0)q(i_d) = q(0)$. *Hence,*

$$\lambda_I(i_j) = \prod_{k=1,\ldots,j-1,j+1,\ldots,d} \frac{-i_k}{i_j - i_k}$$

Q.4 Specify a decryption algorithm $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}, \mathsf{ct}) \to M'$ such that for all d, n, M, att, att' such that $\#(\mathsf{att} \cap \mathsf{att}') \geq d$, when we run

1: $\mathsf{Setup}(d, n) \to (\mathsf{msk}, \mathsf{mpk})$
2: $\mathsf{Gen}(\mathsf{msk}, \mathsf{att}) \to \mathsf{sk}$
3: $\mathsf{Enc}(\mathsf{mpk}, \mathsf{att}', M) \to \mathsf{ct}$
4: $\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}, \mathsf{ct}) \to M'$

then we always have $M' = M$.

Let I be an arbitrary subset of $\mathsf{att} \cap \mathsf{att}'$ of cardinality exactly d. We have

$$\begin{aligned}
\frac{E'}{\prod_{i \in I} e(D_i, E_i)^{\lambda_I(i)}} &= \frac{E'}{\prod_{i \in I} e\left(g^{\frac{q(i)}{t_i}}, g^{t_i s}\right)^{\lambda_I(i)}} \\
&= \frac{E'}{e(g,g)^{s \times \sum_{i \in I} q(i) \lambda_I(i)}} \\
&= \frac{MY^s}{e(g,g)^{ys}} \\
&= M
\end{aligned}$$

So the decryption can work like this.