Advanced Cryptography — Final Exam
Solution

Serge Vaudenay

27.6.2013

- duration: 3h00
- any document is allowed
- a pocket calculator is allowed
- communication devices are not allowed
- the exam invigilators will not answer any technical question during the exam
- the answers to each exercise must be provided on separate sheets
- readability and style of writing will be part of the grade
- do not forget to put your name on every sheet!

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

1 ElGamal using a Strong Prime

Let p be a large strong prime. I.e., p is a prime number and $q = \frac{p-1}{2}$ is prime as well.

Q.1 Show that QR_p is a cyclic group.

> *Let h be a generator of $\mathbf{Z}_p^*$. Clearly, h^2 has order q. It further generates only quadratic residues. So, $g = h^2$ is a generator of QR_p.*

Q.2 Show that -1 is not a quadratic residue modulo p.

> *We have $\left(\frac{(-1)}{p}\right) = (-1)^{\frac{p-1}{2}} = (-1)^q = -1$ since q is large and prime. So, the Legendre symbol of -1 is -1. We deduce that -1 is not a quadratic residue modulo p.*

Q.3 Show that there exists a bijection σ from $\{1,\ldots,q\}$ to QR_p, the group of quadratic residues in Z_p^*, such that for all x, $\sigma(x) = x$ or $\sigma(x) = -x$.

> *Actually, $((-x)/p) = ((-1)/p).(x/p) = -(x/p)$. So, $-x$ and $+x$ have opposite Legendre symbols. Since $x \in \mathbf{Z}_p^*$, this is not 0. So, either $-x$ or $+x$ has a Legendre symbol equal to $+1$ but not both. This is the unique quadratic residue $\sigma(x)$.*
> *Clearly, the sets $\{-x,+x\}$ are disjoint for all $x = 1,\ldots,q$. So, the mapping is injective. Now, since half of the elements in $\mathbf{Z}_p^*$ are in QR_p, we have exactly q of them. So, the sets $\{1,\ldots,q\}$ and QR_p have the same cardinality. Therefore, σ is a bijection.*

Q.4 For $m \in \{1,\ldots,q\}$ and $x \in \mathsf{QR}_p$, give algorithms to compute $\sigma(m)$ and $\sigma^{-1}(x)$.

If $m^q \bmod p = 1$, we set $\sigma(m) = m$, otherwise $\sigma(m) = -m$.
If $x \bmod p \leq q$, we set $\sigma^{-1}(x) = x \bmod p$, otherwise $x = p - (x \bmod p)$.

Q.5 We consider the following variant of the ElGamal cryptosystem over the message space $\{1, \ldots, q\}$. Let g be a generator of QR_p. The secret key is $x \in \mathbf{Z}_{p-1}$. The public key is $y = g^x \bmod p$. To encrypt a message m, we pick $r \in \mathbf{Z}_{p-1}$, compute $u = g^r \bmod p$, and $v = \sigma(m)y^r \bmod p$. The ciphertext is the pair (u, v).
Describe the decryption algorithm.

To decrypt (u, v), we compute $\sigma^{-1}(vu^{-x} \bmod p)$. Here, $\sigma^{-1}(x)$ is the only value between $x \bmod p$ and $(-x) \bmod p$ which is lower or equal to q.

Q.6 Show that this variant is IND-CPA secure when the DDH problem is hard in QR_p.

We have seen in class that the ElGamal cryptosystem is IND-CPA secure when messages are group elements and the DDH problem is hard on this group. Here, we have an equivalent cryptosystem over a different message space but with a bijection from this space to the other. So, the same result holds.

2 BLS Signature

This exercise is inspired from Boneh-Lynn-Shacham, Short Signatures from the Weil Pairing, *Journal of Cryptology vol. 17, 2014.*

Let p be a prime number, G and G_T be two groups (with multiplicative notations) of order p, g be a generator of G, and e be a function from $G \times G$ to G_T such that

- (non-degenerate) there exists $a, b \in G$ such that $e(a, b) \neq 1$;
- (efficiently computable) e can be evaluated efficiently;
- (bilinear) $e(ab, c) = e(a, c)e(b, c)$ and $e(a, bc) = e(a, b)e(a, c)$ for all $a, b, c \in G$.

We assume that the size of p is polynomially bounded. We assume that we have efficient algorithms for group multiplication (in both groups), as well as for comparing group elements. We assume that a random oracle H maps any bitstring to a group element in G. We define a signature scheme as follows:

key generation: we pick the secret key $x \in \mathbf{Z}_p$ and the public key is $v = g^x$;
signature algorithm: to sign a message m, we produce $\sigma = H(m)^x$;
verification algorithm: to verify (v, m, σ), we check that $e(g, \sigma) = e(v, H(m))$.

Q.1 Show that $e(g^x, g^y) = e(g, g)^{xy}$ for all $x, y \in \mathbf{Z}_p$.

By induction on y, we have $e(g, g^y) = e(g, g)^y$, due to bilinearity. By induction on x, we have $e(g^x, g^y) = e(g, g^y)^x$, due to bilinearity. So, $e(g^x, g^y) = e(g, g)^{xy}$.

Q.2 Show that the algorithms in the signature scheme are efficient and that produced signatures are always correct.

Since p has a polynomially bounded length and that the multiplication in G is easy, it is easy to generate a key pair. Since an oracle access costs one unit and that the multiplication in G is easy, it is easy to sign. Since e is easy to evaluate and that G_T elements are easy to compare, it is easy to verify a signature.
If the signature is produced by the signing algorithm, we have

$$e(g, \sigma) = e(g, H(m)^x) = e(g, H(m))^x = e(g^x, H(m)) = e(v, H(m))$$

So, the signature is correct.

Q.3 Show that the Decisional Diffie-Hellman (DDH) problem is easy to solve in G.

Given (g, X, Y, Z), we define $\mathcal{A}(g, X, Y, Z) = 1$ if and only if $e(g, Z) = e(X, Y)$. Clearly, $\mathcal{A}(g, g^x, g^y, g^{xy}) = 1$ whatever x and y. Now, $\Pr[\mathcal{A}(g, g^x, g^y, g^z) = 1] = \Pr[e(g, g)^z = e(g, g)^{xy}]$. Since e is non-degenerate, $e(g, g)$ must have order p in G_T (the order is 1 or p, since p is prime, and it cannot be 1, otherwise e would be degenerate). So, $\Pr[\mathcal{A}(g, g^x, g^y, g^z) = 1] = \Pr[z \equiv xy \pmod p]$. For, x, y, z uniformly distributed in $\mathbf{Z}_p$, this occurs with probability $\frac{1}{p}$. So, the advantage of the DDH solver is $1 - \frac{1}{p}$ which is large: the DDH problem is easy to solve.

Q.4 For an attack using no chosen message, show that making an existential forgery implies solving the Computational Diffie-Hellman (CDH) problem. More precisely, given an algorithm $\mathcal{A}^H(g,v) = (m,\sigma)$ forging a valid signature σ for m under public key v with oracle access to H, we can construct an algorithm $\mathcal{B}(g, g^x, Y)$ to compute Y^x, with complexity comparable to the one of $\mathcal{A}$ and a polynomially bounded overhead. (Assume $\mathcal{A}$ works with probability 1.)
Hint: simulate $H(m')$ by $g^{r(m')}Y$ where r is a random function from $\{0,1\}^*$ to $\mathbf{Z}_p$.

> *We rather use $H(m') = g^{r(m')}Y^{s(m')}$, with $s(m') = 1$ for the moment. In a further question, s will be introduced.*
> *We define $\mathcal{B}(g, X, Y)$ by picking m and simulating $\mathcal{A}(g, X)$. When the $\mathcal{A}$ simulator makes a query m' to H, $\mathcal{B}$ answers by $H(m')$ as suggested by the hint. Clearly, a valid signature is a value $\sigma = g^{xr(m)}Y^{xs(m)}$. So, $Y^x = (\sigma g^{-xr(m)})^{\frac{1}{s(m)}}$ is the solution to the Diffie-Hellman problem.*

Q.5 If now $\mathcal{A}$ works with probability ρ over the uniform distribution of X and H in G, show that we can construct some $\mathcal{B}'$ working with probability ρ as well, for any x and y.

> *To make $\mathcal{B}'(g, X, Y)$ work whatever x and y, we must randomize the inputs to the $\mathcal{A}$ simulator. We now take $v = g^u X$ for some random $u \in \mathbf{Z}_p$. We obtain that v is uniformly distributed in G. If $\mathcal{A}$ gives a correct signature σ, we have*
>
> $$\sigma = (g^{r(m)}Y^{s(m)})^{u+x} = g^{ur(m)}Y^{r(m)+us(m)}Y^{xs(m)}$$
>
> *So, the solution to the Diffie-Hellman problem is now $(\sigma g^{-ur(m)}Y^{-r(m)-us(m)})^{\frac{1}{s(m)}}$.*
> *The $\mathcal{A}$ simulator will work with probability ρ. So, $\mathcal{B}'$ works with probability ρ as well.*

Q.6 Show that by selecting a biased function s from $\{0,1\}^*$ to $\{0,1\}$ and by now simulating H by $H(m') = g^{r(m')}Y^{s(m')}$, we can introduce chosen message attacks in the previous result: making existential forgeries under chosen message attacks implies solving the CDH problem. (The probability of the solving algorithm may be different though.)

> *To be able to answer to a signing query m', we should have $s(m') = 0$. Indeed, the signature $H(m')^{u+x}$ is $g^{(u+x)r(m')} = g^{ur(m')}Y^{r(m')}$ which can be computed. To be able to forge a signature on m, we need to have $s(m) = 1$. So, we take a random function s such that $\Pr[s(m) = 1] = \theta$ and $\Pr[s(m) = 0] = 1 - \theta$. Each signing query can be honored with probability $1 - \theta$. At the end, we have a forgery with probability θ. So, if Q is the total number of signing queries, the probability of success is $(1-\theta)^Q\theta$. By taking $\theta = \frac{1}{Q+1}$, this is a pretty good probability.*

3 PRF Programming

This exercise is inspired from Boureanu-Mitrokotsa-Vaudenay, On the Pseudorandom Function Assumption in (Secure) Distance-Bounding Protocols - PRF-ness alone Does Not Stop the Frauds!*, in LATINCRYPT 2012, LNCS vol. 7533, Springer.*

A function $\delta(s)$ is called negligible and we write $\delta(s) = \mathsf{negl}(s)$ if for any $c > 0$, we have $|\delta(s)| = o(s^{-c})$ as s goes to $+\infty$.

Let s be a security parameter. For simplicity of notations, we do not write s as an input of games and algorithms but it *is* a systematic input.

A family $(f_k)_{k\in\{0,1\}^s}$ of functions f_k from $\{0,1\}^s$ to $\{0,1\}^s$ is called a PRF (Pseudo Random Function) if for any probabilistic polynomial-time oracle algorithm $\mathcal{A}$, we have that

$$|\Pr[\mathcal{A}^{f_K(\cdot)} = 1] - \Pr[\mathcal{A}^{f^*(\cdot)} = 1]| = \mathsf{negl}(s)$$

where $K \in \{0,1\}^s$ is uniformly distributed, f^* is a uniformly distributed function from $\{0,1\}^s$ to $\{0,1\}^s$, $f_K(\cdot)$ denotes the oracle returning $f_K(x)$ upon query x, and $f^*(\cdot)$ denotes the oracle returning $f^*(x)$ upon query x.

Given a PRF $(f_k)_{k\in\{0,1\}^s}$, we construct a family $(g_k)_{k\in\{0,1\}^s}$ by $g_k(x) = f_k(x)$ if $x \neq k$ and $g_k(k) = k$. The goal of the exercise is to prove that $(g_k)_{k\in\{0,1\}^s}$ is a PRF.

We define the PRF game played by $\mathcal{A}$ for g, f, and f^* by

Game Γ^g
1. pick $K \in \{0,1\}^s$
2. run $b = \mathcal{A}^{g_K(\cdot)}$
3. give b as output

Game Γ^f
1. pick $K \in \{0,1\}^s$
2. run $b = \mathcal{A}^{f_K(\cdot)}$
3. give b as output

Game Γ^*
1. pick $f^* : \{0,1\}^s \to \{0,1\}^s$
2. run $b = \mathcal{A}^{f^*(\cdot)}$
3. give b as output

For each integer i, we define an algorithm $\mathcal{A}_i$ (called a *hybrid*) which mostly simulates $\mathcal{A}$ until it makes the ith query. More concretely, $\mathcal{A}_i$ simulates every step and queries of $\mathcal{A}$ while counting the number of queries. When the counter reaches the value i, $\mathcal{A}_i$ does not make this query k but it stops and the queried value k is returned as the output of $\mathcal{A}_i$. If $\mathcal{A}$ stops before making i queries, $\mathcal{A}_i$ stops as well, with a special output $\perp$. We define the following games:

Game Γ^f_i
1. pick $K \in \{0,1\}^s$
2. run $k = \mathcal{A}_i^{f_K(\cdot)}$
3. if $k = \perp$, stop and output 0
4. pick $x \in \{0,1\}^s$
5. if $f_k(x) = f_K(x)$, stop and output 1
6. output 0

Game Γ^*_i
1. pick $f^* : \{0,1\}^s \to \{0,1\}^s$
2. run $k = \mathcal{A}_i^{f^*(\cdot)}$
3. if $k = \perp$, stop and output 0
4. pick $x \in \{0,1\}^s$
5. if $f_k(x) = f^*(x)$, stop and output 1
6. output 0

Let $F(\Gamma)$ be the event that any of the queries by $\mathcal{A}$ in game Γ equals K. We assume that the number of queries by $\mathcal{A}$ is bounded by some polynomial $P(s)$.

Q.1 Show that $|\Pr[\Gamma^f \to 1] - \Pr[\Gamma^* \to 1]| = \mathsf{negl}(s)$.

This is a direct consequence of the definition of the PRF, for f.

Q.2 Show that $\Pr[\Gamma^g \rightarrow 1 | \neg F(\Gamma^g)] = \Pr[\Gamma^f \rightarrow 1 | \neg F(\Gamma^f)]$ and $\Pr[\neg F(\Gamma^g)] = \Pr[\neg F(\Gamma^f)]$.

We run Γ^g and Γ^f with the same coins for K and $\mathcal{A}$. By induction, $\mathcal{A}$ produce identical queries in both games and g and f produce identical answers. This is until the ith query. Since the outcome of the game does not depend on the answer to the ith query, it is identical for both games. So, $\Pr[\Gamma^g \rightarrow 1 | \neg F(\Gamma^g)] = \Pr[\Gamma^f \rightarrow 1 | \neg F(\Gamma^f)]$. as same coins produce identical outcomes. Similarly, $\Pr[\neg F(\Gamma^g)] = \Pr[\neg F(\Gamma^f)]$.

Q.3 Deduce $|\Pr[\Gamma^g \rightarrow 1] - \Pr[\Gamma^f \rightarrow 1]| \leq \Pr[F(\Gamma^f)]$.

We have

$$\Pr[\Gamma^g \rightarrow 1] = \Pr[\neg F(\Gamma^g)] \Pr[\Gamma^g \rightarrow 1 | \neg F(\Gamma^g)] + \Pr[\Gamma^g \rightarrow 1 \wedge F(\Gamma^g)]$$

and the same with f. So, by difference, due to the previous question, we have

$$\begin{aligned} |\Pr[\Gamma^g \rightarrow 1] - \Pr[\Gamma^f \rightarrow 1]| &\leq \max(\Pr[\Gamma^g \rightarrow 1 \wedge F(\Gamma^g)], \Pr[\Gamma^f \rightarrow 1 \wedge F(\Gamma^f)]) \\ &\leq \max(\Pr[F(\Gamma^g)], \Pr[F(\Gamma^f)]) \\ &\leq \Pr[F(\Gamma^f)] \end{aligned}$$

Q.4 Show that $\Pr[F(\Gamma^f)] \leq \sum_{i=1}^{P(s)} \Pr[\Gamma_i^f \rightarrow 1]$.

To any case where $F(\Gamma^f)$ occurs, we can define the index i of the first query equal to K and have $\Gamma_i^f \rightarrow 1$ with the same coins. So,

$$\Pr[F(\Gamma^f)] \leq \Pr\left[\bigvee_{i=1}^{P(s)} \Gamma_i^f \rightarrow 1\right] \leq \sum_{i=1}^{P(s)} \Pr[\Gamma_i^f \rightarrow 1]$$

Q.5 Show that $|\Pr[\Gamma_i^f \rightarrow 1] - \Pr[\Gamma_i^* \rightarrow 1]| = \mathsf{negl}(s)$ for all $i \leq P(s)$.

We define a new adversary $\mathcal{A}'_i$ who simulates $k = \mathcal{A}_i$, then picks $x \in \{0,1\}^s$, then query the oracle with x, then output 1 if and only if the response equals $f_k(x)$. We apply the PRF assumption on $\mathcal{A}'_i$ and obtain $\Pr[\Gamma_i^* \rightarrow 1] = \mathsf{negl}(s)$.

Q.6 Show that $\Pr[\Gamma_i^* \rightarrow 1] = \mathsf{negl}(s)$ for all $i \leq P(s)$.

If x is a fresh query at the end of the Γ_i^* game, $f^*(x)$ if uniformly distributed and independent from $f_k(x)$. So, $f_k(x) = f^*(x)$ with probability 2^{-s} in that case. Now, since x is picked at random, the probability that it is not fresh if bounded by $P(s) \times 2^{-s}$. Overall, we obtain that $\Pr[\Gamma_i^* \rightarrow 1] \leq (P(s)+1)2^{-s}$ which is negligible.

Q.7 Deduce $|\Pr[\Gamma^g \rightarrow 1] - \Pr[\Gamma^* \rightarrow 1]| = \mathsf{negl}(s)$.

We have

$$
\begin{aligned}
& |\Pr[\Gamma^g \to 1] - \Pr[\Gamma^* \to 1]| \\
\le\ & |\Pr[\Gamma^g \to 1] - \Pr[\Gamma^f \to 1]| + |\Pr[\Gamma^f \to 1] - \Pr[\Gamma^* \to 1]| \\
\le\ & |\Pr[\Gamma^g \to 1] - \Pr[\Gamma^f \to 1]| + \mathsf{negl}(s) & \text{(Q.1)} \\
\le\ & \Pr[F(\Gamma^f)] + \mathsf{negl}(s) & \text{(Q.3)} \\
\le\ & \textstyle\sum_{i=1}^{P(s)} \Pr[\Gamma_i^f \to 1] + \mathsf{negl}(s) & \text{(Q.4)} \\
\le\ & \textstyle\sum_{i=1}^{P(s)} (\Pr[\Gamma_i^* \to 1] + \mathsf{negl}(s)) & \text{(Q.5)} \\
\le\ & \textstyle\sum_{i=1}^{P(s)} \mathsf{negl}(s) & \text{(Q.6)} \\
\le\ & \mathsf{negl}(s)
\end{aligned}
$$

So, g is a PRF as well.