Solutions to Problem Set 2

1. Suppose that $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$ is a (t,ϵ) secure pseudorandom function.

   Consider the following randomized MAC scheme (we shall assume for simplicity that m is a multiple of 3 and the scheme is defined only for messages whose length is a multiple of $m/3$ and is at most $\frac{m}{3} \cdot 2^{m/3-1}$):

   - $Tag(K,M)$
     - divide M into blocks $M_1,\ldots,M_\ell$ of length $m/3$
     - pick a random $r \in \{0,1\}^{m/3}$
     - return $(r, F_K(r,0,1,M_1), F_K(r,0,2,M_2), \ldots, F_K(r,1,\ell,M_\ell))$
   - $Verify(K,M,(r,f_1,\ldots,f_\ell))$
     - divide M into blocks $M_1,\ldots,M_\ell$ of length $m/3$
     - check that for every $i \in \{1,\ldots,\ell-1\}$ we have $f_i = F_K(r,0,i,M_i)$ and that we have $f_\ell = F_K(r,1,\ell,M_\ell)$.

   Show that this scheme is $(t/O(L), \epsilon + t^2 \cdot 2^{-m/3} + 2^{-m})$-secure, where L is an upper bound to the length of the messages that we are going to authenticate.

   Solution. We repeat the analysis we did in Section 3 of Lecture 7. The only thing that changes is, at the very end, the second case of the case analysis.

   Let A be an algorithm running in time $t' = t/O(L)$ and mounting a chosen message attack which is able to forge the MAC scheme with probability δ. Consider the MAC $\overline{T},\overline{V}$ which is identical to the scheme described above except that it uses a purely random function instead of a pseudorandom function. Let A' be the algorithm that given a function oracle F simulates A and simulates every authentication query made by A by running the above Tag algorithm with the oracle F instead of the pseudorandom function. At the end, A' outputs 1 if and only if the simulation of A has produced a valid forgery. Note that A' runs in time $\leq t' \cdot L < t$, and so we must have

   $$|\mathop{\mathbb{P}}_K[A'^{F_K()}() = 1] - \mathop{\mathbb{P}}_{R:\{0,1\}^m \to \{0,1\}^m}[A'^{R()}() = 1]| \leq \epsilon$$

   because of the pseudorandomness of F_K.

This implies that

$$\mathop{\mathbb{P}}_{R}[A'^{R}() = 1] = \mathop{\mathbb{P}}_{R}[A^{\overline{T},\overline{V}} \text{ outputs a forged MAC for } \overline{T},\overline{R}] \geq \delta - \epsilon$$

It remains to show that the probability that an algorithm A of running time $t' \leq t$ can produce a forgery for $(\overline{T},\overline{V})$ is at most $t^2 \cdot 2^{-m/3} + 2^{-m}$.

We may assume that A never queries the Tag oracle twice on the same message. Let $FORGE$ be the event that A finds a valid forgery for $\overline{T},\overline{V}$. Consider the event REP that, during the execution of A, the random strings r used by the tagging algorithm are not all different. Note that A can query messages of total length at most t (because it runs in total time at most t), and so

$$\mathop{\mathbb{P}}_{R}[REP] \leq \frac{t^2}{2^{m/3}}$$

Now, consider what happens when we have $FORGE \wedge \neg REP$, that is, A'^{R}, simulating $A^{\overline{T},\overline{V}}$, uses different random strings r in each simulated invocation of $\overline{T}$, and it produces a valid forgery $(r, T_1, \ldots, T_\ell)$ of a new message $(M_1, \ldots, M_\ell)$ at the end. We claim that, in such a case, A' correctly guesses the value of R at an input for which $R()$ had not been evaluated before. Once we prove the claim, we immediately get

$$\mathbb{P}[FORGE \wedge \neg REP] \leq \frac{1}{2^m}$$

and so

$$\mathop{\mathbb{P}}_{R}[FORGE] \leq \mathop{\mathbb{P}}_{R}[FORGE \wedge \neg REP] + \mathop{\mathbb{P}}_{R}[REP] \leq \frac{t^2}{2^{m/3}} + \frac{1}{2^m}$$

as needed.

It remains to Prove the claim. Call $M^1, \ldots, M^q$ the messages that A' authenticates with $\overline{T}$, and let M be the forgery at the end. Let $r^1, \ldots, r^q, r$ be the random strings used in the tagging of $M^1, \ldots, M^q, M$, respectively. We consider two cases:

(a) If r is different from all the r^i, then the first block T_1 in the forged tag $(r, T_1, \ldots, T_\ell)$ of M contains the value $R(r, 0, 1, M_1)$ which was never queried before to the $R()$ oracle.

(b) If r is equal to some of the r^j, then it can be equal to exactly one r^j, because the random strings r^j are different from each other. (Recall that we are considering a computation of A' that satisfied the event $FORGE \wedge \neg REP$.)

Now compare M with M^j. If M and M^j have the same length (measured as number of blocks of length $m/3$ each) ℓ, then when we write $M = M_1, \ldots, M_\ell$ and $M^j = M_1^j, \ldots, M_\ell^j$, there must be a block i such that $M_i \neq M_i^j$. (Otherwise we would have $M = M^j$ which cannot be because we are considering a case that satisfied the event $FORGE$.) Then the block T_i in the forget tag of M is the correct evaluation of $R()$ at a point that had not been queried before.

Finally, if M and M^j have different lengths, let ℓ' be the shortest of the two lengths, and observe that $T_{\ell'}$ is the correct evaluation of $R()$ at a point that had not been queried before.

2. Fix a randomized algorithm P (for "padding") that on input a string in $\{0,1\}^m$ runs in time $\leq r$ and outputs another string in $\{0,1\}^m$. Let (Enc, Dec) be an encryption scheme that encrypts blocks of length $2m$, and cosider the modified encryption scheme $(PEnc, PDec)$ defined so that a message M is first padded by appending $P(M)$ and then it is encrypted with Enc:

   - $PEnc(K, M) := Enc(K, (M, P(M)))$'
   - $PDenc(K, C)$:
     - $(M_1, M_2) := Dec(K, C)$
     - return M_1

   Prove that

   (a) If (Enc, Dec) is (t, ϵ)-message indistinguishable, then $(PEnc, PDec)$ is (t, ϵ)-message indistinguishable.

   [Hint: you may find it easier to first argue the case in which P is deterministic.]

   Solution. Suppose $(PEnc, PDec)$ is not (t, ϵ) message indistinguishable, so that there are messages m_0, m_1 and an algorithm A of complexity $\leq t$ such that

$$|\,\mathbb{P}[A(PEnc(m_0)) = 1] - \mathbb{P}[A(PEnc(m_1)) = 1]| > \epsilon$$

   This is equivalent to

$$|\,\mathbb{P}[A(Enc(m_0, P(m_0))) = 1] - \mathbb{P}[A(Enc(m_1, P(m_1))) = 1]| > \epsilon$$

   If $P()$ is deterministic, then the algorithm A and the plaintexts $(m_0, P(m_0))$ and $(m_1, P(m_1))$ contradict the (t, ϵ) message indistinguishability of (Enc, Dec).

If $P()$ is probabilistic, then we can write $P_r(m)$ for the output of $P()$ when taking the input m and using internal randomness r. Then we have

$$|\,\mathbb{P}[A(Enc(m_0, P_r(m_0))) = 1] - \mathbb{P}[A(Enc(m_1, P_r(m_1))) = 1]| > \epsilon \qquad (1)$$

where the probability is over the randomness of Enc and over the random choice of r.

We can rewrite (1) as

$$\left|\mathop{\mathbb{E}}_r(\mathbb{P}[A(Enc(m_0, P_r(m_0))) = 1] - \mathbb{P}[A(Enc(m_1, P_r(m_1))) = 1])\right| > \epsilon \qquad (2)$$

and, using the triangle inequality,

$$\mathop{\mathbb{E}}_r |\,\mathbb{P}[A(Enc(m_0, P_r(m_0))) = 1] - \mathbb{P}[A(Enc(m_1, P_r(m_1))) = 1]| > \epsilon$$

so that there must exist a particular choice of r, say r_0 such that

$$|\,\mathbb{P}[A(Enc(m_0, P_{r_0}(m_0))) = 1] - \mathbb{P}[A(Enc(m_1, P_{r_0}(m_1))) = 1]| > \epsilon$$

and so the algorithm A and the messages $(m_0, P_{r_0}(m_0))$ contradict the (t, ϵ) message indistinguishability of (Enc, Dec).

(b) If (Enc, Dec) is (t, ϵ) CPA secure, then $(PEnc, PDec)$ is $(t/r, \epsilon)$ CPA secure.

Solution. Suppose $(PEnc, PDec)$ is not $(t/r, \epsilon)$ CPA secure, so that there are messages m_0, m_1 and an algorithm A of complexity $\leq t/r$ such that

$$|\,\mathbb{P}[A^{PEnc}(PEnc(m_0)) = 1] - \mathbb{P}[A^{PEnc}(PEnc(m_1)) = 1]| > \epsilon \qquad (3)$$

Consider the oracle algorithm A' that on input a ciphertext C and given an oracle E, simulates $A(C)$; every time A makes an oracle queries m_i, A' simulates it with the outcome of the query $E(m_i, P(m_i))$, where E is the oracle given to A'. Note that if P is computable in time r, and A runs in time t/r, then A' runs in time $\leq t$. Expression (3) becomes

$$\mathbb{P}[A'^{Enc}(Enc(m_0, P(m_0))) = 1] - \mathbb{P}[A^{Enc}(Enc(m_1, P(m_1))) = 1]| > \epsilon$$

If P is deterministic, then the messages $(m_0, P(m_0))$ and $(m_1, P(m_1))$ and the algorithm A' contradict the (t, ϵ) CPA security of (Enc, Dec). If $P()$ is probabilistic, we can use the same averaging trick we used in part (a).

(c) If (Enc, Dec) is (t, ϵ) CCA secure, then $(PEnc, PDec)$ is $(t/O(r), \epsilon)$ CCA secure.

Solution. Suppose $(PEnc, PDec)$ is not $(t/O(r), \epsilon)$ CCA secure, so that there are messages m_0, m_1 and an algorithm A of complexity $\leq t/O(r)$ such that

$$|\,\mathbb{P}[A^{PEnc,PDec}(PEnc(m_0)) = 1] - \mathbb{P}[A^{PEnc,PDec}(PEnc(m_1)) = 1]| > \epsilon \quad (4)$$

Consider the oracle algorithm A' that on input a ciphertext C and given oracle E, D, simulates $A(C)$ as follows:

- every time A makes an oracle queries m_i to $PEnc$, A' simulates it with the outcome of the query $E(m_i, P(m_i))$, where E is the first oracle given to A';
- every time A makes an oracle query C_i to $PDec$, A' simulates it by querying C_i into its second oracle D, receiving a pair (m_i, P_i) as an answer, and it continues the simulation as if m_i had been the query returned by $PDec$ to A.

Note that if P is computable in time r, and A runs in time $t/O(r)$, then A' runs in time $\leq t$. Expression (4) becomes

$$\mathbb{P}[A'^{Enc,Dec}(Enc(m_0, P(m_0))) = 1] - \mathbb{P}[A^{Enc,Dec}(Enc(m_1, P(m_1))) = 1]| > \epsilon$$

If P is deterministic, then the messages $(m_0, P(m_0))$ and $(m_1, P(m_1))$ and the algorithm A' contradict the (t, ϵ) CPA security of (Enc, Dec). If $P()$ is probabilistic, we can use the same averaging trick we used in parts (a) and (b).