Problem Set 2 — Solutions

Thanks to Dov Gordon for his help with these solutions

1. We use counter-mode encryption, but use the fixed nonce ’1’ rather than a random nonce. More formally, let $F$ be a pseudorandom function that (for security parameter $n$) maps $n$-bit strings to $n$-bit strings. Then the encryption of a message $m = m_1 \| \cdots \| m_\ell$ (with $|m_i| = n)$ using key $k$ is given by:

$$m_1 \oplus F_k(\langle 1 \rangle) \| m_2 \oplus F_k(\langle 2 \rangle) \| \cdots \| m_\ell \oplus F_k(\langle \ell \rangle),$$

where $\langle i \rangle$ denotes the $n$-bit representation of the integer $i$. Decryption is done in the obvious way.

This scheme handles arbitrary-length messages (that are a multiple of the block-length, $n$) and a proof that it has indistinguishable encryptions in the presence of an eavesdropper is essentially as in class. (The only potential problem is a “wrap-around” in the counter, but this only occurs if the message has block-length greater than $2^n$. A polynomial-time adversary cannot output a message this long for $n$ sufficiently large.) Be sure that you would be able to write such a proof, if asked, on an exam!

The scheme is trivially insecure against a multi-message attack since it is deterministic.

2. (With help from a large hint in Goldreich’s book [Chapter 5, exercise 33])

We start with the scheme $(\text{Enc}, \text{Dec})$ we saw in class: Let $F$ be a pseudorandom function, and define $\text{Enc}_k(m)$ as follows: choose $r \leftarrow \{0,1\}^n$, and output $\langle r, F_k(r) \oplus m \rangle$. We modify this encryption scheme in the following way. Keys are now $2n$ bits long (parsed as two $n$-bit strings $k, s$) and encryption is defined as:

$$\text{Enc}_k^*(m) = \begin{cases} \langle 0, s, \text{Enc}_k(m) \rangle & \text{if } m \neq s \\ \langle 1, k, \text{Enc}_k(m) \rangle & \text{if } m = s \end{cases}$$

Decryption simply ignores the first two components of the ciphertext.

It is easy to see that this scheme is not secure against chosen-plaintext attacks. Using two adaptively-chosen queries to the encryption oracle, the adversary can recover $k$, at which point the scheme is completely broken.

Consider the adversary that attempts to distinguish whether a vector of ciphertexts corresponds to the encryption of the vector $(m_1^0, \ldots, m_\ell^0)$ or the vector $(m_1^1, \ldots, m_\ell^1)$. (Where these vectors are both output at once.) It is not too hard to see that, unless there exists an $i, b$ with $m_i^b = s$, the modified encryption $\text{Enc}'$ is as secure as the original encryption $\text{Enc}$. Because $s$ is a randomly-chosen $n$-bit string, and all the messages are output by the adversary before it has any information about $s$, the probability that there exists an $i, b$ with $m_i^b = s$ is negligible.

This can easily be turned into a proof that $(\text{Enc}', \text{Dec}')$ is secure in the sense of multi-message indistinguishability: Let $\mathcal{A}'$ be a PPT adversary attacking $\Pi' = (\text{Enc}', \text{Dec}')$ in the sense of multi-message indistinguishability, and construct the following PPT adversary $\mathcal{A}$ attacking $\Pi$
in the same sense: $\mathcal{A}$ runs $\mathcal{A}'$, obtains two vectors of messages, and outputs these vectors. When $\mathcal{A}$ is given a vector of ciphertexts $(c_1,\ldots,c_\ell)$, it chooses a random $s \leftarrow \{0,1\}^n$ and gives to $\mathcal{A}'$ the vector $((0,s,c_1),\ldots,(0,s,c_\ell))$. Then $\mathcal{A}$ outputs whatever “guess” is output by $\mathcal{A}'$.

Because the view of $\mathcal{A}'$, above, is only different from its view when attacking $\Pi'$ if $s \in \{m_i^b\}$, we have

$$\Pr[\mathcal{A} \text{ guesses correctly when attacking } \Pi] \geq \Pr[\mathcal{A}' \text{ guesses correctly when attacking } \Pi'] - \Pr[s \in \{m_i^b\}] .$$

We have already noted that $\Pr[s \in \{m_i^b\}]$ is negligible. Since security of $\Pi$ implies that

$$\Pr[\mathcal{A} \text{ guesses correctly when attacking } \Pi] \leq \frac{1}{2} + \text{negl}(n)$$

for some negligible function $\text{negl}$, we have

$$\Pr[\mathcal{A}' \text{ guesses correctly when attacking } \Pi'] \leq \frac{1}{2} + \text{negl}'(n),$$

for some negligible function $\text{negl}'$. This shows that $\Pi'$ is secure in the desired sense.

3. Say nonces $r$ and $r'$ overlap if $|r - r'| < \ell(n)$. A proof of security boils down to showing that the probability that some pair of nonces overlap is negligible. (Make sure you understand why this is the case!)

Let overlap$_{i,j}$ denote the event that nonces $r_i$ and $r_j$ overlap, and let Overlap denote the event that some pair of nonces overlap. Note that $\Pr[\text{overlap}_{i,j}] = (2\ell(n) - 1)/2^n$, assuming each nonce is uniformly-random $n$-bit string.

Then

$$\Pr[\text{Overlap}] = \Pr\left[\bigvee_{i \neq j} \text{overlap}_{i,j}\right] \leq \sum_{i \neq j} \Pr[\text{overlap}_{i,j}] = \sum_{i \neq j} \frac{2\ell(n) - 1}{2^n} = \binom{q(n)}{2} \cdot \left(\frac{2\ell(n) - 1}{2^n}\right),$$

since $q(n)$ nonces are chosen. This is negligible in $n$, concluding the proof.

4. The adversary queries the oracle with some (arbitrary) message $m$ of length $n$, where $n$ is the input/output length of the PRF $F_k$. He receives in response a tag $MAC_k(m) = F_k(0^n \oplus m) = F_k(m)$. He then queries the message $m||0^n$ and receives the tag $MAC_k(m||0^n) = F_k(F_k(m))$. Finally, he outputs the (message, tag) pair $(F_k(m),F_k(F_k(m)))$. Note that the adversary had never queried the oracle with the message $F_k(m)$, and $MAC_k(F_k(m)) = F_k(F_k(m))$, so this is a forgery.

5. (a) The scheme in the problem is secure. To formally prove this, we need to modify the standard experiment defining security of a message authentication code. Consider the following experiment:

i. A random key $k$ is chosen.
ii. The adversary $A$ gets to specify some (polynomial) length $i^*$, and then gets to interact with an oracle that computes CBC-MAC using key $k$ for messages of block-length $i^*$.

iii. The adversary succeeds if it outputs a message/tag pair $(m, t)$ such that (1) $m$ has block-length $i^*$; (2) $m$ was never queried to the MAC oracle; and (3) $t$ is a CBC-MAC tag on $m$ with respect to key $k$.

Although we did not explicitly state this in class, it can be shown that if $F$ is a pseudo-random function then any ppt adversary $A$ succeeds with only negligible probability in the above experiment. (In class, we assumed the length was fixed, not chosen by $A$.)

Say we have an adversary $A'$ attacking the variant of CBC-MAC as in the problem. Let $\epsilon(n)$ be the probability that $A'$ succeeds in outputting a forgery. We construct an adversary $A$ as follows: first, it guesses a random $i^* \leftarrow \{1, \ldots, \ell\}$ and outputs it. Then it chooses keys $k_i \leftarrow \{0, 1\}^n$ for all $i \neq i^*$, and runs $A'$. When $A'$ requests a MAC for a message $m$, there are two cases:

- If $m$ has length $i^*$, then $A$ requests a MAC on $m$ from its own MAC oracle and returns the result to $A'$.
- If $m$ has length $i \neq i^*$, then $A$ computes the MAC on its own using key $k_i$.

When $A'$ outputs $(m, t)$, if $m$ has length $i^*$ then $(m, t)$ is output by $A$.

Note that (1) $A$ carries out a valid attack on the original CBC-MAC (as discussed above), and (2) $A$ provides a perfect simulation for $A'$. Since $i^*$ is chosen at random and is independent of the view of $A'$, the probability that its final output $(m, t)$ has length $i^*$ is $1/\ell(n)$ and the probability that $A$ outputs a forgery is $\epsilon(n)/\ell(n)$. Because this must be negligible (by security of regular CBC-MAC), we conclude that $\epsilon$ is negligible as well.

(b) Let $k$ be a key of length $n$, and let $F$ be a pseudorandom function. Then to compute a MAC on a message $m$ of length $i$, do:

i. Set $k_i := F_k(i)$.
ii. Compute a CBC-MAC on $m$ using key $k_i$.

We leave the proof that this is secure as an exercise.