Problem Set 4

Problem 1: Short tags

Say $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$ is a secure MAC, and for $k \in \{0,1\}^n$ the tag-generation algorithm Mac_k always outputs tags of length $t(n)$. Prove that t must be super-logarithmic or, equivalently, that if $t(n) = O(\log n)$ then Π cannot be a secure MAC

Hint: Consider the probability of randomly guessing a valid tag.

Problem 2: A simple MAC from a PRF

Consider the following MAC for messages of length $\ell(n) = 2n - 2$ using a pseudorandom function F: On input a message $m_0 \| m_1$ (with $|m_0| = |m_1| = n - 1$) and key $k \in \{0,1\}^n$, algorithm Mac outputs $t = F_k(0\|m_0)\|F_k(1\|m_1)$. Algorithm Vrfy is defined in the natural way. Is $(\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$ secure? Prove your answer.

Problem 3: Modified CBC-MAC

Prove that the following modifications of basic CBC-MAC do not yield a secure MAC (even for fixed-length messages):

1. Mac outputs all blocks $t_1, \ldots, t_\ell$, rather than just t_ℓ. (Verification only checks whether t_ℓ is correct.)

2. A random initial value is used each time a message is authenticated. That is, $t_0 \in \{0,1\}^n$ is chosen uniformly at random rather than being fixed to 0^n, and the tag is $\langle t_0, t_\ell \rangle$. Verification is done in the natural way.

Problem 4: The birthday attack

1. Assume that people's birthdays (the dates without the year) are independent and uniformly distributed among the dates except February 29th. What is the smallest number of people such that the probability that two of them have the same birthday is larger than 99%?

2. Consider the birthday attack on a hash function $H : \{0,1\}^* \to \{0,1\}^n$, i.e. the attack by calculating $H(x)$ for $2^{n/2}$ random values of x and checking for collisions. This attack uses $(m+n)2^{n/2}$ bits of memory, assuming we use m-bit inputs. In this blog post, Randall Munroe estimates Google's total memory to be 15 exabytes. How do you have to choose the output length of your hash function to prevent a birthday attack by Google? Assume that Google uses $m \geq n$ and provide a length n along with a proof that it is sufficient to prevent an attack.

Problem 5: HMAC?

Let $H : \{0,1\}^* \to \{0,1\}^n$ be a collision-resistant hash function. Use H to construct a collision-resistant hash function H' such that the MAC function given as $\mathsf{MAC}_k(m) := H'(k \oplus m_0 \| m_1)$, where $m = m_0 \| m_1$ and $|m_0| = |k|$ is insecure.

Problem 6: Authenticate-then-encrypt

Given a CPA-secure encryption scheme $(\mathsf{Gen}_1, \mathsf{Enc}, \mathsf{Dec})$ and a MAC $(\mathsf{Gen}_2, \mathsf{MAC}, \mathsf{Vrfy})$, we construct an encryption scheme $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$. Gen' generates independent keys k_1 and k_2 for both the encryption scheme and the MAC. The encryption is defined by $\mathsf{Enc}'_{k_1 k_2}(m) = \mathsf{Enc}_{k_1}(m \| \mathsf{MAC}_{k_2}(m))$, and the decryption $\mathsf{Dec}'_{k_1 k_2}$ runs Dec_{k_1} and then Vrfy_{k_2} and outputs $\perp$ if the latter does. Show that this scheme is not always CCA-secure.

Hint: Let $(\widetilde{\mathsf{Gen}}_1, \widetilde{\mathsf{Enc}}, \widetilde{\mathsf{Dec}})$ be a CPA-secure encryption scheme. Now consider $(\mathsf{Gen}_1, \mathsf{Enc}, \mathsf{Dec})$ defined as $\mathsf{Gen}_1 = \widetilde{\mathsf{Gen}}_1$ and $\mathsf{Enc}_k(x) = \widetilde{\mathsf{Enc}}_k(x) \| d$ for a random bit $d \leftarrow \{0,1\}$, and $\mathsf{Dec}_k(c\|d) := \widetilde{\mathsf{Dec}}_k(c)$.

★ Problem 7: A randomized variable-length MAC from a PRF

Let F be a pseudorandom function. Show that the following MAC is insecure for variable-length messages. Gen outputs a uniform $k \in \{0,1\}^n$. Let $\langle i \rangle$ denote an $n/2$-bit encoding of the integer i.

To authenticate a message $m = m_1 \| \dots \| m_\ell$, where $m_i \in \{0,1\}^{n/2}$, choose a uniform $r \leftarrow \{0,1\}^n$, compute $t := F_k(r) \oplus F_k(\langle 1 \rangle \| m_1) \oplus \cdots \oplus F_k(\langle \ell \rangle \| m_\ell)$ and let the tag be (r, t).

★ Problem 8: Appending the message length in CBC-MAC

Show that appending the message length to the *end* of the message before applying basic CBC-MAC does not result in a secure MAC for arbitrary-length messages.

★ Problem 9: Hash functions and short inputs

Section 5.4.2 in [KL] describes a variant of the birthday attack that uses only a small amount of memory. For a hash function with output length n, it traverses the space of n bit strings by computing a sequence $x, H(x), H(H(x)), H(H(H(x)))...$ in a clever way.

1. Argue that there could be collision-resistant hash functions where this attack never succeeds.

2. Prove that any collision-resistant hash function H is literally collision-free for inputs of length $O(\log n)$, i.e. $H(x) \neq H(x')$ for $x \neq x'$ for $|x| = O(\log n)$

★ Problem 10: SHA-1

Construct an efficient algorithm that breaks SHA-1.
Hint: Why do hash functions have a key in [KL]?