University of Maryland
CMSC456 — Introduction to Cryptography
Professor Jonathan Katz

Problem Set 4

Due at *beginning* of class on Nov. 8

1. Consider the following modification of the XOR-MAC. Let $F : \{0,1\}^k \times \{0,1\}^m \rightarrow \{0,1\}^n$ be a (t, ϵ)-PRF. The sender and receiver share a random key $s \in \{0,1\}^k$ and fix a parameter ℓ. Let $\langle i \rangle$ denote the ℓ-bit representation of integer i. To authenticate message M, the sender parses M as a sequence of $(m - \ell)$-bit blocks $M_1, \ldots, M_t$ (assume that message lengths are always a multiple of $(m - \ell)$), chooses a random $r \in \{0,1\}^m$ and computes:

$$\mathsf{tag} = F_s(r) \oplus F_s(\langle 1 \rangle \circ M_1) \oplus \cdots \oplus F_s(\langle t \rangle \circ M_t).$$

The sender sends both tag and r as the authentication code for M. Note that this is different from XOR-MAC because the random block is not prefixed by 0 and the message blocks are not prefixed by 1.

   (a) How can the receiver verify correctness of a tag (tag, r) on message M?

   (b) Say an adversary (who has access to the MAC oracle, as always) knows that a sender and receiver are using the above scheme, but does not know the value of ℓ. How can the adversary determine the value of ℓ? Assume that the MAC oracle returns an error if the message length is not a multiple of $(m - \ell)$.

   (c) In the XOR-MAC scheme, an adversary asking q MAC queries was unable to forge a new tag with probability better than $2q^2 \cdot 2^{-m} + 2^{-n} + \epsilon$. Suggest how an adversary can do better for the scheme presented here. (*Hint*: I am aware of one attack in which the adversary can forge a new tag with probability $O(q^2 \cdot 2^{-\ell})$, which is better since $\ell < m$. Someone in the class suggested an even better attack. Anything better than $O(q^2/2^m)$ is ok.)

2. Assume that $(\mathcal{E}, \mathcal{D})$ is an indistinguishable private-key encryption scheme (for arbitrary-length messages) and $(\textsc{mac}, \mathsf{Vrfy})$ is a secure message-authentication scheme (for arbitrary-length messages). We want to achieve simultaneous private-key encryption and message authentication.

   One possible approach is to separately encrypt and authenticate. Here, the sender and receiver share two random, independent keys k_1, k_2 and every time the sender wants to transmit message M, he computes $C \leftarrow \mathcal{E}_{k_1}(M)$ and $\mathsf{tag} \leftarrow \textsc{mac}_{k_2}(M)$ and sends C, tag.

   (a) How would the receiver perform decryption and verification in this new scheme?

   (b) Is this scheme secure as a message authentication code? Briefly state why or why not.

(c) Is this scheme secure in the sense of left-or-right indistinguishability? Give a proof or sketch of proof if it is, or an explicit attack if it is not.

(d) **Graduate students only.** The sender and receiver want to store a shorter key so they decide to use the same key k for both encryption and authentication; i.e., the sender now computes $C \leftarrow \mathcal{E}_k(M)$ and $\texttt{tag} \leftarrow \textsc{mac}_k(M)$ and sends $C, \texttt{tag}$. Is this secure (as a message authentication code and in the sense of indistinguishability) in general? Give a proof or an explicit attack in each case depending on your answer.

3. Let p be a prime such that $p = 3 \bmod 4$. Let $x \in \mathbb{Z}_p$ be a quadratic residue.

   (a) Show that $(p+1)/4$ is an integer, and argue that therefore $x^{(p+1)/4} \bmod p$ can be efficiently computed. (Efficient here means polynomial in $|p|$).

   (b) Show that $x^{(p+1)/4}$ gives a square root of x. (Hint: use the fact that $y^{p-1} = 1 \bmod p$ for *any* $y \in \mathbb{Z}_p$ and the fact that x is a quadratic residue).

   (c) How would you find both square roots of x?

4. In this problem we will construct a collision-resistant hash function based on the hardness of computing discrete logarithms. Let $\mathbb{G}$ be a cyclic group of order q, where q is prime. Recall that such groups have the property that any element $g \in \mathbb{G}$ (with $g \neq 1$) is a generator, so that $\{g^0, g^1, \ldots, g^{q-1}\}$ is all of $\mathbb{G}$. Thus, if g is a generator then for any $h \in \mathbb{G}$ we can define the discrete logarithm of h with respect to g (denoted $\log_g h$) as the unique number $x \in \mathbb{Z}_q$ for which $g^x = h$.

   The discrete logarithm assumption states that given random generator g and random $h \in \mathbb{G}$, it is hard to compute $\log_g h$. Let g, h be generators of $\mathbb{G}$, and define hash function $H_{g,h} : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$ as follows: $H_{g,h}(x, y) = g^x h^y$.

   (a) Show that for any $h' \in \mathbb{G}$ there is at least one pair (x, y) such that $g^x h^y = h'$.

   (b) Show that for any $h' \in \mathbb{G}$ there are at least *two* distinct pairs (x_1, y_1), (x_2, y_2) such that $g^{x_1} h^{y_1} = g^{x_2} h^{y_2} = h'$.

   The remaining questions are for **graduate students only**, but undergraduates may answer them for extra credit.

   (c) Show that for any $h' \in \mathbb{G}$, there are exactly q distinct solutions (x, y) for which $g^x h^y = h'$.

   (d) Show that, given (x_1, y_1) and (x_2, y_2) such that $(x_1, y_1) \neq (x_2, y_2)$ and $g^{x_1} h^{y_1} = g^{x_2} h^{y_2}$ it is possible to *efficiently* compute $\log_g h$. Use the fact that $\mathbb{Z}_q$ is a field since q is prime.

   (e) Argue that when g and h are randomly chosen, $H_{g,h}$ is a collision-resistant hash function. (Hint: what happens if an algorithm can find a collision?)