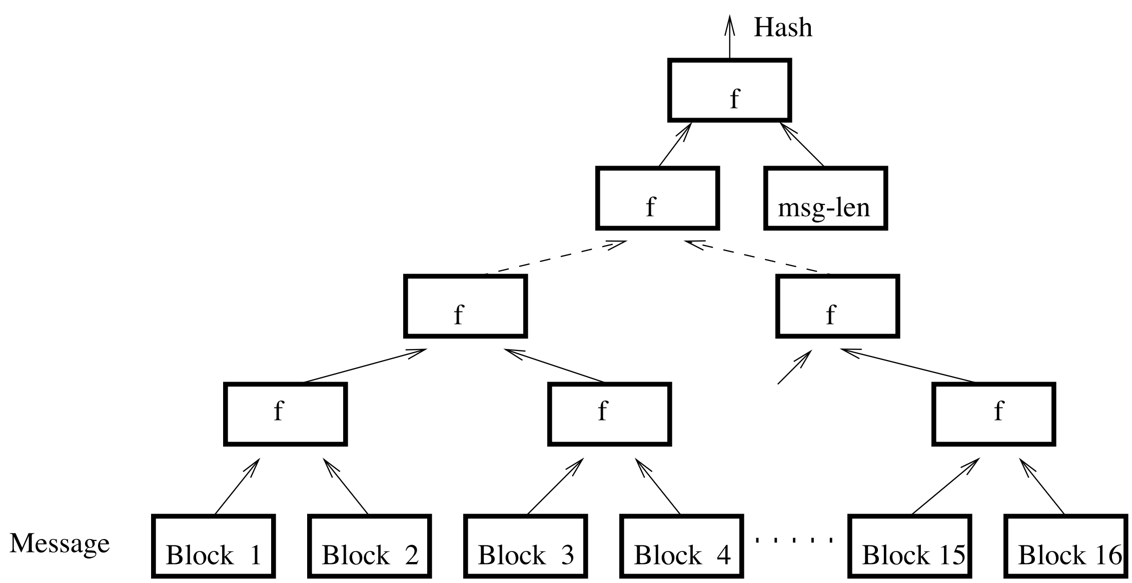
CS255: Cryptography and Computer Security **Winter 2016**

Assignment #2

Due: Monday, Feb. 22, 2016, by Gradescope (each answer on a seperate page).

Problem 1. Merkle hash trees.

Merkle suggested a parallelizable method for constructing hash functions out of compression functions. Let f be a compression function that takes two 512 bit blocks and outputs one 512 bit block. To hash a message m one uses the following tree construction:



For simplicity, let's assume that the number of blocks in m is always a power of 2.

a. Prove that if one can find a collision for the resulting hash function then one can find collisions for the compression function.

b. Show that if the msg-len block is eliminated (e.g. the contents of that block is always set to 0) then the construction is not collision resistant.

Problem 2. In the lecture we saw that Davies-Meyer is used to convert an ideal block cipher into a collision resistant compression function. Let $E(k, m)$ be a block cipher where the message space is the same as the key space (e.g. 128-bit AES). Show that the following methods do not work:

$$f_1(x, y) = E(y, x) \oplus y \quad \text{and} \quad f_2(x, y) = E(x,\ x \oplus y)$$

That is, show an efficient algorithm for constructing collisions for f_1 and f_2. Recall that the block cipher E and the corresponding decryption algorithm D are both known to you.

Problem 3. Suppose user A is broadcasting packets to n recipients $B_1, \ldots, B_n$. Privacy is not important but integrity is. In other words, each of $B_1, \ldots, B_n$ should be assured that the packets he is receiving were sent by A. User A decides to use a MAC.

a. Suppose user A and $B_1, \ldots, B_n$ all share a secret key k. User A computes the MAC for every packet she sends using k. Every user B_i can verify the MAC using k. Using at most two sentences explain why this scheme is insecure, namely, show that user B_1 is not assured that packets he is receiving are from A.

b. Suppose user A has a set $S = \{k_1, \ldots, k_m\}$ of m secret keys. Each user B_i has some subset $S_i \subseteq S$ of the keys. When A transmits a packet she appends m MACs to it by MACing the packet with each of her m keys. When user B_i receives a packet he accepts it as valid only if all MAC's corresponding to keys in S_i are valid. What property should the sets $S_1, \ldots, S_n$ satisfy so that the attack from part (a) does not apply? We are assuming the users $B_1, \ldots, B_n$ do not collude with each other.

c. Show that when $n = 10$ (i.e. ten recipients) the broadcaster A need only append 5 MAC tags to every packet to satisfy the condition of part (b). Describe the sets $S_1, \ldots, S_{10} \subseteq \{k_1, \ldots, k_5\}$ you would use.

Problem 4. Timing atacks. Let (S, V) be a deterministic MAC system where tags $\mathcal{T}$ are n-bytes long. The verification algorithm $V(k, m, t)$ is implemented as follows: it first computes $t' \leftarrow S(k, m)$ and then does:

for $i \leftarrow 0$ to $n - 1$ do:
  if $t[i] \neq t'[i]$ output reject and exit
output accept

a. Show that this implementation is vulnerable to a timing attack. An attacker who can submit arbitrary queries to algorithm V and accurately measure V's response time can forge a valid tag on every message m of its choice with at most $256 \cdot n$ queries to V.

b. How would you implement V to prevent the timing attack from part (a)?

Problem 5. Authenticated encryption. Let (E, D) be an encryption system that provides authenticated encryption. Here E does not take a nonce as input and therefore must be a randomized encryption algorithm. Which of the following systems provide authenticated encryption? For those that do, give a short proof. For those that do not, present an attack that either breaks CPA security or ciphertext integrity.

a. $E_1(k,m) = \big[c \leftarrow E(k,m),\ \text{output}\ \ (c,c)\ \big]$ and $D_1(k,\ (c_1,c_2)\ ) = D(k,c_1)$

b. $E_2(k,m) = \big[c \leftarrow E(k,m),\ \text{output}\ \ (c,c)\ \big]$ and $D_2(k,\ (c_1,c_2)\ ) = \begin{cases} D(k,c_1) & \text{if } c_1 = c_2 \\ \mathsf{fail} & \text{otherwise} \end{cases}$

c. $E_3(k,m) = \big(\ E(k,m),\ E(k,m)\ \big)$ and $D_3(k,\ (c_1,c_2)\ ) = \begin{cases} D(k,c_1) & \text{if } D(k,c_1) = D(k,c_2) \\ \mathsf{fail} & \text{otherwise} \end{cases}$

To clarify: $E(k,m)$ is randomized so that running it twice on the same input will result in different outputs with high probability.

d. $E_4(k,m) = \big(\ E(k,m),\ H(m)\ \big)$ and $D_4(k,\ (c_1,c_2)\ ) = \begin{cases} D(k,c_1) & \text{if } H(D(k,c_1)) = c_2 \\ \mathsf{fail} & \text{otherwise} \end{cases}$

where H is a collision resistant hash function.

Problem 6. Let p be a prime with $p \equiv 2 \bmod 3$.

a. Show an efficient algorithm that takes $\alpha \in \mathbb{Z}_p^*$ as input and outputs the cube root of α in $\mathbb{Z}_p^*$. That is, show how to efficiently solve the equation $x^3 - \alpha = 0$ in $\mathbb{Z}_p$.
Hint: recall how RSA decryption works.

b. Is your algorithm from part (a) able to compute cube roots modulo a composite $N = pq$ when the factorization of N is unknown? If so explain why, if not explain why not.

Problem 7. Let G be a finite cyclic group. Suppose the order of G is $2q$ for some odd integer q. Show that the Decision Diffie-Hellman problem does not hold in the group G.
Hint: given a tuple (g, h, u, v) try raising g, h, u, v to the power of q.