Homework 3[1]

Question 1

Suppose a MAC system (S,V) is used to protect files in a file system by appending a MAC tag to each file. The MAC signing algorithm S is applied to the file contents and nothing else. What tampering attacks are not prevented by this system?

1. Swapping two files in the file system.

2. Replacing the tag and contents of one file with the tag and contents of a file from another computer protected by the same MAC system, but a different key.

3. Replacing the contents of a file with the concatenation of two files on the file system.

4. Erasing the last byte of the file contents.

Question 2

Let (S,V) be a secure MAC defined over (K,M,T) where $M=\{0,1\}^n$ and $T=\{0,1\}^{128}$ (i.e. the key space is K, message space is $\{0,1\}^n$, and tag space is $\{0,1\}^{128}$). Which of the following is a secure MAC: (as usual, we use $\|$ to denote string concatenation).

1. $S'(k,m)=S(k,\ m[0,\ldots,n-2]\|0)$ and $V'(k,m,t)=V(k,\ m[0,\ldots,n-2]\|0,\ t)$

2. $S'(k,m)=S(k,m)[0,\ldots,126]$ and $V'(k,m,t)=\big[V(k,\ m,\ t\|0)\ \text{or}\ V(k,\ m,\ t\|1)\ \big]$

(i.e., $V'(k,m,t)$ outputs "1" if either $t\|0$ or $t\|1$ is a valid tag for m).

3. $S'(k,m)=S(k,\ m\|m)$ and $V'(k,m,t)=V(k,\ m\|m,\ t)$.

4.

$$S'(k,m)=S(k,m) \quad \text{and}$$
$$V'(k,m,t)=\begin{cases} V(k,m,t) & \text{if } m\neq 0^n \\ \text{“1”} & \text{otherwise}\end{cases}$$

5.

$$S'(k,m)=S(k,m) \quad \text{and}$$
$$V'(k,m,t)=\big[V(k,\ m,\ t)\ \text{or}\ V(k,\ m\oplus 1^n,\ t)\big]$$

(i.e., $V'(k,m,t)$ outputs "1" if t is a valid tag for either m or $m\oplus 1^n$).

6.

$$S'((k_1,k_2),\ m)=\big(S(k_1,m),S(k_2,m)\big) \quad \text{and}$$
$$V'\big((k_1,k_2),m,(t_1,t_2)\big)=\big[V(k_1,m,t_1)\ \text{and}\ V(k_2,m,t_2)\big]$$

(i.e., $V'\big((k_1,k_2),m,(t_1,t_2)\big)$ outputs "1" if both t_1 and t_2 are valid tags).

[1] https://class.coursera.org/crypto-012/

Question 3

Recall that the ECBC-MAC uses a fixed IV (in the lecture we simply set the IV to 0). Suppose instead we chose a random IV for every message being signed and include the IV in the tag. In other words, $S(k,m) := \big(r,\ \text{ECBC}_r(k,m)\big)$ where $\text{ECBC}_r(k,m)$ refers to the ECBC function using r as the IV. The verification algorithm V given key k, message m, and tag (r,t) outputs "1" if $t = \text{ECBC}_r(k,m)$ and outputs "0" otherwise.

The resulting MAC system is insecure. An attacker can query for the tag of the 1-block message m and obtain the tag (r,t). He can then generate the following existential forgery: (we assume that the underlying block cipher operates on n-bit blocks)

1. The tag $(r,\ t \oplus r)$ is a valid tag for the 1-block message 0^n.

2. The tag $(m \oplus t,\ t)$ is a valid tag for the 1-block message 0^n.

3. The tag $(r \oplus 1^n,\ t)$ is a valid tag for the 1-block message $m \oplus 1^n$.

4. The tag $(m \oplus t,\ r)$ is a valid tag for the 1-block message 0^n.

Question 4

Suppose Alice is broadcasting packets to 6 recipients $B_1, \ldots, B_6$ should be assured that the packets he is receiving were sent by Alice.

Alice decides to use a MAC. Suppose Alice and $B_1, \ldots, B_6$ all share a secret key k. Alice computes a tag for every packet she sends using key k. Each user B_i verifies the tag when receiving the packet and drops the packet if the tag is invalid. Alice notices that this scheme is insecure because user B_1 can use the key k to send packets with a valid tag to users $B_1, \ldots, B_6$ and they will all be fooled into thinking that these packets are from Alice.

Instead, Alice sets up a set of 4 secret keys $S = \{k_1, \ldots, k_4\}$. She gives each user B_i some subset $S_i \subseteq S$ of the keys. When Alice transmits a packet she appends 4 tags to it by computing the tag with each of her 4 keys. When user B_i receives a packet he accepts it as valid only if all tags corresponding to his keys in S_i are valid. For example, if user B_1 is given keys $\{k_1, k_2\}$ he will accept an incoming packet only if the first and second tags are valid. Note that B_1 cannot validate the 3rd and 4th tags because he does not have k_3 or k_4.

How should Alice assign keys to the 6 users so that no single user can forge packets on behalf of Alice and fool some other user?

1. $S_1 = \{k_1,k_2\},\ S_2 = \{k_1\},\ S_3 = \{k_1,k_4\},\ S_4 = \{k_2,k_3\},\ S_5 = \{k_2,k_4\},\ S_6 = \{k_3,k_4\}$

2. $S_1 = \{k_1,k_2\},\ S_2 = \{k_1,k_3\},\ S_3 = \{k_1,k_4\},\ S_4 = \{k_2,k_3,k_4\},\ S_5 = \{k_2,k_3\},\ S_6 = \{k_3,k_4\}$

3. $S_1 = \{k_1,k_2\},\ S_2 = \{k_1,k_3\},\ S_3 = \{k_1,k_4\},\ S_4 = \{k_2,k_3\},\ S_5 = \{k_2,k_4\},\ S_6 = \{k_3,k_4\}$

4. $S_1 = \{k_1,k_2\},\ S_2 = \{k_1,k_3\},\ S_3 = \{k_1,k_4\},\ S_4 = \{k_2,k_3\},\ S_5 = \{k_2,k_4\},\ S_6 = \{k_4\}$

Question 5

Consider the encrypted CBC MAC built from AES. Suppose we compute the tag for a long message m comprising of n AES blocks. Let m' be the n-block message obtained from m by flipping the last bit of m (i.e. if the last bit of m is b then the last bit of m' is $b \oplus 1$). How many calls to AES would it take to compute the tag for m' from the tag for m and the MAC key? (in this question please ignore message padding and simply assume that the message length is always a multiple of the AES block size)

1. 4

2. 5

3. 2

4. n

Question 6

Let $H : M \to T$ be a collision resistant hash function. Which of the following is collision resistant: (as usual, we use $\|$ to denote string concatenation).

1. $H'(m) = H(m\|m)$

2. $H'(m) = H(m) \oplus H(m)$

3. $H'(m) = H(|m|)$ (i.e. hash the length of m)

4. $H'(m) = H(m) \bigoplus H(m \oplus 1^{|m|})$ (where $m \oplus 1^{|m|}$ is the complement of m)

5. $H'(m) = H(m)\|H(m)$

6. $H'(m) = H(0)$

7. $H'(m) = H(m)\|H(0)$

Question 7

Suppose H_1 and H_2 are collision resistant hash functions mapping inputs in a set M to $\{0,1\}^{256}$. Our goal is to show that the function $H_2(H_1(m))$ is also collision resistant. We prove the contra-positive: suppose $H_2(H_1(\cdot))$ is not collision resistant, that is, we are given $x \neq y$ such that $H_2(H_1(x)) = H_2(H_1(y))$. We build a collision for either H_1 or for H_2. This will prove that if H_1 and H_2 are collision resistant then so is $H_2(H_1(\cdot))$. Which of the following must be true:

1. Either $x, H_1(y)$ are a collision for H_2 or $H_2(x), y$ are a collision for H_1.

2. Either x, y are a collision for H_2 or $H_1(x), H_1(y)$ are a collision for H_1.

3. Either x, y are a collision for H_1 or $H_1(x), H_1(y)$ are a collision for H_2.

4. Either $H_2(x), H_2(y)$ are a collision for H_1 or x, y are a collision for H_2.

Question 8

In this question and the next, you are asked to find collisions on two compression functions:

$$f_1(x,y) = \text{AES}(y,x) \bigoplus y \qquad \text{and}$$
$$f_2(x,y) = \text{AES}(x,x) \bigoplus y$$

where $\text{AES}(x,y)$ is the AES-128 encryption of y under key x.

Your goal is to find four distinct pairs $(x_1,y_1),(x_2,y_2),(x_3,y_3),(x_4,y_4)$ such that $f_1(x_1,y_1) = f_1(x_2,y_2)$ and $f_2(x_3,y_3) = f_2(x_4,y_4)$. In other words, the first two pairs are a collision for f_1 and the last two pairs are a collision for f_2.

Question 9

Let $H : M \to T$ be a random hash function where $|M| \gg |T|$ (i.e. the size of M is much larger than the size of T). In lecture we showed that finding a collision on H can be done with $O\big(|T|^{1/2}\big)$ random samples of H. How many random samples would it take until we obtain a three way collision, namely distinct strings x,y,z in M such that $H(x) = H(y) = H(z)$?

1. $O\big(|T|^{1/3}\big)$

2. $O\big(|T|^{1/2}\big)$

3. $O\big(|T|\big)$

4. $O\big(|T|^{2/3}\big)$