Week 3 - Problem Set

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?

- [✓ Changing the name of a file.]
- Replacing the contents of a file with the concatenation of two files on the file system.
- Erasing the last byte of the file contents.
- Appending data to a file.

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)

- $S'(k,m)=S(k,m)[0,\ldots,126]$ and $V'(k,m,t)=\left[V(k,\ m,\ t\|0)\ \text{ or }\ V(k,\ m,\ t\|1)\ \right]$
  (i.e., $V'(k,m,t)$ outputs ``1" if either $t\|0$ or $t\|1$ is a valid tag for m)
- $S'(k,m)=S(k,\ \ m[0,\ldots,n-2]\|0)$ and $V'(k,m,t)=V(k,\ \ m[0,\ldots,n-2]\|0,\ \ t)$
- $S'(k,m)=S(k,m)$ and $V'(k,m,t)=\begin{cases}V(k,m,t) & \text{if } m\neq 0^n\\ ``1" & \text{otherwise}\end{cases}$
- [✓ $S'(k,m)=S(k,m\oplus m)$ and $V'(k,m,t)=V(k,\ \ m\oplus m,\ t)$]
- [✓ $S'(k,m)=S(k,\ \ m\|m)$ and $V'(k,m,t)=V(k,\ \ m\|m,\ t)$.]
- $S'(k,\ m)=\left[t\leftarrow S(k,m),\ \text{output }(t,t)\ \right)$ and $V'(k,m,(t_1,t_2))=\begin{cases}V(k,m,t_1) & \text{if } t_1=t_2\\ "0" & \text{otherwise}\end{cases}$
  (i.e., $V'(k,m,(t_1,t_2))$ only outputs "1" if t_1 and t_2 are equal and valid)

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, $S(k,m):=\big(r,\ \ \mathrm{ECBC}_r(k,m)\big)$ where $\mathrm{ECBC}_r(k,m)$ 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 $t=\mathrm{ECBC}_r(k,m)$ 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)

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

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

Question 4

Suppose Alice is broadcasting packets to 6 recipients $B_1, \ldots, B_6$. Privacy is not important but integrity is. In other words, each of $B_1, \ldots, B_6$ 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 $B_2, \ldots, B_6$ 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?

- $S_1 = \{k_1, k_2\},\ \ S_2 = \{k_1\},\ \ S_3 = \{k_1, k_4\},\ \ S_4 = \{k_2, k_3\},\ \ S_5 = \{k_2, k_4\},\ \ S_6 = \{k_3, k_4\}$
- $S_1 = \{k_1, k_2\},\ \ S_2 = \{k_2, k_3\},\ \ S_3 = \{k_3, k_4\},\ \ S_4 = \{k_1, k_3\},\ \ S_5 = \{k_1, k_2\},\ \ S_6 = \{k_1, k_4\}$
- $S_1 = \{k_1, k_2\},\ \ S_2 = \{k_1, k_3, k_4\},\ \ S_3 = \{k_1, k_4\},\ \ S_4 = \{k_2, k_3\},\ \ S_5 = \{k_2, k_3, k_4\},\ \ S_6 = \{k_3, k_4\}$
- [✓ $S_1 = \{k_2, k_3\},\ \ S_2 = \{k_2, k_4\},\ \ S_3 = \{k_3, k_4\},\ \ S_4 = \{k_1, k_2\},\ \ S_5 = \{k_1, k_3\},\ \ S_6 = \{k_1, k_4\}$]

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)

- [✓ 5]
- $n+1$
- 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)

- [✓ $H'(m) = H(m \| m)$]
- $H'(m) = H(0)$

- [✓] $H'(m) = H(H(m))$
- [ ] $H'(m) = H(m[0, \ldots, |m| - 2])$ (i.e. hash m without its last bit)
- [ ] $H'(m) = H(m)[0, \ldots, 31]$ (i.e. output the first 32 bits of the hash)
- [✓] $H'(m) = H(H(H(m)))$
- [✓] $H'(m) = H(m) \oplus H(m)$

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:

- ○ Either $H_2(x), H_2(y)$ are a collision for H_1 or x, y are a collision for H_2.
- [✓ ●] Either x, y are a collision for H_1 or $H_1(x), H_1(y)$ are a collision for H_2.
- ○ Either $x, H_1(y)$ are a collision for H_2 or $H_2(x), y$ are a collision for H_1.
- ○ Either x, 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:

- $f_1(x,y) = \text{AES}(y,x) \bigoplus y,$ 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.

We provide an AES function for you to play with. The function takes as input a key k and an x value and outputs $\text{AES}(k,x)$ once you press the "encrypt" button. It takes as input a key k and a y value and outputs $\text{AES}^{-1}(k,y)$ once you press the "decrypt" button. All three values k, x, y are assumed to be hex values (i.e. using only characters 0-9 and a-f) and the function zero-pads them as needed.

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. Once you find all four pairs, please enter them below and check your answer using the "check" button.

> Note for those using the NoScript browser extension: for the buttons to function correctly please allow Javascript from class.coursera.org and cloudfront.net to run in your browser. Note also that the "save answers" button does not function for this question and the next.

AES Function

Enter data in hex:

Key (Hex):	00000000000000000000000000000000
Plaintext (Hex):	37ad6f86bd0c4d781c60cad081960f09
Ciphertext (Hex):	7b1e2fffe8a114009d8fe22f6db5f876

Encrypt Decrypt

Please enter $(x_1,y_1),\ (x_2,y_2)$ such that $f_1(x_1,y_1) = f_1(x_2,y_2)$. Find these pairs using the AES function above.

x_1:

	00000000000000000000000000000000
y_1:	10000000000000000000000000000000
x_2:	37ad6f86bd0c4d781c60cad081960f09
y_2:	00000000000000000000000000000000

Check f1 Correct!

Question 9

Please enter $(x_3, y_3),\ (x_4, y_4)$ such that $f_2(x_3, y_3) = f_2(x_4, y_4)$. You can use the AES function from the previous question.

x_3:	11100000000000000000000000000000
y_3:	00000000000000000000000000000000
x_4:	00000000000000000000000000000000
y_4:	43a12976032b84f1ea0c1d35e8ba7a26

Check f2 Correct!

Question 10

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 $O\left(|T|^{1/2}\right)$ 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 $H(x) = H(y) = H(z)$?

- $O\left(|T|^{1/3}\right)$
- [✓ $O\left(|T|^{2/3}\right)$]
- $O\left(|T|^{1/4}\right)$
- $O\left(|T|^{3/4}\right)$

Submit Answers Save Answers