CPSC 418 / MATH 318 — Introduction to Cryptography
ASSIGNMENT 3 Problems 1-5

Name: Theodore Yamit
Student ID: 30141383 (replace by your ID number)

Problem 1 — Pre-image resistance versus collision resistance (14 marks)

(a) i. From the question, we know that h is pre-image resistant, such that given a hash x and $H(M) = x$, it is computationally infeasible to find M. We now want to prove that the new hash function H is also pre-image resistant.

Proof by Contradiction:
Let's assume that H is NOT pre-image resistant.
This means that we can find a pre-image of x, such that we can find M for $H(M) = x$.

Since we are able to find a pre-image of x under H, and $H(M) = h(M')$, then this means we are also able to find a pre-image under h. However, since h is a pre-image resistant has function, then this leads to a contradiction (since if H is no pre-image resistant, then h must also not be pre-image resistant).

Thus, our initial assumption of H not being pre-image resistant must be wrong, making H pre-image resistant.

ii. To prove that H is not collision resistant, we can provide a counter example.
Note: We will be proving this for strong collisions (since this will prove H is not weak-collision resistant by definition as well).

Since we a proving for strong collision resistance, it is sufficient to show that it is computationally infeasible to find two distinct M_1 and M_2 such that $H(M_1) = H(M_2)$.

Counterexample:
Let M_1 equal any bit string of length n, but have the last bit be a 0.
Let M_2 be another unique bit string of length n, where the first $n-1$ bits are the same $n-1$ bits of M_1, except for the last bit, which is a 1. From the description of this question, M' agrees with M, except for the last bit, which is changed from 1 to a 0 (in the case it is 1).

Thus, we have:

$M_1' = M_1$
$M_2' = M_2$ (Where the last bit of M_2 is changed from a 1 to a 0)

We can see that $M_1' = M_2'$, since the first $n-1$ bits of both messages are the same, and M_2's last bit was changed from a 1 to a 0, matching M_1.

Since $M_1' = M_2'$, we have $h(M_1) = h(M_2)$, and thus, $H(M_1) = H(M_2)$.

Since we have found a collision, and $M_1 \neq M_2$, we have proven that H is not a strongly collision resistant hash function (which also makes it not a weakly collision resistant hash function either).

(b) i. We want to prove that H is collision resistant (strongly collision resistant, since this proves it is also weakly collision resistant). H being strongly collision resistant means that it is computationally infeasiable to find two distinct messages M_1 and M_2, such that $H(M_1) = H(M_2)$.

Proof by Contradiction: Let's assume that H is NOT collision resistant.
This means that it is computationally feasible to find two messages M_1 and M_2, where $M_1 \neq M_2$, such that $H(M_1) = H(M_2)$.

Suppose we have two messages M_1 and M_2.

There are two cases to consider such that they land on the same case (since one case prepends 0, and the other cases prepends 1):

1. When M_1 and M_2 are both 0 (both are just the 0 bit).
2. When M_1 and M_2 are both distinct and not just the 0 bit.

1st case (Both M_1 and M_2 are both the 0 bit):
While this results in the same hash of $1||0^n$, which showcases a collision ($H(M_1) = H(M_2)$), $M_1 = M_2$, meaning they aren't distinct messages. This leads to a contradiction.

2nd case (Both M_1 and M_2 are both distinct and not just the 0 bit):
Then $H(M_1) = 0||h(M_1)$ and $H(M_2) = 0||h(M_2)$. Since $M_1 \neq M_2$, we want it such that there is a collision for $h(M_1)$ and $h(M_2)$, such that $h(M_1) = h(M_2)$. However, since h is a collision resistant hash function, this can't happen, and thus we have another contradiction.

Since both cases lead to a contradiction, our initial assumption of H not being collision resistant must be wrong, and thus, H is a collision resistant hash function.

ii. Pre-image resistance means that given a hash x, such that $H(M) = x$, it is computationally infeasible to find M. So we can make a counterexample with a specific x.

Thus, given a hash x, such that $H(M) = x$, suppose that x's string starts with 1. Then we instantly obtain M. This is because for x to start with 1, M must be the 0 bit (since 1 is prepended in this case, while 0 is prepended if M is not the 0 bit). Since it is not computationally infeasible in this case, where the hash x starts with 1, then it is possible to find M such that $H(M) = x$ (namely, $M = 0$), which proves that H is not pre-image resistant.

Problem 2 — CBC-MAC and one-register CFB-MAC (6 marks)

(a) First, let's state definitions from both CBC-MAC and CFB-MAC.
For CBC-MAC:

$$C_0 = 0^n \text{ (Initial State)} \tag{1}$$
$$C_i = E_K(M_i \oplus C_{i-1}) \text{ (For } 1 \leq i \leq L) \tag{2}$$
$$\text{CBC-MAC}(M) = C_L \text{ (Final State)} \tag{3}$$

For CFB-MAC:

$$C'_0 = M_1 \text{ (Initial State)} \tag{4}$$
$$C'_i = M_{i+1} \oplus E_K(C'_{i-1}) \text{ (For } 1 \leq i \leq L-1) \tag{5}$$
$$\text{CFB-MAC}(M) = E_K(C'_{L-1}) \text{ (Final State)} \tag{6}$$

Let's use induction on i to prove that $C_i = C'_i \oplus M_{i+1}$ for $0 \leq i \leq L-1$.

Base Case - i = 0:

$$\begin{aligned}
C_0 &= C'_0 \oplus M_{0+1} && \textbf{By our assumption} \\
C_0 &= M_1 \oplus M_1 && \textbf{By definition of CFB-MAC (5)} \\
C_0 &= 0^n && \textbf{XOR laws: } \mathbf{A \oplus A = 0^n : (n = |A|)}
\end{aligned}$$

Since $C_0 = 0^n$, by definition in (1), $C_i = C'_i \oplus M_{i+1}$ is true for the base case.

Inductive Hypothesis: Suppose that $C_k = C'_k \oplus M_{k+1}$ holds true for some k, where $0 \leq k \leq L-1$.
Inductive Step: Thus, it is sufficient to show that $C_{k+1} = C'_{k+1} \oplus M_{k+2}$ holds true for $k+1$, where $0 \leq k \leq k+1 \leq L-1$.

Note: k is different from K (K is the key and not really apart of the proof)

Proof by induction:

$$\begin{aligned}
C_{k+1} &= E_K(M_{k+1} \oplus C_k) && \textbf{By definition for CBC-MAC (2)} \\
C_{k+1} &= E_K(M_{k+1} \oplus C'_k \oplus M_{k+1}) && \textbf{By our inductive hypothesis} \\
C_{k+1} &= E_K(M_{k+1} \oplus M_{k+1} \oplus C'_k) && \textbf{Property of XOR: Commutative} \\
C_{k+1} &= E_K(C'_k) && \textbf{XOR rule: } \mathbf{A \oplus A = 0}
\end{aligned}$$

Now,

$$\begin{aligned}
C'_{k+1} &= M_{k+2} \oplus E_K(C'_k) && \textbf{By definition for CFB-MAC (5)} \\
C'_{k+1} &= M_{k+2} \oplus C_{k+1} && \textbf{From previous step } \mathbf{C_{k+1} = E_K(C'_k)} \\
C_{k+1} &= C'_{k+1} \oplus M_{k+2} && \textbf{XOR property: } \mathbf{A = B \oplus C \rightarrow C = A \oplus B}
\end{aligned}$$

We have shown that if this statement holds for k, then this statement also holds for $k+1$, proving our inductive hypothesis.

Thus, using induction on i, we have proven that $C_i = C'_i \oplus M_{i+1}$ for $0 \leq i \leq L-1$. ■

(b) Let M be any message, and K be a given key.
We want to show that CBC-MAC(M) = CFB-MAC(M).

Direct proof:
For $i = L - 1$ (the second last block), we have:

$$C_{L-1} = C'_{L-1} \oplus M_L \qquad \textbf{By our inductive proof in part (a)}$$

For $i = L$ (The last block), we have:

$$\begin{aligned}
\text{CBC-MAC}(M) &= C_L && \textbf{By definition of CBC-MAC (3)} \\
&= E_K(M_L \oplus C_{L-1}) && \textbf{By definition of CBC-MAC (2)} \\
&= E_K(M_L \oplus C'_{L-1} \oplus M_L) && \textbf{From our previous step on } \mathbf{C_{L-1}} \\
&= E_K(M_L \oplus M_L \oplus C'_{L-1}) && \textbf{Property of XOR: Commutative} \\
&= E_K(C'_{L-1}) && \textbf{XOR rule: } \mathbf{A \oplus A = 0} \\
\text{CBC-MAC}(M) &= \text{CFB-MAC}(M) && \textbf{By definition of CFB-MAC (6)}
\end{aligned}$$

Thus, we have proven that CBC-MAC(M) = CFB-MAC(M). ■

Problem 3 — Flawed MAC designs (14 marks)

(a) Let's list what we know so far:
- The attacker has a message/PHMAC pair (M_1, PHMAC$_K(M_1)$)
- X is an n-bit block and $M_2 = M_1||X$ - We want to compute PHMAC$_K(M_2)$ without the key K.

Notice that $M_2 = M_1||X$, or more specifically, that M_2 starts with M_1.

From the ITHASH function, we know that this is an iterated hash function that works on a message M, consisting of L blocks.

We are given a message/PHMAC pair (M_1, PHMAC$_K(M_1)$). We can see from the ITHASH function, when the input is M_1, such that PHMAC$_K(M_1)$, the returned PHMAC$_K(M_1)$ is the final state when $i = L$.

We are given PHMAC$_K(M_1)$ already, which would have already have had the key K appended at the start of the ITHASH algorithm. Since we don't have the key, we can just start the algorithm ITHASH again but starting from the final state of PHMAC$_K(M_1)$, and then just continue processing from the block X (Which makes sense because $M_2 = M_1||X$).

(b)

Problem 4 — El Gamal is not semantically secure (12 marks)

Problem 5 — An IND-CPA, but not IND-CCA secure version of RSA (12 marks)