CS 198 Spring 2023	Codebreaking at Cal Makeup Assignment	Week 4

Question 1

Suppose we have a hash function H that takes in a bitstring M. We define $H(M) = M_1 \oplus M_2$, where we can split M in half as $M = M_1 || M_2$.

1. Is H preimage resistant?

   Solution: No. Suppose we know $H(M)$. We can choose $M = 0||H(M)$, and this will be a valid preimage of $H(M)$.

2. Is H weak collision resistant?

   Solution: No. Suppose we know $M = M_1||M_2$ and $H(M)$. We can choose $M' = 0||(M_1 \oplus M_2)$ (among many other possibilities) so that $H(M) = H(M')$.

3. Is H strong collision resistant?

   Solution: No. A hash function that isn't weak collision resistant cannot be strong collision resistant.

Question 2

Instead of working with bitstrings, we decide to work with the set of English uppercase letters. Define $\alpha = \{A, B, \ldots, Z\}$. Suppose we have a cryptographic hash function H that takes in variable-length messages and outputs a string of letters of length n (in math notation, $H : \alpha^* \to \alpha^n$).

Note: It's OK if your answer to either of the following 2 subparts is off by a constant factor (e.g. $\frac{1}{2}(2^n)$ *instead of* 2^n*).*

1. Suppose we know the hash $H(M)$ for an unknown message M. In terms of n, how many guesses do we need before the probability we've found M is over 50%?

   Solution: Since the hash function is cryptographic, we need to brute force possibilities here. There are 26^n possible outputs of H, so after looking at $\frac{1}{2}(26^n)$ messages, there is a 50% chance we will have found M.

2. In terms of n, how many messages M would we need to examine before the probability that we've found a collision (between any of the two messages we've looked at) is 50%?

   Solution: Because of the birthday paradox, we only need about $26^{n/2}$ guesses before we will have found a collision. (See the lecture slides or recording for a proof why this is the case.)

Question 3

Suppose $Enc(K,M)$ is an IND-CPA secure encryption function that takes a key K and message M, and H is a cryptographic hash function. Alice and Bob share two symmetric keys K_1 and K_2 that Mallory doesn't know. Alice sends Bob $Enc(K_1,M)$ and $H(H(K_2||M))$.

1. Does this scheme provide integrity? Why or why not?

   Solution: Yes. If Mallory tampers with any part of the message, the result will be detected when Bob decrypts M and computes $H(H(K_2||M))$. Further, this is not vulnerable to a length extension attack because we apply H a second time.

2. Why is this scheme *not* IND-CPA secure?

   Solution: Note that the second part of the message (the MAC) is completely deterministic. Therefore, even though *Enc* is IND-CPA, the scheme as a whole is not.

3. Modify this scheme to make it IND-CPA secure.

   Solution: We can replace the MAC with $H(H(K_2||Enc(K_1,M)))$. This is no longer deterministic since *Enc* is not deterministic, so it won't leak anything about the contents of M.

Contributors:

- Ryan Cottone, Will Giorza