Exercises on hash functions and MACs

Exercise 6.5: Give an example of a hash function that is presumably preimage resistant, but not second preimage resistant.

Hint for solution: Let $x = x_0||x_1$, and let the hash function $h(x) = h_1(x_1)$, where h_1 is presumably preimage resistant. Then h is presumably preimage resistant, but not second preimage resistant since you can just change x_0 without changing the hash value, if you know x.

Exercise 6.6: Define a hash function h to hash n-bit strings to m-bit strings (it is then a compression function). Assume h is constructed as $h : \mathbb{Z}_{2^n} \to \mathbb{Z}_{2^m}$ through

$$h(x) = ((\sum_{i=0}^{d} a_i x^i) \bmod 2^n) \bmod 2^m,$$

for some fixed d and coefficients $a_i, i = 0..d$. Show how you solve the second preimage problem without solving some polynomial equation.

Hint for solution: Add a multiple of 2^m to the given message.

Exercise 6.7: Assume we have a hash function h_1 (compression function) mapping $\{0,1\}^{2m}$ to $\{0,1\}^m$, i.e., compressing $2m$ bits to m bits. Assume furthermore that h_1 is collision resistant. Now construct a new hash function h_2 mapping $\{0,1\}^{4m}$ to $\{0,1\}^m$, i.e., compressing $4m$ bits to m bits, by

$$h_2(x) = h_2(x_1||x_2) = h_1(h_1(x_1)||h_1(x_2)),$$

where $x \in \{0,1\}^{4m}$ is written as $x = x_1||x_2, x_1, x_2 \in \{0,1\}^{2m}$ and $||$ means bitstring concatenation.

Prove that h_2 is collision resistant.

Hint for solution: Assume the opposite. Let $y_1 = h_1(x_1)$ and $y_2 = h_1(x_2)$. Then $h_2(x_1||x_2) = h_1(h_1(x_1)||h_1(x_2)) = h_1(y_1||y_2)$. Assume the collision is with $x_1'||x_2'$. By definition $x_1||x_2$ is different from $x_1'||x_2'$, so assume for example $x_1 \neq x_1'$. They give intermediate values $y_1 = h_1(x_1)$ and $y_1' = h_1(x_1')$. If you have $y_1 = y_1'$ then you have a collision in h_1, a contradiction. So $y_1 \neq y_1'$. But then $y_1||y_2$ is different from $y_1'||y_2'$ and since we assumed a collision, this also leads to a contradiction.

Exercise 6.8: [difficult] Consider the generation of b-bit MACs through CBC-MAC (slide 5, lec15). The MAC is denoted as $MAC(x,k)$, where $x = x_1 x_2 \ldots x_n$, and k is the secret key, and generated by computing $y_i = E_k(y_{i-1} \oplus x_i)$ and finally setting $MAC(x,k) = y_n$. The block size is b bits.

Assume that Eve can get Alice to generate the MACs for about $q = 2 \cdot 2^{b/2}$ different messages of her choice. Show how she can then find a correct MAC (that is correctly verified by Bob) for a new message for which Alice never generated a MAC.

Hint: For the q messages, let x_1 run through q different values, let x_2 be randomly chosen and let $x_3, \ldots$ be the same for all messages. Then use birthday paradox arguments.

Hint for solution: Consider the q different messages $x^{(i)}$ that Alice is going to provide MACs for. Let $x^{(i)} = x_1^{(i)} x_2^{(i)} x_3^{(i)} \ldots$. Now select $x_1^{(i)} = [i]$, where $[i]$ means the binary representation of i (so they are all different). Then $x_2^{(i)}$ is a random choice for every i. For the remaining blocks, $x_3^{(i)}$ is the same fixed value for all i. Now the birthday argument says that it is very likely that among all these q messages, there are at least two that have the same MAC value. Call these two messages x and x' and their MACs is $y_n = y_n'$.

Since they share the same blocks from the third block and onwards, this leads to $y_2 = y_2'$. So for these two MACs we get the relationship $y_1 \oplus x_2 = y_1' \oplus x_2'$. The main observation is now that a change on the x_2 part does not change the y_1 part. So we can construct two new messages that are known to have the same MAC, namely $x_{new} = x_1(x_2 + \delta)x_3 \ldots$ and $x_{new}' = x_1'(x_2' + \delta)x_3 \ldots$.

So Eve is asking for one more message to be authenticated by Alice, x_{new}, and gets the MAC. She then has another message x_{new}' that was never sent by Alice but Eve can generate its MAC with probability 1.