ECE 3170 Cryptographic Hardware for Embedded Systems

Fall 2025

Assoc. Prof. Vincent John Mooney III

Georgia Institute of Technology

Homework 4, 100 pts.

Due Friday Sept. 12 prior to 11:55pm

(please turn in homework electronically on Canvas)

As always this semester, you are required to solve any and all homework questions alone.

Notation (copied from the lecture notes for ease of reference):

- $\{X\}$ is a set of elements of type X
- m is a message in plaintext
  - m is composed of smaller blocks m_i suitable for individual encryption steps
  - $m = \{m_i\}$
- c_i is ciphertext corresponding to message block m_i
- c is ciphertext corresponding to message m
- Enc_k is encryption with key k
  - $c \leftarrow Enc_k(m)$
- Dec_k is decryption with key k
  - $m \leftarrow Dec_k(c)$
- MAC_k is generation of a message authentication code t with key k
  - $t \leftarrow Mac_k(m)$ or, alternatively, $t \leftarrow Mac_k(c)$
- $<a,b>$ is a concatenation of a followed by b

1) (20 pts.) Hash functions and collisions.
   a. (10 pts.) Use your own words (do **not** copy from **any** source) to describe what is *collision resistance* for a hash function.

Collision resistance means that it is impractical for an adversary to pick any two values that map to the same output hash value when using the same hash function.

Many students incorrectly stated collision resistance is only a function of the probability that two inputs share an output, and did not take into account the ability of an adversary to determine another input through knowledge of the hash function.

b. (5 pts.) Consider MD5 which takes a 512-bit input and produces a 128-bit hash output. Is it possible for each unique input to be associated with a unique output? If your answer is no, for MD5 with a 512-bit input, do at least the majority of inputs have unique associated hash outputs? If your answer is no the majority of inputs do not have unique associated hash outputs, do at least 10% of inputs have unique associated hash outputs? If your answer is that 10% of the inputs do not have unique associated hash outputs, do at least 1% of inputs have unique associated hash outputs? Please explain your answers; a correct sequence of "yes" or "no" answers with no explanations will receive zero points.

No, it is not possible for each unique input to be associated with a unique output. A 512 bit input means there are 2^{512} possible input combinations. A 128 bit output means there are 2^{128} possible output combinations.

Note that $\frac{2^{512}}{2^{128}} = 2^{384}$

This means that, on average, each hash output will have 2^{384} inputs that map to the same output hash.

No, the majority of inputs do not have unique associated hash outputs. No, not even 10% of the inputs will have unique associated hash outputs.

$\frac{2^{512}}{10} \cong 2^{509}$ possible input combinations

Since 2^{509} is significantly larger than the 2^{128} output space, it is not possible for 10% of the inputs to each map to a unique hash output.

No, not even 1% of the inputs will not have unique associated hash outputs.

$\frac{2^{512}}{100} \cong 2^{506}$ = size of the input space

Since 2^{506} possible inputs define a space that is significantly larger than 2^{128}, it is not possible for 1% of the inputs to map uniquely to distinct hash values.

c. (5 pts.) Does *target-collision resistance* (also known as *second preimage resistance*) imply collision resistance? Why or why not?

Target collision resistance says that given a specific *m*, it is hard to find an *m'* that maps to the same hash. This does not imply collision resistance because *m* is specific as opposed to arbitrary.

Another way to say this is that given it is hard to find an *m'* which maps to the same hash value as a particular *m*, this difficulty does not mean that it is hard in general to find any two messages m_1 and m_2 which both map to the same hash value.

2) (30 pts.) Consider the following scenario describing an attack on documentation signed by a hash function.

You are given a legitimate message x_1, a fraudulent message x_2, and a one-way hash function (keyless) denoted by $h()$ which produces an m-bit output where $m = 10$.

The goal of this attack is to substitute the fraudulent message for the legitimate; for example, if $h(x_1) = h(x_2)$ then the attacker is done! However, since this is extremely unlikely, the plan then is for the attacker to modify x_1 and x_2 in minor ways – e.g., introducing meaningless spaces, tabs or extra punctuation – to produce messages x_1' and x_2' with the result that $h(x_1') = h(x_2')$. If this attack is successful, the attacker aims to convince an unsuspecting party to sign x_1' (or x_1) but later implement x_2'.

If $h(x_1) \neq h(x_2)$, the following approach is taken: first a modification x_1' is tried, and then second a modification x_2' is tried. Third, a different modification x_1'' is tried, and then fourth a different modification x_2'' is tried (i.e., the modifications alternate). Assume that all $h(x_1)$, $h(x_2)$, $h(x_1')$, $h(x_2')$, $h(x_1'')$, $h(x_2'')$, etc., values are stored in an efficient manner (in other words, ignore the time it takes to search the stored values). Further assume that each output of the hash function is random, i.e., each time $h()$ is calculated for any x_1, x_2, x_1', x_2', x_1'', x_2'', x_1''', x_2''', etc., the m-bit hash output is random and has no correlation with previous hash outputs. Do not assume that the output of the hash function is necessarily unique; after all, a truly random process does have a non-zero possibility of repeating an output bit pattern.

a. (5 pts.) Consider the starting point with documents x_1 and x_2, and recall that the output distribution of $h()$ is random. What is the probability that $h(x_1) = h(x_2)$? (HINT: the obvious answer is correct!)

$$P\big(h(x_1) == h(x_2)\big) = \frac{1}{2^{10}}$$

b. (5 pts.) Let $\{h(x_1')\}$ denote the set of all hash values of x_1 or of any of the attempted modifications of x_1. Similarly, let $\{h(x_2')\}$ denote the set of all hash values of x_2 or of any of the attempted modifications of x_2. After the fourth step described above (i.e., after a different modification x_2'' is tried), what is the size of $\{h(x_1')\}$ and what is the size of $\{h(x_2')\}$ (i.e., how many elements are contained in each set)?

After the fourth step, each set has three elements:

$$Set1 = \{h(x1), h(x1'), h(x1'')\}$$
$$Set2 = \{h(x2), h(x2'), h(x2'')\}$$

Some students assumed that each set does not contain the hash of the original message, h(x1) and h(x2). While this is not exactly correct, points were given nonetheless.

c. (15 pts.) How many tries are needed to arrive at a 50% chance that one of the hash entries in {*h*(*x₁*')} matches a hash entry in {*h*(*x₂*')}?

Method:

Number of elements in set1	Number of elements in set2	P(no collision between set1 and set2)
1	1	$\frac{2^{10}-1}{2^{10}}$
2	1	$\left(\frac{2^{10}-1}{2^{10}}\right)^{2}$
2	2	$\left(\frac{2^{10}-1}{2^{10}}\right)^{2*2}$
3	2	$\left(\frac{2^{10}-1}{2^{10}}\right)^{3*2}$
3	3	$\left(\frac{2^{10}-1}{2^{10}}\right)^{3*3}$

Let the number of elements in set1 be n_1.
Let the number of elements in set2 be n_2.
Generalizing, we find

$$P(no\ collision\ between\ set1\ and\ set2) = \left(\frac{2^{10}-1}{2^{10}}\right)^{n_1 * n_2}$$

$$P(at\ least\ one\ collision\ between\ set1\ and\ set2)$$
$$= 1 - P(no\ collisions\ between\ set1\ and\ set2) = 0.5$$
$$= 1 - \left(\frac{2^{10}-1}{2^{10}}\right)^{n_1 * n_2}$$

Solve for n_1 and n_2. Note that either (i) n_1 equals n_2 or (ii) n_1 +1 is equal to n_2. We find that $n_1 * n_2 = \frac{\ln(0.5)}{\ln\left(\frac{2^{10}-1}{2^{10}}\right)} = 709$

$$When\ n_1 = 27\ and\ n_2 = 27,$$
$$P(at\ least\ one\ collision\ between\ set1\ and\ set2) = 0.509$$

Number of tries = n_1+ n_2 = 54

Note that due to the previous question beginning 'tries' after the sets were formed (one comparison already done), 53 will be accepted as well (with proper work shown).

d. (5 pts.) As with earlier homeworks, there may be multiple valid assumptions for the answer to part c (just prior to this part, part d, of question 2 on homework 4). Please clearly state at least one important assumption you made in your answer to part c above. NOTE: full credit will be given only for reasonable assumptions (i.e., clearly incorrect assumptions will lose all or nearly all of the points).

We assumed that the hash function outputs are truly random. If the hash outputs are not drawn from a true random distribution, there might be recurring patterns that influence the probability of a collision.

3) (50 pts.) The lecture "Crypto VIII: Two Attacks on Encryption" covered the padding-oracle attack on Construction 3.30.

a. (10 pts.) Describe first stage of the padding-oracle attack on PKCS #5 padding. More specifically, use the notation on the first page of this homework and also use your own words (do not copy from the book chapter section that covers this attack; a recommendation is to use your own notes taken while listening to Lecture 14 Cryptography Part VIII) to describe the first stage of the attack where *b*, the amount of padding, is learned.

Note: It is important to explain all steps and variables used when providing instructions. Many students mentioned Δ or C' without explaining how they are constructed or modified.

The first stage of the padded oracle attack on PKCS #5 is to manipulate one byte of a given ciphertext block at a time until a padding error message is returned. The byte selected reveals to the length of the message, which is what the attacker wants to learn.

Process:

- Valid message intercepted containing IV, C1, C2
- Attacker sends message IV, C1', C2, where C1'=C1 except byte L (the byte number L from the least significant byte where the least significant byte is byte number one) is modified.
    - If a padding error message returned, the message is L bytes long. Else keep going to the next step.
- Attacker sends message IV, C1", C2, where C1"=C1 except byte L-1 is modified.

    - If a padding error message returned, the message is L-1 bytes long. Else keep going to the next step.
- Attacker sends message IV, c1''', c2, where c1'''=c1 except byte L-2 is modified.
    - If a padding error message returned, the message is L-2 bytes long. Else keep going to the next step.
- …
- Repeat until message length b is found.

b. (15 pts.) Continue to describe the padding-oracle attack. In particular, use the notation above and your own words to describe the second stage of the attack where *B0*, the final byte of the message, is learned by the attack. You should assume that *b*, the amount of padding, has already been learned.

The second step is to use the fact that you know the plaintext of byte b and all bytes after it to deduce the value of the previous byte, B0, which is the final byte of the message. This method relies on the fact that if you bitwise $\oplus$ a chosen value with a message, the ciphertext of the modified message is the original ciphertext $\oplus$ with the value you choose. This can be expressed as [m'=m $\oplus$ Δ] if [c' = c $\oplus$ Δ]. This method consists of changing the padded bits from the plaintext value b using a $\oplus$ function with an appropriate Δ so that the plaintext result is b+1. Now the padding error is reintroduced, because the decryption now expects b+1 padding bytes. The attacker would then cycle through every value of B0, the last message byte, until the padding error disappears. If the attacker keeps track of the modification made to B0, the attacker can deduce the value of B0 because the attacker knows what modification was made as well as the fact that the modified B0 corresponds to plaintext b+1. The process can be expressed as follows:

- Attacker finds Δ using m' = m $\oplus \Delta$ to convert the last b bytes of m to value b+1.
- Attacker sends message IV, c1 $\oplus$ Δ, c2
    - If no error message is returned, byte B0 of m is b+1. Else continue to the next step.
- Attacker sends message IV, c1' $\oplus$ Δ, c2, where c1'=c1 except byte B0 = B0 $\oplus$ 1.

    - If no error message returned, byte Bo of m is (b+1) $\oplus$ 1. Else continue to the next step.
- Attacker sends message IV, C1" $\oplus$ Δ, C2, where C1"=C1 except byte Bo = Bo $\oplus$ 2.
    - If no error message returned, byte Bo of m is (b+1) $\oplus$ 2. Else continue to the next step.
- …

Repeat until byte Bo is found

c. (25 pts.) Extend the padding-oracle attack to the byte *B1* which is the byte just before the final byte *B0* of the message. You should assume that *B0*, the final byte of the message, and *b*, the amount of padding, have already been learned.

The second-to-last byte, B1, can be found using the same methodology of finding byte Bo. This time, the padded bytes AND Bo of the plaintext are changed to b+2 using a bitwise $\oplus$ with a vector Δ'. The attacker then uses the same methodology described in part b on the previous page to determine byte B1. The process can be expressed as follows:

- Attacker finds Δ' using m' = m $\oplus$ Δ' to convert the last b+1 bytes of m to value b+2.
- Attacker sends message IV, C1 $\oplus$ Δ', C2
    - If no error message is returned, byte B1 of m is b+2. Else continue to the next step.
- Attacker sends message IV, C1' $\oplus$ Δ', C2, where C1'=C1 except byte B1 = B1 $\oplus$ 1.
    - If no error message is returned, byte B1 of m is (b+2) $\oplus$ 1. Else continue to the next step.
- Attacker sends message IV, C1" $\oplus$ Δ', C2, where C1"=C1 except byte B1 = B1 $\oplus$ 2.
    - If no error message is returned, byte B1 of m is (b+2) $\oplus$ 2. Else continue to the next step.
- …
- Repeat until byte B1 is found

YOU MAY NOT CONSULT HOMEWORK SOLUTIONS OF THESE EXACT PROBLEMS FROM OTHER COURSES. ALL LABORATORY SUBMISSIONS MUST INCLUDE YOUR NAME, COURSE NUMBER, SECTION, AND THE HOMEWORK SET NUMBER. ALL SHEETS MUST BE STAPLED TOGETHER. ALL WRITING MUST BE EASY TO READ (FOR EXAMPLE, YOU MAY HAVE TO WRITE ONLY ON ONE SIDE OF EACH SHEET OF PAPER THAT YOU SUBMIT AND MAY NOT BE ABLE TO USE RECYCLED PAPER). FAILURE TO FOLLOW INSTRUCTIONS MAY RESULT IN ZERO POINTS. ALL WORK MUST BE YOUR OWN. NO

PLAGIARISM IS ALLOWED, AND YOU MUST PROPERLY REFERENCE ALL SOURCES OF YOUR INFORMATION – ALTHOUGH YOU SHOULD NOT LOOK FOR AND MAY NOT CONSULT "SOLUTIONS" AVAILABLE FROM OTHER SOURCES (TO REPEAT, YOU MAY NOT CONSULT HOMEWORK SOLUTIONS OF THESE EXACT PROBLEMS FROM OTHER COURSES!).