Assignment # 4

Homework

Homework problems are a preparation for the quizzes. They are *not* graded. Please use the `mywpi` forum to post questions you have on these problems.

- 11.2, 11.5, 11.6, 11.8, 12.2, 12.3, 12.5

Project

Note: For submissions on `mywpi`: Please submit a single pdf file containing your results. Please submit source code as a separate file, but make sure to have it listed in the pdf as well.

1. 12.4

2. Block ciphers can be turned into hash functions in a very simple way (cf. Section 11.3.2 of the book for a detailed explanation). One such construction is the *Matyas-Meyer-Oseas* construction defined as $H_i = \mathsf{Enc}_{H_{i-1}}(m_i) \oplus m_i$, where m_i is the i-th block of the message, H_i is the internal state of the hash function and the output $h(m) = H_l$ is the last state of the hash function. This construction is considered secure for an appropriate block cipher.

   A slight modification of the *Matyas-Meyer-Oseas* mode results in the following construction:

   $$H_i = \mathsf{Enc}_{H_{i-1}}(m_i)$$

   (a) Draw the block diagram for both constructions.

   (b) Show why the modified construction is not secure by using the decryption function of the block cipher to obtain a collision. Assume H_0 is all-zeros.

3. The goal of this problem is to implement CBC-MAC yourself using a preexisting implementation of AES.

   (a) Your implementation should accept inputs of arbitrary length in bytes. Make sure to provide a single-1 padding (a single 1 followed by zeroes) combined with length strengthening where the length in bits is appended as a 64-bit number.

   (b) Implement the raw CBC-MAC using AES. This block takes the raw message and the key as input; it should call the padding function to create the input to the raw CBC-MAC; finally it should output the last ciphertext output without further processing. Make sure that inputs, outputs and intermediate states process on binary byte values.

   A template for this will be provided.

4. Next, we experiment with SHA-256. Please use any existing implementation (e.g. by importing `hashlib` in Python) you would like for this problem. Please provide the code you used to complete this problem.

   (a) Try hashing some similar strings with SHA-256. By how much do the hashes differ?

   ```
   The quick brown fox jumps over the lazy dog

   The quick brown fox jumps over the lazy doh
   ```

   (b) One of the files of the last projects is the preimage of the following SHA-256 output. Which one is it?

   The hash digest of the file is (in hex):

   ```
   811108d75af8e060ce31cdab7b7c02be062c5ba0db5079a0d87c2e325dfda387
   ```

Bonus Problem

Block ciphers can be turned into hash functions in a very simple way (cf. Section 11.3.2 of the book for a detailed explanation). One such construction is the *Davies-Meyer* construction defined as $H_i = \mathsf{Enc}_{m_i}(H_{i-1}) \oplus H_{i-1}$, where m_i is the i-th block of the message, H_i is the internal state of the hash function and the output $h(m) = H_l$ is the last state of the hash function. This construction is considered secure for an appropriate block cipher.

Let Enc be a block cipher for which it is easy to find *fixed points* for some key, i.e. there is a key k for which it is easy to find inputs x for which $\mathsf{Enc}_k(x) = x$.

1. Draw the block diagram for the Davies-Meyer construction.

2. Show how to find a collision if the Davies Meyer Construction is used with a block cipher for which it is easy to find *fixed points*. You may choose the IV, i.e. H_0, as any constant, but fixed value.

Good Luck and Have Fun!