CBC MAC

Sarthak Mahajan, 2018111005

March 7, 2022

1 Code structure:

- This folder contains mac.py and run.py code files
- mac.py contains the code to generate MAC tags for a given message.
- run.py is used to take inputs and return outputs of the CBC MAC.

2 Instructions to run the code:

- Run the following command in the folder where this document is contained: python3 run.py
- There will be prompts asking you to input the relevant information serially, and will return the output.

3 Explanation of Functions in run.py:

3.1 run():

- It asks for inputs of the choice of tag generation/authentication, key, key length in binary, message, and tag(only for authentication) (all in decimal format)
- Calculates p: the order of the group for DLP and g: the generator of this group.
- prints the tag for generation and whether the tag was correct or not for authentication.

4 Explanation of Functions in mac.py:

4.1 CBC_MAC(m, k, key_len,p,g):

- p,g are in decimal format, they are needed to be passed to the dlp function. p is the largest prime possible in n bits, where n= length of the key, g is a random number in the range(1,p-1); note that this is fixed for the program in terms of p(so that same seed gives same value for different iterations of the program), but it could be any random number.

- all inputs are in decimal format, m is the message to be authenticated, k is the key, key_len is the key size in binary format.
- Returns the tag of the message in binary format using the CBC mode of operation which handles messages of different sizes
- First we convert m into binary format, and cut it to chunks of size key_len, if the last chunk's size is lesser the key's size we pad it with zeros on the right side
- We use the CBC mode of operation using a for loop, where k, key_len serves as the key to the PRF used in each block of the CBC.
- Uses pad,PRF functions