Fixed length collision resistant hash function from DLP

Sarthak Mahajan, 2018111005

March 7, 2022

1 Code structure:

- This folder contains fhash.py and run.py code files
- fhash.py contains the code to create the output of a fixed length collision resistant hash function hashing 2n to n bits.
- run.py is used to take inputs and return outputs of the fixed length collision resistant hash function

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 key length in binary, the 2 n bit numbers whose concatenation needs to be hashed to length n, where n= key length (all in decimal format)
- Calculates q: the order of the group for DLP ,g: the generator of this group, and h: a random element in the group
- prints the hashed value

4 Explanation of Functions in fhash.py:

4.1 fixed_hash$(x_1, x_2, key_len, q, g, h)$:

- q,g,h are in decimal format. q is the largest prime possible in n bits, where n= length of the key, g ,h are random numbers in the range(1,q-1); note that they are fixed for the program in terms of q(so that same seed gives same value for different iterations of the program), but they could be any random number in the given range.
- all inputs are in decimal format,key_len is length of the key in binary format, let it be equal to n , x_1, x_2 are decimal format of n bit numbers.
- It returns a number in decimal format which is the hash of the 2n bit number x_1x_2 to a n bit number
- the hash is created using the formula: $((g^{x_1} \mod q)(h^{x_2} \mod q)) \mod q$, where g is generator of the group and h is any random of the group Z_q
- q is a number with the same length in binary format as the key