Hashes: Implementation of a Weak Hashing algorithm and experimenting with hash length extension attack

In this lab, you will be experimenting with cryptographic hash functions. At first you will experiment with avalanche effect on hash functions, implement a weak hashing algorithm and give evidence of its unsuitability as a cryptographic hash function. You will also learn about hash length extension attack and write program that takes advantage of this.

This exercises must be submitted by Saturday 11:59PM 13 August 2016 at cseducrypto2016@gmail.com

3.1 Avalanche Effect (5 points)

Files

1. 3.1_input_string.txt: original string
2. 3.1_perturbed_string.txt: perturbed string i.e exact copy of the original string with one bit flipped

We're going to use these two strings to demonstrate the avalanche effect by generating the SHA-256 hash of both strings and counting how many bits are different in the two results (a.k.a. the Hamming distance.)

What are their SHA-256 hashes? Verify that they're different.
($ openssl dgst -sha256 3.1_input_string.txt 3.1_perturbed_string.txt)

Compute the number of bits different between the two hash outputs and submit it as a hex string in solution31.hex.

3.2 Weak Hashing Algorithm (5 points)

Files
1. 3.2_input_string.txt: input string

Below you'll find the pseudocode for a weak hashing algorithm we're calling WHA. It operates on bytes (block size 8-bits) and outputs a 32-bit hash.

```
WHA:
Input {inStr: a binary string of bytes}
Output {outHash: 32-bit hashcode for the inStr in a series of hex values}
Mask: 0x3FFFFFFF
outHash: 0
    for byte in input
        intermediate_value = ((byte XOR 0xCC) Left Shift 24) OR
                             ((byte XOR 0x33) Left Shift 16) OR
                             ((byte XOR 0xAA) Left Shift 8) OR
                             (byte XOR 0x55)
outHash =(outHash AND Mask) + (intermediate_value AND Mask)
return outHash
```

First, you'll need to implement WHA in the language of your choice. Here are some sample inputs to test your implementation: WHA("Hello world!") = 0x50b027cf and WHA("I am Groot.")=0x57293cbb

Your goal is to find another string that produces the same WHA output as the given input string. In other words, demonstrate that this hash is not second preimage resistant.

Submit a string with the same WHA output as 3.2_input_string.txt and submit it in *solution32.txt*. Also, submit the code for WHA algorithm code with the name *wha_collision.py*

3.3 Conduct a Length Extension Attack (10)

Please read the corresponding documentation from the resources section of course webpage.

Files
1. 3.3_query.txt: query
2. 3.3_command3.txt: command3
3. pymd5.py: A python file containing useful functions

One example of when length extension causes a serious vulnerability is when people mistakenly try to construct something like an HMAC by using hash (secret k message), where k indicates concatenation. For example, a web application with an API that allows client-side programs to perform an action on behalf of a user by loading URLs of the form:

http://www.example.com/api?token=b301afea7dd96db3066e631741446ca1&user=admin&command1=ListFiles&command2=NoOp

where token is MD5(user's 8-character password k user=.... [the rest of the URL starting from user= and ending with the last command]).

Text files with the query of the URL 3.3_query.txt and the command line to append 3.3_command3.txt are provided. Using the techniques that you learned in the lectures and without guessing the password, apply length extension to create a new query in the URL ending with command specified in the file, *&command3=DeleteAllFiles*, that is treated as valid by the server.

Historical fact: In 2009, security researchers found that the API used by the photo-sharing site Flickr suffered from a length-extension vulnerability almost exactly like the one in this exercise.

Submit A Python script named len_ext_attack.py and text file solution33.txt that should contained the updated query

Python script should perform such that:
1. Modifies the query so that it will execute the *DeleteAllFiles* command as the user.
2. Verify that your length extension works.