**CS 4732/57322 Homework #6**

***Due electronically by midnight July 27th, 2020***.

For submission, if done on paper please scan and submit as a pdf. If done in word, please submit the .docx or .doc format.

**IMPORTANT**: Clearly indicate outside resources utilized and sign below. Failure to cite use of outside resources will be reported for appropriate disciplinary actions. Note that discussions with other students are encouraged; copying – with or without modifications – is unacceptable and will also be reported.

I discussed one or more problems with the following people:

I hereby certify that any outside resources utilized, other than the textbook and class materials, are clearly cited. All other material I provide for this homework submission is my own original work.

*Printed name*

1. (6 points) Give me the requirements for a secure hash function. Which ones would not be needed for a simple hash lookup table?

Pre-image resistance

Second pre-image resistance

Collision resistance

Pseudo-randomness

The ones not needed for a hash lookup table is the image resistance. The values however will need to be unique and therefore cant collide

2. (12 points) Consider a hash function that works on messages that are composed of integers. The hash value is calculated by multiplying all the individual values of the digits together and then taking that value modulo n.

a) Is this a secure hashing algorithm? In particular, what requirements of a hash function does it violate and demonstrate this.

The first factor is that **you cant find two different messages with the same hash value**. For example, if we take 1234, then the hash value will be (1\*2\*3\*4)mod 5 =4

All strings containing 1,2,3,4 like (2,3,4,1 or 3,1,2,4, ect) will hash to the same value 4. Hence it is not a secure algorithm.

b) Consider a hashing function that takes the sums of the squares of individual digits and then does modulo n. For example, the number 135 would be hashed to (1\*1 + 3\*3 + 5\*5 ) mod n. Make an argument for how secure you think this hash algorithm would be.

**This algorithm is also as strong as the previous one. Strings containing the same set of numbers will produce the same hash value, which also is not very strong, as they both have hash collision as a weakness.**

3. (6 points) Suppose you wanted to create a block cipher that was based at least in part on a hash function. We know that hash functions are one-way, while a cipher needs to be reversible in order to decrypt it. Come up with a way that you could use a hash function in this way. Looking over DES could be a good way to start thinking about this question.

You could use the feistel structure which is what DES uses.

\*\* I got the following from WIKI\*\*

Schematics:  you would use the hash function for the "F" part, which combines one (sub)key and one half of the current block, to produce a value which is to be XORed with the other half of the current block. The beauty of the scheme is that the "F" function is always invoked in the same direction, both for encryption and for decryption. Therefore, it can be a one-way function, like a hash function.

Luby and Rackoff have demonstrated in 1988 that the Feistel scheme offers remarkable security with as little as four rounds, provided that the "F" function is "perfect" and that the cipher block size is big enough (to get the standard "128-bit security" out of the Luby-Rackoff proof, you need 256-bit blocks).

**Of course**, any concrete hash function cannot be really "perfect" and there are a lot of subtle details which can destroy the security of the best thought cipher structure. As usual, you are strongly advised **not to build your own crypto**.

Also, if you build such a cipher, you will probably notice that the resulting performance is disappointing. With a secure hash function like SHA-256, you could expect an encryption bandwidth roughly 20 times lower than what AES would get you.

4. (8 points) In what order should the signature function and the confidentiality function be applied to a message and why? Give an example of doing it in the wrong order and the problem that this could cause or allow.

Well a signature function uses a private key of a sender to sign or encrypt the hash of the message and uses the public key of the sender by the receiver to verify(decrypt) the message. The confidentiality function uses the private key of the receiver by sender to encrypt the message and uses the public key f the receiver to decrypt the message.

So the **signature function should be performed first** and then the outer confidentiality function should be applied second. If there is a dispute, third parties should review messages and its signature, if the signature is in the inner operation then the recipient can store the plaintext message and its signature for later use in the dispute resolution. Whereas if a signature is calculated on the encrypted message, then the third-party needs access to the decryption key to read the message.

A digital signature does not provide confidentiality (as non-encrypted data can bear a signature). Knowledge of a public key does not guarantee the identity of the owner of the corresponding private key, and so the encryption of information using a public key cannot prevent encryption falling into the wrong hands. Before a public key can be safely used to encrypt or decrypt information, the identity of the holder of the private key must be assured. **This assurance is provided by a digital certificate which binds the public key to the identity of the private key’s owner. THIS IS WHY IT NEEDS TO BE FOLLOWED.**

5. (6 points) What problems do MACs solve that simple use of public-key encryption to provide authentication do not?

**Public-key encryption does not provide integrity. MAC provides integrity while encryption provides confidentiality.**

**Mac doesn’t provide confidentiality by itself without any encryption, so we often have to combine both encryption and mac to achieve confidentiality**

6. (6 points) Would you expect a cryptographic hash function to be useful if you wanted to create a cryptographically secure PRNG? If yes, point to specific properties of a good hash function that would make this so. If not, describe why not.

**Yes**, Cryptographic hash functions are useful to create a secure PRNG, If the hash function has the properties like the hask function should uniformly distributes the data over the possible set of hash values and it should be giving the same value to same number and it should be having a fixed length of enciphered text length called hash value and after that the hash value should be used to store the value and later on it can be used to verify it.