**CS 4732/57322 Homework #6**

***Due electronically by midnight July 28th, 2024***.

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.

Jamie Harris

*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?

The requirements for a secure hash function are variable input size, fixed output size, computational efficiency, pre-image resistance (one-way), second pre-image resistance (weak collision resistance), strong collision resistance, and pseudorandomness. Pre-image resistance, second pre-image resistance, and strong collision resistance are not needed for simpler hash applications.

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.

This would not be a secure hashing function. There is a fairly large chance for collisions. Since multiplication is an associative and commutative operation, any permutation of the same digits will arrive at the same value. For example, take 135 and 351. You would then get 1\*3\*5 mod n and 3\*5\*1 mod n, which will both give 15 mod n.

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. Show that this is a bad hashing function by showing a collision.

Two types of collisions can occur:

1. Permutations of the same digits. IE 135 and 351 give 1\*1+3\*3+5\*5 and 3\*3+5\*5+1\*1 respectively, which are both equal to 37.
2. If you take a number, say 135, and add one or more 0 digits to it, such as 1035, 1350, 1305, etc, you get 1\*1 + 0\*0 + 3\*3 + 5\*5, etc. which gives you the same result.

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 hash one half of the current message block before XORing with the other half of the block.

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.

You should sign, and then encrypt the signed message. This ensures that you are checking the authenticity of the original message instead of the ciphertext. If you encrypted first and then sign, it would be possible for someone to replace the original ciphertext with a different ciphertext before you sign, allowing a different message to be sent without verifying authenticity.

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

MACs allow you to know for sure that a message has not been tampered with, even if that data does not have any underlying structure, such as noise data.

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, I would expect a cryptographic hash function to be useful in the creation of a PRNG. The preimage resistance means it wouldn’t be possible to find the initialization settings of the PRNG just by seeing its output, and collision resistances mean you wouldn’t be able to generate the same random number more than once, so you don’t have to worry about, for instance, generating the same cipher key multiple times.