University of Bahrain
College of Information Technology
Department of Information Systems

ITCY 201 – Fundamental of Cybersecurity

Practical Session 5

Assignment # 5 (Hashing)

Last date for submission: 10-11-2024

Learning Objectives

In this Lab, you will learn how hash functions work to enable integrity and nonrepudiation. At the end of this lab exercise, you'll be able to

- Hash plaintext into a message digest
- Understand how hashing works and why it's used
- Understand how hashing is different than encryption

Lab Materials and Setup

The materials you need for this lab are

- A web browser with an Internet connection

Encryption is a two-way function. Take plaintext and a key, put them in an encryption algorithm, and ciphertext output is produced. Take ciphertext and a key, put them in the algorithm, and plaintext output is produced. Hashing is not encryption, nor does it work like encryption. Hashing is a one-way function. Take an input, put it in a hash function, and an output called a message digest, also known as a hash, is produced. You cannot reverse a hash by putting it back into the hash function to produce the original input.

Step 1: The first two hash function characteristics, listed at the beginning of this Lab, include variable-length input producing fixed-length output and the slightest change in the input yielding a completely different output. Let's see both of those characteristics in action!

1. Go to www.fileformat.info/tool/hash.htm.
   In the String Hash textbox type **bob**. Then click the Hash button.
2. Scroll down to see all of the message digests that were simultaneously calculated. The longer the output, the more secure the hash function, because with longer outputs it is harder to find a collision for multiple inputs.

3. Open Notepad by typing **Notepad** in the Windows search box. Then click the Notepad icon. Copy and paste the SHA-256 hash calculated in Step 1c into Notepad.
4. Go to any website and copy a bunch of text (highlight the text and press CTRL-C).
5. Paste (press CTRL-V) the text into the String Hash textbox from Step 1b and click the Hash button.
6. Copy and paste the SHA-256 hash into Notepad, below the first hash. As you can see, even though the first input was significantly shorter than the second output, the size of the message digests in the output of the hash function are exactly the same.
7. Back in the String Hash textbox, type **Bob** and click the Hash button. Scroll down to see all of the message digests that were simultaneously calculated.
8. Copy and paste the SHA-256 hash into Notepad, on the third line.
9. Compare the hash of bob to the hash of Bob. You can see that, just by changing the lowercase *b* to an uppercase *B*, the hashes are radically different. Also, notice that there is only a single bit difference in the binary representations of b and B:

   b: 62 (hex), 01100010 (binary)

   B: 42 (hex), 01000010 (binary)

Step 2: The third hash function characteristic, listed at the beginning of this Lab Exercise, is that hashing is a one-way function, which means you shouldn't be able to go back to an input when provided the hash (preimage resistance).

Imagine blending a banana, some strawberries, milk, and vanilla syrup in a blender. While enjoying your smoothie, you think to yourself, "I wish I could have that banana back." Sorry. You can't get the banana back. You might know the process of taking a banana out of a smoothie and reconstructing it, but it doesn't mean that could be done. This is how one-way functions can't be reversed. Get it?

1. **Think of two numbers that when multiplied together equal 100.**
2. **Pick a classmate and ask them to guess the factors you chose.**
3. Theoddsaretheirguesswouldbewrong.100×1?50×2?25×4? 20 × 5? 10 × 10? They wouldn't know what two numbers you picked.

Hash functions involve doing some math and throwing away the inputs; then, doing some more math, and throwing away the subsequent inputs in the same way. Hashing is considered a *one-way function* because it's not feasible to try all possible combinations in a realistic amount of time to go back the other way (in our example, starting with the product of 100) and wind up with the original numbers you started with (in our example, one of the pairs of factors of 100). Although it might be easy to go through all numbers that when multiplied equal 100, when multiple rounds of math are used and one output is at the end, how can you go back to the original numbers that came before many rounds of math? Trying all possible combinations is not feasible. Let's say you multiplied that 100 by 8. Now you've got 800, and you have *two*

rounds of inputs to get back to the original factors of 100. This example is using only multiplication. Hashing functions, of course, are much more mathematically intensive and use much larger numbers.

Step 3: Now it's time to see how hashes can verify the integrity of files that you download from various websites. How do you know that file you're downloading is the original and not an altered version laced with malware or spyware?

- Choose two files to download from the Internet. The files should be similar to each other, but they are taken from two different websites.
- Use this website to hash both files: https://hash-file.online/
- Select the first file to hash and take a screenshot of the result.
- Select the second file to hash using the same hash function used for hashing the previous file, and take a screenshot of the result.
- Compare both results and discuss the difference between them.

END OF LAB 5