CSCI 3403 **Erika Bailon**
Homework 4 **Fall 2018, CU-Boulder**

1. **Not in the book question** Bloom filters

   Question 1A) Show the resulting Bloom Filter by marking each bit as either 0 or 1.

0	1	2	3	4	5	6	7	8	9
0	1	1	0	1	0	0	0	1	1

   Question 1B) List one value for H3(mouse) that would cause the "mouse" to be rejected as a password or explain why no value of H3(mouse) could cause this.

   One of the values that would $H3$ to be rejected is **1**, the other options are 2 and 9. And this is because those numbers already have a has, they have a one in the table so it would not be possible for "Mouse" to hash there. It CANNOT use it.

   Question 1C) List one value for H3(mouse) of H3(mouse) that would cause the "mouse" to be accepted as a password or explain why no value of H3(mouse) could cause this.

   One of the values that would cause "mouse" to be accepted is **3**, the other options are 0, 5, 6, or 7. This is because those places have a 0 on the table this means they haven't been hashed so then is free for "mouse" to use it.

2. **Review 3.1** In general terms, what are four means of authenticating a user's identity?

   (a) Something the individual knows: Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions.

   (b) Something the individual possesses: Examples include electronic keycards, smart cards, and physical keys. The type of authenticator is referred to as a *token* .

   (c) Something the individual is (static biometrics): Examples include recognition by fingerprint, retina, and face.

   (d) Something the individual does (dynamic biometrics): Examples are voice patter, handwriting, typing rhythm.

3. **Review 3.8** Define the terms *false match rate* and *false nonmatch rate*, and explain the use of a threshold in relationship to these two rates.

   A false match rate is the frequency with which biometric samples from different sources are erroneously assessed to be from the same source.
   The false non-match rate is the frequency with which samples from the same source are erroneously assessed tp be from different sources.
   The difficulty is that the range of matching scores produced by two individuals, one genuine and one an imposter, compared to a given reference template, are likely to overlap.
   Once they overlap, there is a line named the threshold that is basically the line that gives you the option of deciding what is more probably to happen and how you wanna manage that.
   For example, if your threshold line is leaving more area to the left, then the authenticity check will not always work because it will think that is not you even thought it is you. If you leave more area to the right, then the people trying to access will be able to access because even thought is not you, it will think that is you. This is explain in the book with this sentence - "By moving the threshold, left or right, the probabilities can be altered, but note that a decrease in false match rate results in an increase in false non-match rate, and vice versa. "

4. **Problem 3.3** Assume passwords are selected from four-character combinations of 26 alphabetic characters. Assume an adversary is able to attempt passwords at a rate of one per second.

   (a) Assuming no feedback to the adversary until each attempt has been completed, what is the expected time to discover the correct password?

   Since passwords are selected from a four-character combination, then we can do 26^4 to know how many combinations we can have. However, as the teacher said it in class, we need to divide that by two to get the average of attempts. So then we get

   $$\frac{26^4}{2} = \frac{456976}{2} = 228488 \text{ attempts.}$$

   Now, the adversary is able to attempt one password per second, which means it will take him 228488 seconds, but we have 3600 seconds in an hour so:

   $$\frac{228488}{3600} = \mathbf{63.47\ hours}.$$

(b) Assuming feedback to the adversary flagging an error as each incorrect character is entered, what is the expected time to discover the correct password?
If the adversary gets an error flag every time he puts a letter then we only need to do each letter for each one of the "slots" or "place" in the 4 letter string and then to take the average of attempts we do

$$\frac{26*4}{2} = \frac{104}{2} = 52 \text{ attempts.}$$

but then each attempt only takes him a second so it would take the adversary **52 seconds**.

5. **Problem 3.4** Assume source elements of length k are mapped in some uniform fashion into a target elements of length p. If each digit can take on one or r values, then the number of source elements is r^k and the number of target elements is the smaller number r^p. A particular source element x_i is mapped to a particular target element y_j.

   (a) What is the probability that the correct source element can be selected by an adversary on one try?
   Since the source is of length k, then all numbers of source elements are given by r^k. We have a formula to calculate the probability of a successful try, given by geometric series, and plugging the values we have we obtain:

   $$\mathbf{\frac{1}{r^k}}$$

   (b) What is the probability that a different source element $x_k(x_i \neq x_k)$ that results in the same target element, yj, could be produced by an adversary?
   For this question we want to divide the number of possible sources which are r^k, that are mapping to y, by the number of total sources. However, we need to subtract the one source that is different, who's being divided by the total number of sources, this is:

   $$\frac{\frac{r^k}{r^p}}{r^k} - \frac{1}{r^k}$$

   Following the math to reduce this expression we multiply everything by r^p, so we get:
   $\frac{r^k}{r^p*r^k} - \frac{r^p}{r^k*r^p}$

Simplifying even more, since we have the same denominator we get:

$$\mathbf{\frac{r^k - r^p}{r^k * r^p}}$$

(c) What is the probability that the correct target element can be produced by an adversary on one try?
Because of the same formula and logic of a) we obtain:

$$\mathbf{\frac{1}{r^p}}$$

6. **Problem 3.6**. Assume passwords are limited to the use the 95 printable ASCII characters and that all passwords are 10 characters in length. Assume a password cracker with an encryption rate of 6.4 million encryptions per second. How long will it take to test exhaustively all possible passwords on a UNIX system? If we have 95 characters and 10 ”slots” or ”places” for the letters, then we have a total of 95^{10} combinations. Assuming the rate or encryption is 6.4 million encryption per second, then:

$$\frac{95^{10}}{6.4 million} = \frac{95^{10}}{6{,}400{,}000} = 9355264675600 \text{ seconds!}$$

As we saw before we have 3600 seconds in 1 hour, so $\frac{9355264675600}{3600} = 2598684632$ hours! We can reduce that number even more, we have 24 hours in a day so $\frac{2598684632}{24} =$ 108278526.3 days! Reduced even more, we have 365 days in a year so $\frac{108278526.3}{365} =$ 296653.4968 years!! Rounding, is about **296,654 years**, which is almost 300 millennia!

7. **Problem 3.8**. The inclusion of the salt in the UNIX password scheme increase that difficult of guessing by a factor of 4096. But the salt is stored in plaintext in the same entry as the corresponding ciphertext password. Therefore, those two characters are known to the attacker and need not be guessed. Why is it asserted that the salt increases security?

Adding salt to the passwords increases the difficulty for the attacker to guess the passwords for several reason. One, the passwords containing a salt will not hash to the

same place even if the password is the same because the salt is a random generated number that's added to the password so therefore, when this is encrypted, it will not have the same value as the other password that contains the same word, simply because it contains a different salt. Second, salt will cause a larger rainbow table so then the attacker will have more difficulty to find patterns because there won't be one.
If the passwords didn't have a salt, the attacker would easily find a dictionary that can be compared to all the encrypted passwords in the file and decrypt them. So the salt adds an extra challenge to the attacker.

8. **Problem 3.9**. Assuming you have successfully answered the preceding problem and understand the significance of the salt, here is another question. Wouldn't it be possible to thwart completely all password crackers by dramatically increasing the salt size to, say, 24 or 48 bits?

   The size of the salt wouldn't really matter because it depends more in how many passwords you have to populate the rainbow table. Increasing the size of the salt will be beneficial if you increase the size of the table. Therefore, it would not be possible to thwart completely all passwords by increasing the salt size because there would be less patterns to follow and more hash functions to try to figure out.