Maureen Hsu(yh863), Cealia Yu-Tung Pai(yp332)

Exercise 3.1: Consider an attacker that wants to guess some user's password, where the user's password pw is chosen according to a distribution p, i.e., p(pw) represents the probability that pw was chosen by the user. Let q be the number of guesses the attacker can make.

1. In an offline brute-force guessing attack, what limits the number of guesses q made by an attacker? In an online (also called remoted) brute-force guessing attack, what limits the number of guesses q made by an attacker?

a. Offline:
   i. Computing resources
   ii. Time limitation: it takes time to run brute-force
b. Online:
   i. The issue of maximum password attempt limits: certain IP is banned after failing more than k times
   ii. Communication bandwidth
   iii. Computing resources
   iv. When "challenge" has changes in Challenge-Response

2. Suppose the adversary makes some number of q guesses. What is the optimal guessing strategy and what is its probability of success (as a function of p)? This is called the q-success probability of the distribution p.

Optimal guessing strategy: guessing from most used passwords in the order of rank
P(right password in guessed first q passwords) = sum of probability of password of rank 1~q in distribution p.

3. Define Shannon entropy and given an example of a distribution p whose q-success probability is high and whose Shannon entropy is also high. Explain why Shannon entropy is a misleading estimate of password strength.

Take to follow distribution for example,

N = 10000000
$p_1 = 1/40$
$p_2 = (1 - 1/40)/9999999 \approx 1 / 2^{20}$
…
$p_N = (1 - 1/40)/9999999 \approx 1 / 2^{20}$
—----------

In this case, Entropy ≈ 19, that means this has an overall entropy of about 19 bits. High entropy tells us that we have an extremely strong password. However, the truth is an attacker will have 1/40 chance to get the correct password with one guess.

Exercise 3.2: Salts for password hashing prevent use of rainbow tables and, more generally, ensure that cracking effort against one user's hash is not reusable for another user's hash. Some web services will additionally employ a pepper, which is a single random value also

added to the password hashing input and stored separately from the password hashes. Explain the purpose of a pepper.

It is used in the selected hash function, added to an input such as password, to make the hash value more not reversible.

Exercise 3.3: Describe how you would modify the web service to limit the efficacy of online guessing attacks.

Use “Challenge-Response” authentication, the user and the system share a secret (key or password), **challenge** is that the system presents the user with some string and user **responses** based on the secret and the challenge. The challenge value is different each time. The challenge expires before the minimum time for brute-force attack to work. The hash function should also be designed to make it difficult to recover secrets from responses.

Exercise 3.4: A colleague suggests you might strengthen password selection by enforcing a password composition policy, such as requiring an uppercase letter, lower case letter, and symbol. Provide an explanation of why this is a bad idea. What should you do instead to encourage stronger password selection?

Too many restrictions would cause people to stick to one password because creating a valid and memorizable password is too difficult. We think a better approach would be suggesting users to use a strong password that can be stored in an online keychain.