CS 5435 – Security and Privacy Concepts in the Wild

Homework 1

Name: Arystan Tatishev(at855) , Nikhil Reddy(nc463) *Spring 2024*

Exercise 3.1

1. **Number of Guesses**

   - **Offline Brute-Force Attack**

     The limit on the number of guesses q is primarily determined by the computational resources available to the attacker (such as CPU/GPU power for calculating hash functions) and the efficiency of the attack method (e.g., the use of rainbow tables or advanced cracking software). There's no inherent limit set by the target system on the number of guesses an attacker can make.

   - **Online (Remote) Brute-Force Attack**

     The number of guesses q is limited by security mechanisms implemented by the system, such as account lockouts after a certain number of failed attempts, CAPTCHAs, or rate-limiting, which slows down the attack by forcing delays between guess attempts.

2. **Optimal Strategy**

   - **Optimal Strategy**

     The optimal strategy for an attacker, given a number of guesses q, is to guess passwords in the order of their probability $p(pw)$, starting with the most likely password.

   - **Probability**

     The success probability, given q guesses, is the sum of the probabilities of the q most likely passwords. If p_i is the probability of the i-th most likely password, and the passwords are guessed in descending order of their probability, then the success probability $P_{success}$ for q guesses is given by:

$$P_{success} = \sum_{i=1}^{q} p_i$$

3. **Shannon Entropy**

   - **Definition**

Shannon entropy is a measure of the unpredictability or randomness of a distribution. For a password distribution p, Shannon entropy $H(p)$ is defined as:

$$H(p) = -\sum p(pw) \log_2 p(pw)$$

It quantifies the average amount of information (in bits) required to identify the password.

- **Example**

  Consider a distribution where a small set of passwords is very common (high $p(pw)$ for these passwords), but there is also a long tail of rare, unique passwords. This distribution can have high Shannon entropy if the rare passwords are numerous and diverse enough, indicating a high level of unpredictability overall. However, the q-success probability can also be high if the attacker guesses the common passwords first.

- **Why it is misleading**

  If a significant portion of users choose common passwords (which is often the case in real-world scenarios), an attacker using an optimal guessing strategy (guessing more common passwords first) can achieve a high success rate with relatively few guesses. This discrepancy occurs because Shannon entropy considers the average uncertainty across the entire distribution, without accounting for the distribution's shape or the concentration of high-probability outcomes.

Exercise 3.2

Peppers, unlike salts, are confidential and uniform across an entire system, without randomization. They are intentionally kept undisclosed and are not stored alongside salts and hashes in the database. The purpose of this separation is to ensure the effectiveness of peppers. These confidential elements are securely maintained in a distinct section of the site application. This practice is crucial as it prevents potential hackers, even if they gain access to passwords and salts, from deciphering any passwords due to their lack of knowledge about the pepper.

Exercise 3.3

- **Lockout after X failed attempts** Implement an account lockout policy where, after a certain number of failed login attempts (e.g., 5), the account is temporarily locked for a period of time (e.g., 15 minutes).
- **CAPTCHA Integration:** Require users to solve a CAPTCHA after a few consecutive failed login attempts to distinguish between automated bots and humans.
- **Rate Limiting:** Implement rate limiting on login attempts from the same IP address or user account to prevent automated tools from performing rapid guessing attacks.

- **Multi-Factor Authentication (MFA):** Require or offer users the option to enable MFA or 2FA. This adds an additional layer of security, making it harder for attackers to gain unauthorized access.
- **Anomaly Detection:** Use anomaly detection systems to monitor for unusual login attempts and respond accordingly.
- **Strength meters** Nudge/force users to pick strong, unique passwords.
- **Implementing Device Recognition:** Recognize and trust previously used devices or locations, requiring additional verification for new ones.
- **Compromised Credentials Check:** Check if password is in the known breaches

Exercise 3.4

Enforcing a strict password composition policy, such as requiring an upper case letter, lower case letter, and symbol, can be counterproductive and may not necessarily lead to stronger password security. Here are some reasons why this approach may be considered a bad idea:

Predictability: Users may resort to predictable patterns when forced to include specific types of characters. For example, they might use a common word followed by a number and a symbol, leading to passwords that are easily guessable.

Difficulty and Memorability: Forcing users to include specific character types can make passwords harder to remember, leading to an increased likelihood that they will write them down or use easily guessable variations.

False Sense of Security: A password's strength is not solely determined by its composition but also by its length and uniqueness. Enforcing a composition policy might give a false sense of security without addressing these crucial aspects.

Usability Issues: Strict composition policies may discourage users from creating strong passwords, leading to frustration and resistance. Users may choose weaker passwords or reuse passwords across multiple accounts.

Instead, nudge the users into selecting a password that is difficult to guess and take the help of the strength meter while setting their password. Users should also be encouraged to enable multi-factor authentication in order to secure their account.