Computer Security (COM-301)

Authentication – Passwords & Biometrics

Salts

When storing passwords in a database, which of the following statements about salts are true:

(a) The salt is stored in the clear next to the hash of the password.

(b) We use salts because they are large numbers, so concatenating them with the password will result in hashes that are very long.

(c) Salts increase the cost of brute force attacks because users often select common passwords from dictionaries

(d) Salts increase the cost of brute force attacks because calculating the hash of the concatenation of two strings is very costly.

The correct statements are (a) and (c). (b) wrong since hash output is fixed size. (d) is wrong because salts are not primarily used to increase the cost of calculating the hash but rather to protect from precomputed tables by adding uniqueness.

Protecting from snapshot attacks

When designing a password-based authentication system, which of the following mechanisms should you use to mitigate the impact of offline attacks when the adversary gets access to the database:

A) Requiring knowledge of a nonce (random number) that has just been sent to the authenticating principal before accepting a password

B) Concatenating a salt with the password before hashing

C) Using a fast hash function

D) Storing the hash of the password beside a hash of a random salt

(B) All the others do not protect against an adversary that has access to the database

Password updates

AcmeCorp forces all employees to come up with new passwords every three months. Provide at least one advantage and one disadvantage of such policy.

Justify using the security principles.

Examples of valid answers:

Advantage:
In case of password database compromise, all (hashed) passwords will become irrelevant at most in three months (as opposed to never changing passwords). (failsafe principle)

Mechanism is simple (economy of mechanism) - changing the password every three months is simpler compared to using 2FA for instance.

Instead of password database compromise -> spearphished password, keylogged password, network-sniffed password; prevents password reuse across services. (increase work factor)

Disadvantages:
Uncomfortable for employees, will likely result in them coming up with weak passwords (psychological acceptability principle, see this post: https://www.ftc.gov/policy/advocacy-research/tech-at-ftc/2016/03/time-rethink-mandatory-password-changes)

Password still common point of failure (least common mechanism)

What you know or who you are

You are opening a new bank account at NotSoSec Bank and are asked to choose between two different authentication mechanisms to access your online account.

Option 1 is to use an 8 character password that would be stored as a salted hash on the bank's server.

Option 2 is to use your fingerprint with a dedicated fingerprint reader provided to you by the bank. To authenticate, the reader sends your fingerprint to the bank's server where it is processed and compared to the stored biometric template.

Once you have decided on the authentication mechanism you will not be able to change it.

Which of the two options would you choose in this case? Why do you think this is the favourable option? Justify your choice in relation to the use case in the question. Take into account security considerations, as well as the usability advantages and disadvantages of these mechanisms mentioned in the lectures.

Example answer. (Almost anything is valid in this question as long as the justification works)
A password you can change your fingerprint you can not. As the sensitive information that is supposed to be protected is stored there long term (you want to keep this bank account) you want to be able to change authentication code after a compromise of the server that stores the secret. Harder to do with fingerprint than with password.

focus on storage.

When machines get compromised

You are hired at WhoAreYou.com to audit their authentication system. They use login and password, and their main engineers store every password in a file `securepass.txt` where each line reads:

```
H(password || k), salt
```

where `k` is a symmetric key stored in another file on the same machine.

Does this scheme protect the confidentiality of the passwords if an adversary compromises the machine hosting the file `securepass.txt`? Justify your answer

No it does not, if you control the machine, you have k. Also the salt is not used (?!)