زانکۆی سەلاحەدین - هەولێر
Salahaddin University-Erbil

Information Security

Lecture Two

Authentication- Part 1

Instructor: Newroz N. Abdulrazaq

Science College- Department of Computer Science & I.T.

newroz.abudlrazaq@su.edu.krd

Text Book: Mark Stamp, Information Security: Principles and Practice.+ Educational Websites.

Lecture2: Authentication

- Introduction.
- Password.
  - ✓ Keys Versus Passwords.
  - ✓ Choosing Passwords.
  - ✓ Attacking System Via Passwords.
  - ✓ Passwords File (Hashed & salted).
  - ✓ Math of Password Cracking.

Introduction

- Authentication is the process of determining whether a user (or other entity) should be allowed access to a system.
- Authorization: Authenticated users are allowed access to system resources. However, an authenticated user is generally not given carte blanche access to all system resources.
- Authentication: Are you who you say you are?

  Authorization: Are you allowed to do that?

Authentication Methods

A human can be authenticated to a machine based on any of the following:

- Something you Know.
  - ✓ e.g.: Passwords.
- Something you Have.
  - ✓ e.g.: ATM Card.
- Something you are.
  - ✓ e.g.: Fingerprint.

Some Thing You know?

- **Passwords.**

- **Lot of things acts as Passwords:**
  - ✓ PIN.
  - ✓ Social Security Number.
  - ✓ Date of Birth.
  - ✓ Name of your pet. …etc.

Passwords

- An ideal password is something that you know, something that a computer can verify that you know, and something nobody else can guess.
- Condition of creating password:

1. Should be: at least as strong as cryptography keys. Containing multicharacter (small, capital, numbers, special characters).
2. Humans must remember their passwords.
3. Unique per account.

Why Passwords?

"some thing you know" are more popular than "some thing you have" and "some thing you are" due to:

1. Cost: Passwords are free.

2. Convenience: easier for overworked system administrator to reset a password than to provide a new smartcard or issue a user a new thumb.

Keys Versus Passwords

	Keys	Passwords
1	For Key with a 64-bit = 2^{64} possible Keys, so the eavesdropper must on average try 2^{63} keys (without shortcut attacks).	Password of 8 characters with 256 possibility=$256^8 = (2^8)^8 = 2^{64}$, Trudy must on average try 2^{63} Passwords (without Dictionary attack).
2	Choose key at random.	Users do not select passwords at random.
3	Difficult to remember the key.	Easy to remember the Password.

Bad Passwords

Everyone would probably agree that the following passwords are weak:

✗ azad	→	Name of the user/ child/ etc.
✗ 1976321	→	Birthday: 21/3/1976
✗ Erbil	→	Cities Name.
✗ Password	→	Regular words

Good Passwords

are the following passwords better than the weak passwords above?

- E@dlb(!Nz86 → Bad for remembering.
- 02314982176145 → Bad for remembering.
- FtFs,Wsd2e. → Passphrase?

For the first semester, we should do two exams.

Attacking Systems via Passwords

A common attack path for Attackers (outsider) would be:

- Attackers will initially seek access to any account on the system and then attempt to upgrade her level of privilege.
- In this scenario, one weak password on an entire network could be enough for the first stage of the attack to succeed.

Attacking Systems via Passwords

Another interesting issue concerns the proper response when attempted password cracking is detected.

- For example, systems often lock users out after three bad passwords attempts.
- If this is the case, how long should the system lock? Five seconds? Five minutes? Until the administrator manually resets the service?

Attacking Systems via Passwords

- Five seconds might be insufficient to deter an automated attack.

If it takes more than five seconds for attacker to make three password guesses for every user on the system, then:

Attacker could simply cycle through all accounts, making three guesses on each. By the time she returns to a particular user's account, more than five second will have elapsed and she will be able to make three more guesses without any delay.

Attacking Systems via Passwords

On the other hand, five minutes might open the door to a denial of service attack, where Attacker is able to lock accounts indefinitely by periodically making three password guesses on an account.

The correct answer to this dilemma is not readily apparent.

Attacking Systems via Passwords

So, Attacker could:

Target one particular account.

Target any account on system.

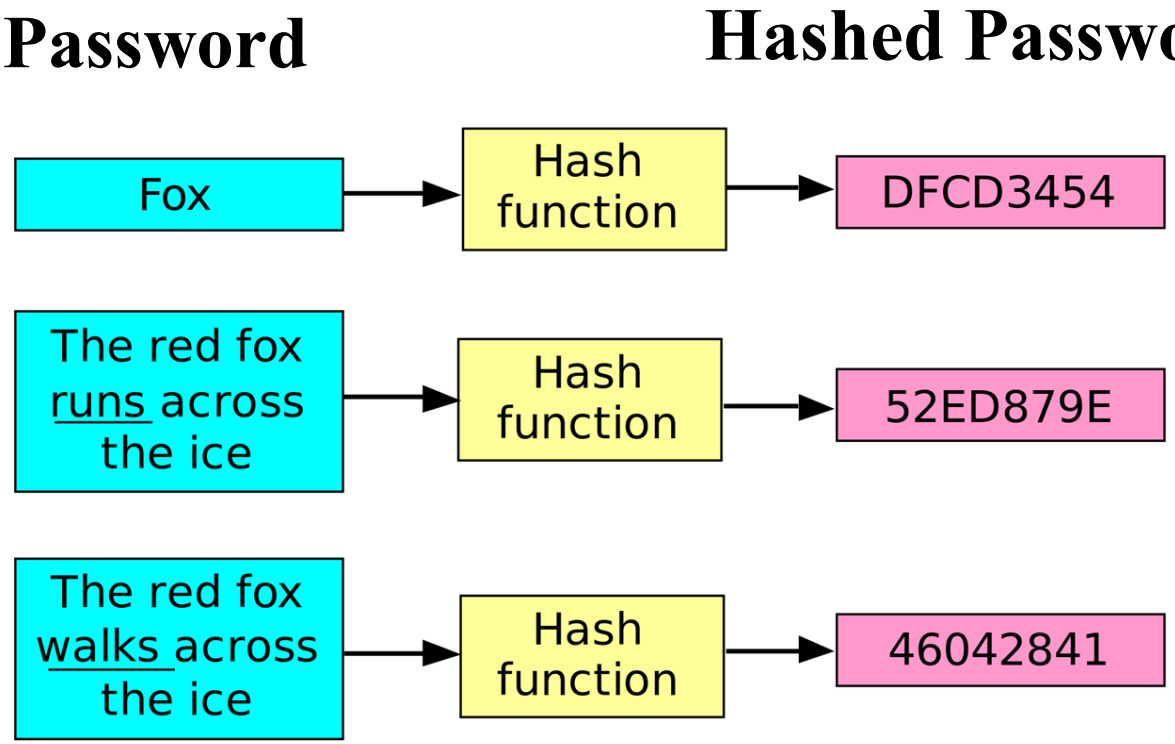
Target any account on any system.

Attempt denial of service (DoS) attack.

Password File?

- Bad idea to store passwords in a file.
- But we need to verify passwords.
- Solution? Hash passwords.
  - Store y = h(password).
  - Can verify entered password by hashing.
  - If Trudy obtains the password file, she does not (directly) obtain passwords.

Password File- With Hash Function

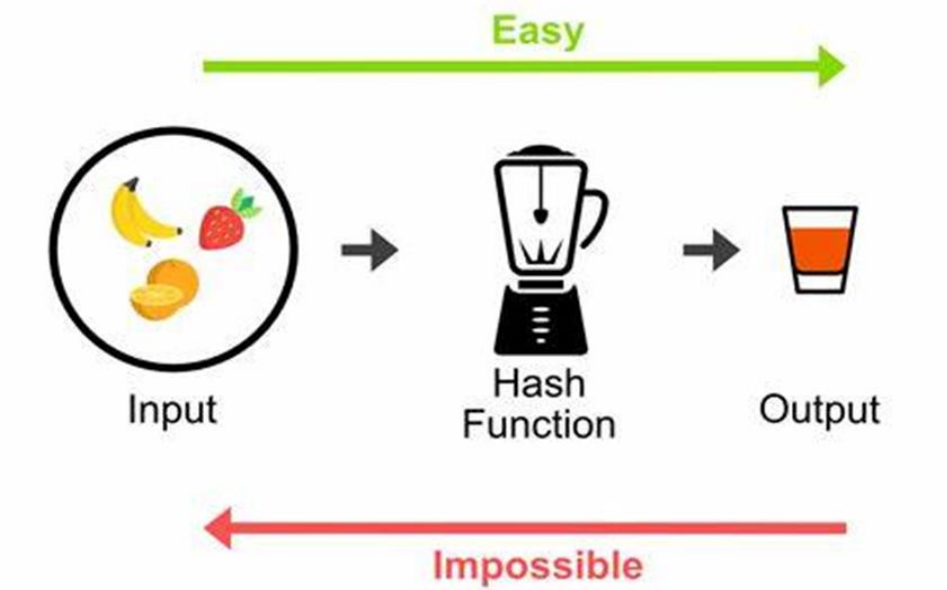


- But Attacker can try a forward search Attack:

  Guess x and check whether y = h(x)

Password File- With Hash Function

- Passwords are hashed, they are not encrypted.
- When something is encrypted, it can be decrypted.
- Hashes cannot be decrypted.

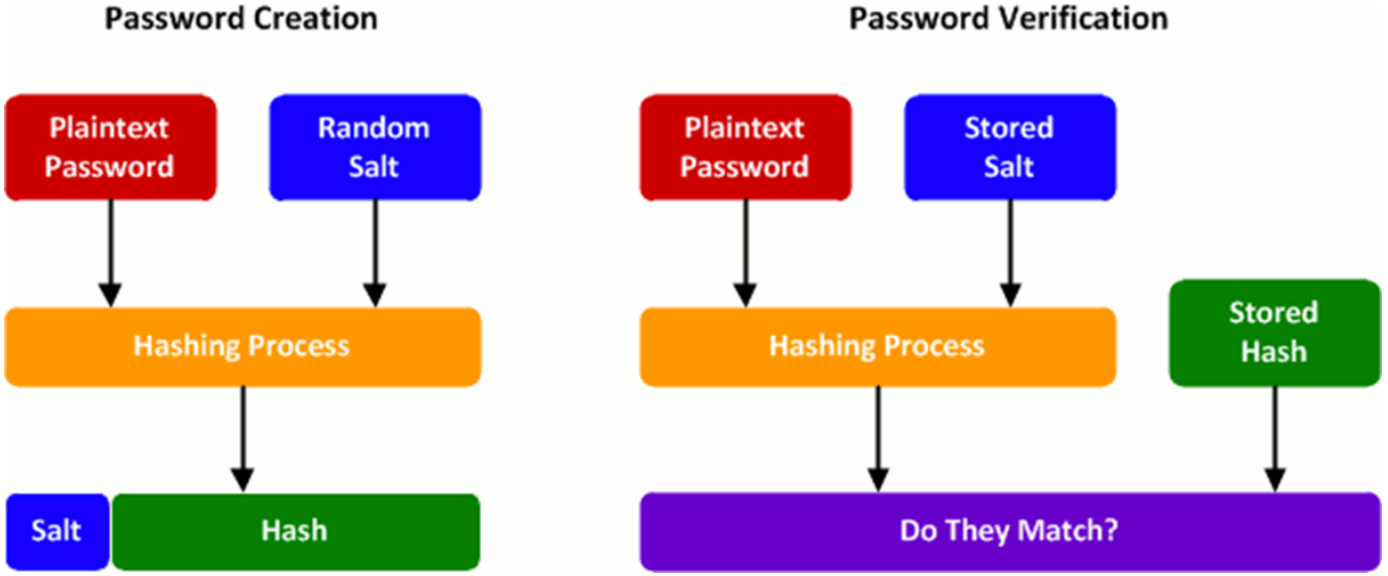


Dictionary Attack

- Attacker pre-computes h(x) for all x in a dictionary of common passwords.
- Suppose Attacker gets access to password file containing hashed passwords:
  - Attacker only needs to compare hashes to her pre-computed dictionary.
  - After one-time work of computing hashes in dictionary, actual attack is trivial.
- Can we prevent this forward search attack? Or at least make it more difficult?

Password File- Salt

- Salt is a non-secret value that causes identical password to hash into different values.
- It may prevent forward search attack, or at least make it more difficult.



Password Cracking: Do the Math

Case Study:

- Passwords are created with 8 characters, and there are 128 choices for each character.
  - Then $128^8 = 2^{56}$ possible passwords.
- There is a **password file** with 2^{10} passwords.
- Attacker has **dictionary** of 2^{20} common passwords.
- **Probability** 1/4 that password is in dictionary.
- **Work** is measured by number of hashes.

Password Cracking: Case I

- Attack 1 specific password without using a dictionary.
  - e.g., administrator's password.
  - Must try $2^{56}/2 = 2^{55}$ on average.
  - Like exhaustive key search.
- The result here is the same whether the passwords are salted or not, unless someone has precomputed, sorted, and stored the hashes of all possible passwords.

Password Cracking: Case II

- Attack 1 specific password using a **dictionary** with **salt**.
  - Expected work: $1/4\ (2^{19}) + 3/4\ (2^{55}) \approx 2^{54.6}$
  - In practice, try all passwords in dictionary…
  - …then work is at most 2^{20} and probability of success is 1/4.
- Attack 1 specific password using a **dictionary** with **No salt**.
  - One-time work to compute dictionary: 2^{20}
  - Expected work is of same order as above.
  - But with precomputed dictionary hashes, the "in practice" attack is essentially free…

Password Cracking: Case III

- Any of 1024 passwords in file, without **dictionary** with **salt**.
  - Each comparison requires a hash computation.
  - Need 255 comparisons before expect to find Passwords.
- Any of 1024 passwords in file, without **dictionary** with **No salt**.
  - Each computed hash yields 2^{10} comparisons.
  - So expected work (hashes) is $2^{55}/2^{10} = 2^{45}$

Example

- Consider Case I from Slide 22.

a. If the passwords are unsalted, how much work is it for Trudy to precompute all possible hash values?

b. If each password is salted with a 16-bit value, how much work is it for Trudy to precompute all possible hash values?

c. If each password is salted with a 64-bit value, how much work is it for Trudy to precompute all possible hash values?

- **Solution:?**

HomeWorks#2

1) Suppose all passwords on a given system are 8 characters and that each character can be any one of 64 different values. Attacker also has a dictionary of 2^{30} common passwords and the probability that any given password is in her dictionary is 1/4. The password file on this system contains 256 password hashes.
   a. How many different passwords are possible?
   b. What is the expected work for attacker to crack the administrator's password with dictionary and salt?
   c. What is the probability that at least one of the 256 passwords in the password file is in the dictionary?
   d. What is the expected work for Trudy to recover any one of the passwords in the password file with dictionary and salt?

HomeWorks#2

2) Give two passwords derived from the passphrase "Gentlemen do not read other gentlemen's mail."

3) This problem deals with storing passwords in a file.

a) Why is it a good idea to hash passwords that are stored in a file?

b) Why is it a much better idea to hash passwords stored in a file than to encrypt the password file?

c) What is a salt and why should a salt be used whenever passwords are hashed?