Chapter 2

Password Based Authentication

AUTHENTICATION MECHANISMS

- Password-based authentication

- Token-based authentication

- Biometric-based authentication

Password

- Widely used user authentication mechanism

- User provide username and password

- The system compares the data provided by the user with the data stored in the database

The user Id provides security in the following ways:

-The ID number determines whether this user is allowed to enter the system or not

-The ID number identifies the user's privileges within the system

The ID number helps control access. For example, a person owns a file in the system, allowing this person to determine the ID numbers who can access this file.

PASSWORD-BASED ATTACKS

- Guessing
- Social Engineering
- Dictionary Attacks
- Password Sniffing

GUESSING:

- Guessing is the easiest method to acquire a password illegally. The attacker may get lucky If the user uses a **short password** or If he forgets to change the **default password** of an account

GUESSING:

- Guessing is the easiest method to acquire a password illegally. The attacker may get lucky If the user uses a short password or If he forgets to change the default password of an account

default password:

These are the passwords that are set by the company. For example, router companies set initial passwords and these passwords are known, so the user must change the password when purchasing a new router.

SOCIALENGINEERING

- Social engineering is a method of using social skills to steal secret information from the victims.

For example, attackers may try to impersonate people with authority or to trick users to reveal sensitive information.

PHISHING

Phishing attacks are mass social engineering attacks that take advantage of people with a tendency to trust others.

For example, attackers may try to impersonate people with authority or to trick users to reveal sensitive information.

PASSWORD SNIFFING (MITM)

Password sniffers are software programs, used to capture remote login information
Such as usernames and user passwords.

COUNTER MEASERSPASSWORD ATTACKS

- Stop unauthorized access to password file
- Automatic workstation logout
- Encrypted passwords
- Hashed passwords
- Encrypted network links

PASSWORD HASHING

The process of converting password from a plaintext format into unreadable format through deploying a hashing algorithm such as **(MD5 and SHA-256)**

HASHING ALGORITHMS ONE – WAY PROGRAMS

HASH FUNCTION

A hash function that convert a plaintext to ciphertext (unreadable)

X: plaintext

h : hash function

h(x) = hash value – Digest value.

PRE-IMAGE RESISTANCE

given the digest(output) , attacker cannot find the input string.

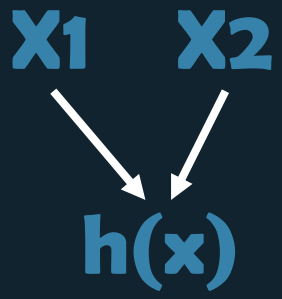
SECOND PRE-IMAGE RESISTANCE

X: plaintext

h : hash function

h(x) = hash value – Digest value.

Collision : two different have same digest



given one specific input string, attacker cannot find another input string with same digest. (weak collision resistance).

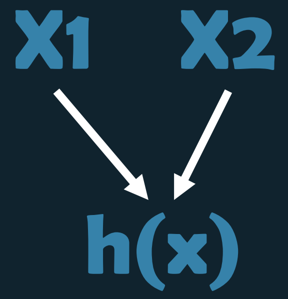
COLLISION RESISTANCE

X: plaintext

h : hash function

h(x) = hash value – Digest value.

Collision : two different have same digest



X1 X2

h(x)

cannot find any two input strings that produce the same digest.(strong collision resistance).

EXAMPLE OF COLLISION :

If there is a hash function that maps a variable bit-length input string (x) to an output (digest) of 20 bits. Let’s say x equals 1000 bit How many collisions are in this example?

Answer :

possible input : 2^{1000}

possible output : 2^{20}

Collisions = $2^{\text{input size}} / 2^{\text{output size}} = 2^{1000} / 2^{20} = 2^{980}$

BRUTE FORCE ATTACKS ON HASH FUNCTIONS

Pre-image and Second pre-image attack:

Find x that gives a specific h(x); try all possible values of X. With n-bit hash function, effort required (tries) to defeat such algorithm is 2^n تجربة كل الاحتمالات

Collision resistance attack:

Find any two input strings that have the same hash values. With n-bit hash function, effort required (tries) to defeat such algorithm is $2^{n/2}$ تجربة نصف الاحتمالات

MESSAGEAUTHENTICATION CODE (MAC)

- **Message Authentication Code (MAC)**
- Takes message and a secret key as input and returns unique and random-looking output

Different inputs (key and/or data) will produce different outputs

Output called: tag (t)

$t = \mathrm{MAC}(K,M)$

HASHINGALGORITHM

Characteristics

Mathematical: Strict rules that manage the work of an algorithm, and those rules nearly can’t be broken or adjusted.

Uniform: Implementing a specific hashing algorithm, and data of any character length will generate predetermined length output.

Consistent: The algorithm does just one thing (compress data) and nothing else.

One way: Once transformed by the algorithm, it’s nearly impossible to revert the data to its original state.

HASHINGALGORITHM

MD5 : OUTPUT 128 bit

SHA-256 : OUTPUT 256 bit

SHA-512 : OUTPUT 256 bit

HASHING ALGORITHM MECHANISM

1- Create the message

2- choose the type of hashing algorithms

3- Enter the message

4- start the hash

5- Store

HASHING ALGORITHM APPLICATIONS

1- Password storage

2- Digital signatures

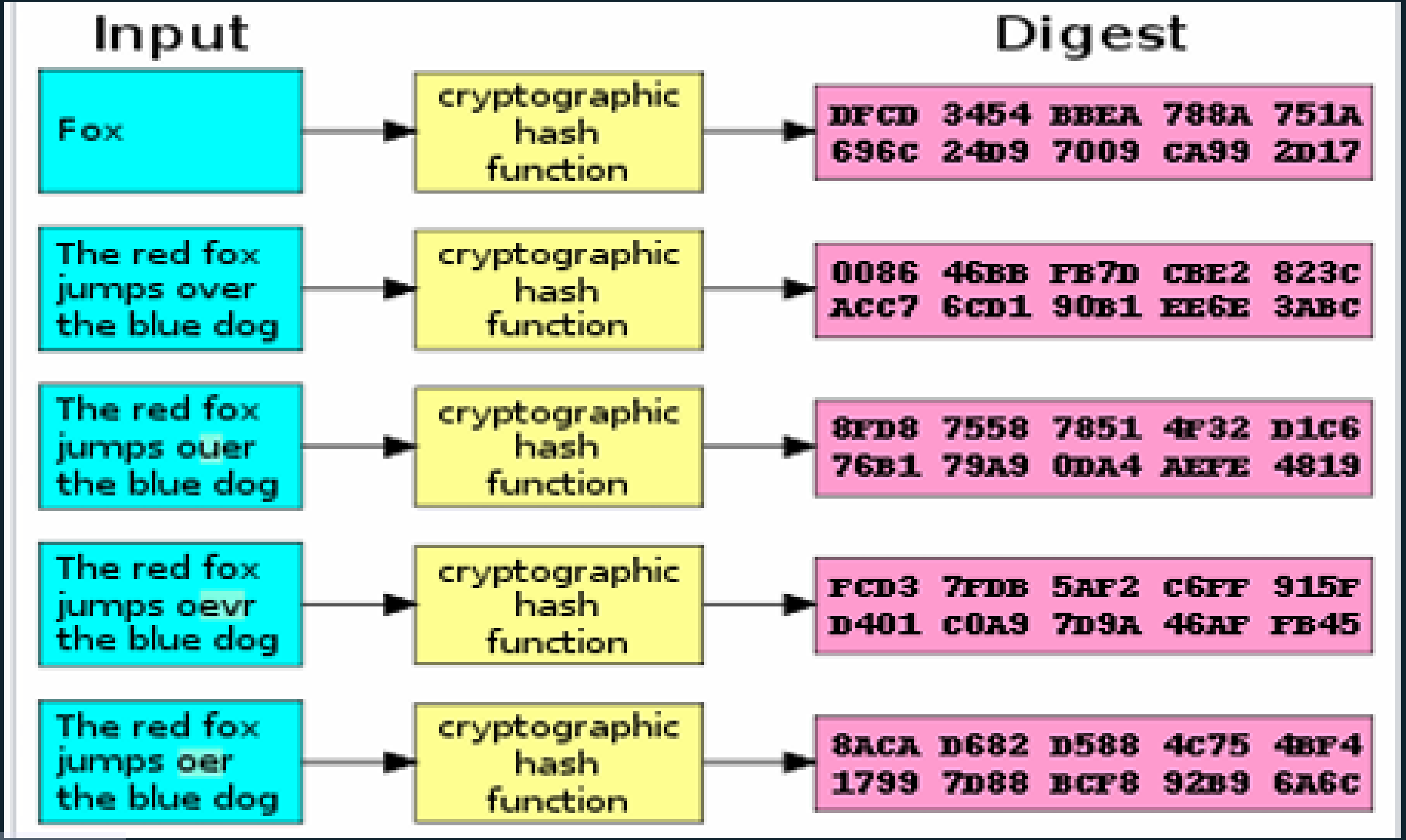
3- Enter the message

4- Document management

5- File management

HASHING

A small change in input leads to large changes in output



Input

Digest

Fox → cryptographic hash function → DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17

The red fox jumps over the blue dog → cryptographic hash function → 0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC

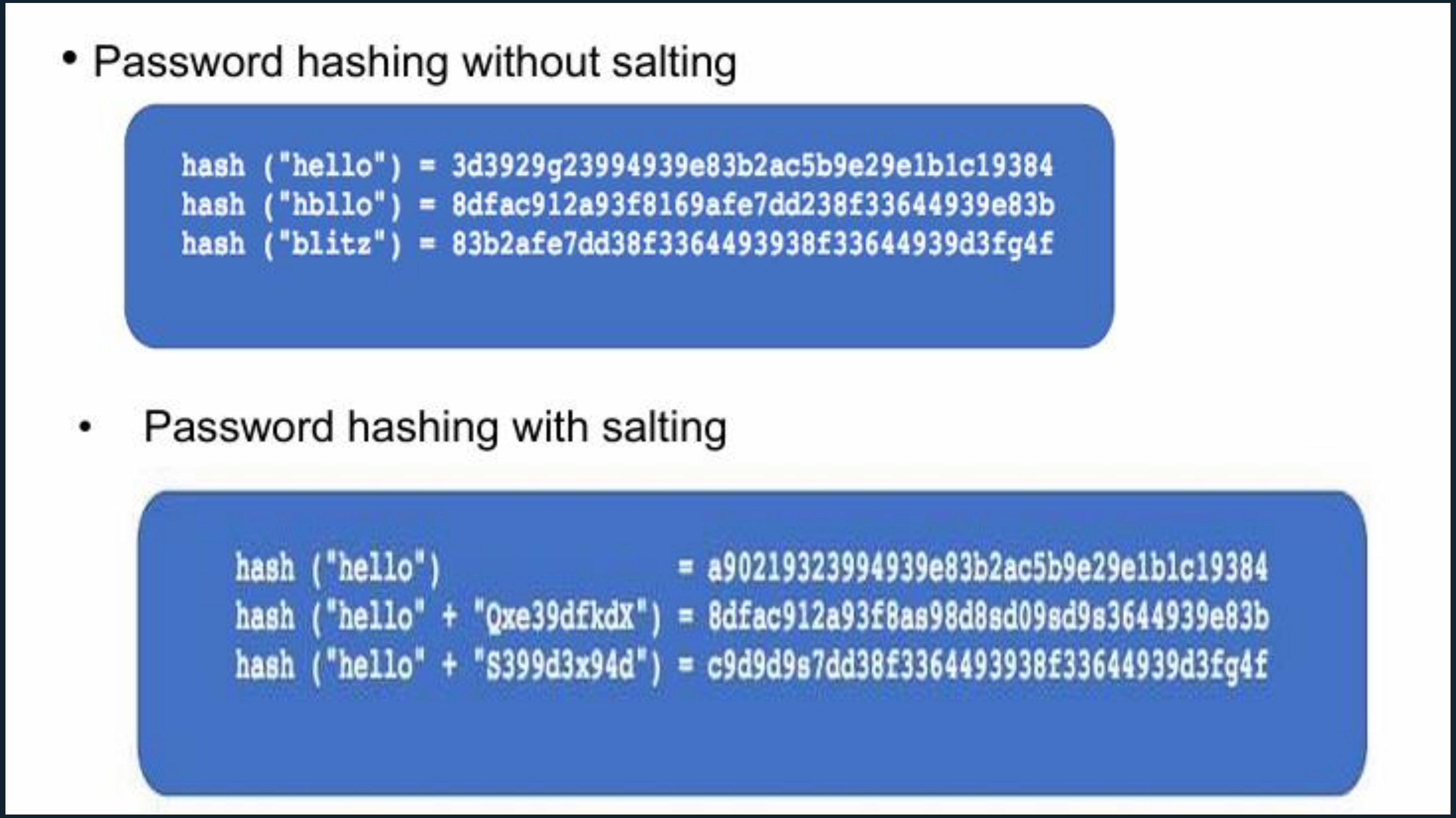
The red fox jumps ouer the blue dog → cryptographic hash function → 8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819

The red fox jumps oevr the blue dog → cryptographic hash function → FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45

The red fox jumps oer the blue dog → cryptographic hash function → 8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

PASSWORD SALTING

Salt values provide higher level of randomness to the hashed passwords, which leads to different digests each time



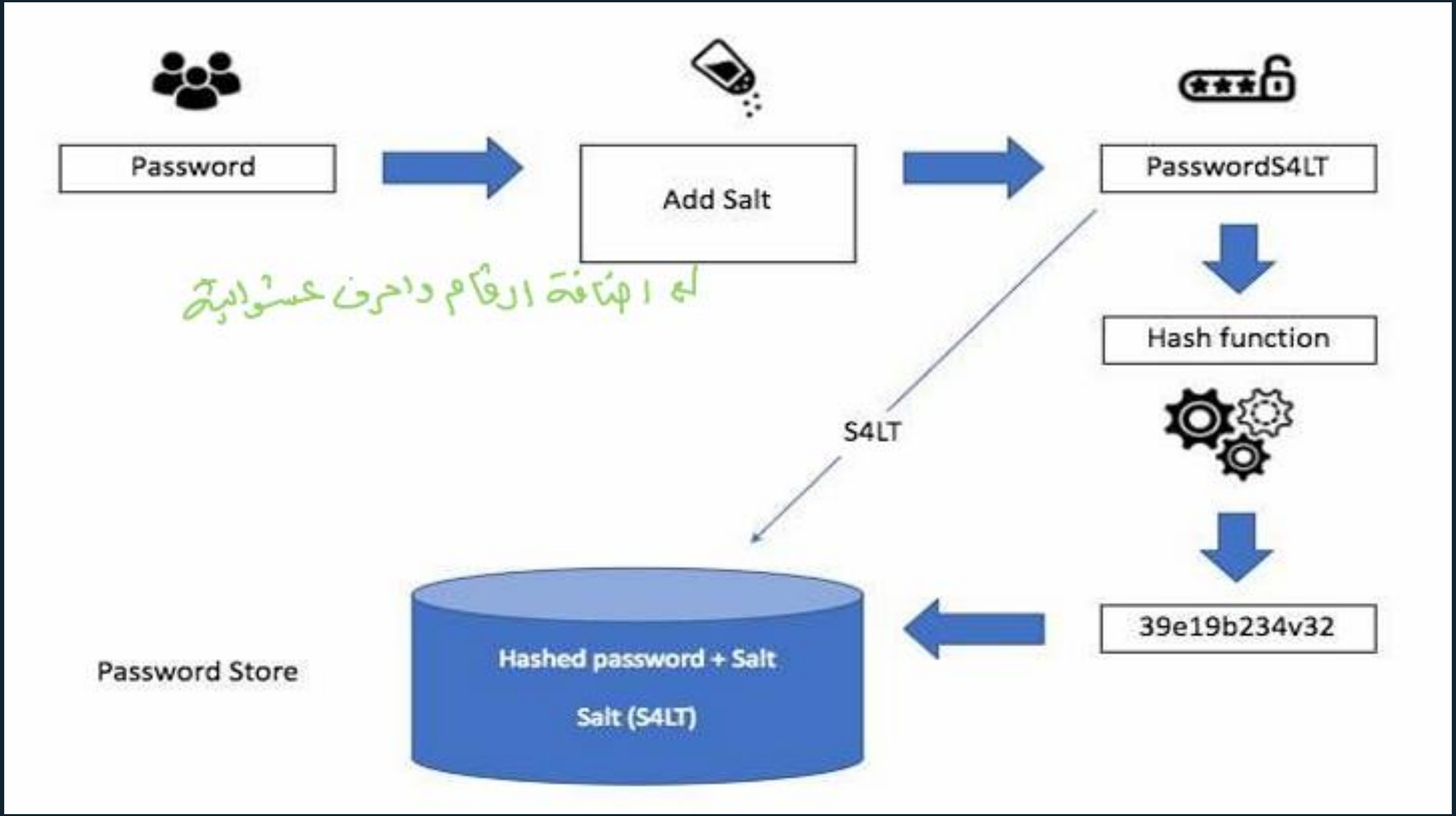
- Password hashing without salting

```
hash ("hello") = 3d3929g23994939e83b2ac5b9e29e1b1c19384
hash ("hbllo") = 8dfac912a93f8169afe7dd238f33644939e83b
hash ("blitz") = 83b2afe7dd38f3364493938f33644939d3fg4f
```

- Password hashing with salting

```
hash ("hello")                = a90219323994939e83b2ac5b9e29e1b1c19384
hash ("hello" + "Qxe39dfkdX") = 8dfac912a93f8as98d8sd09sd9s3644939e83b
hash ("hello" + "S399d3x94d") = c9d9d9s7dd38f3364493938f33644939d3fg4f
```

PASSWORD SALTING



لي اضافة ارقام واحرف عشوائية