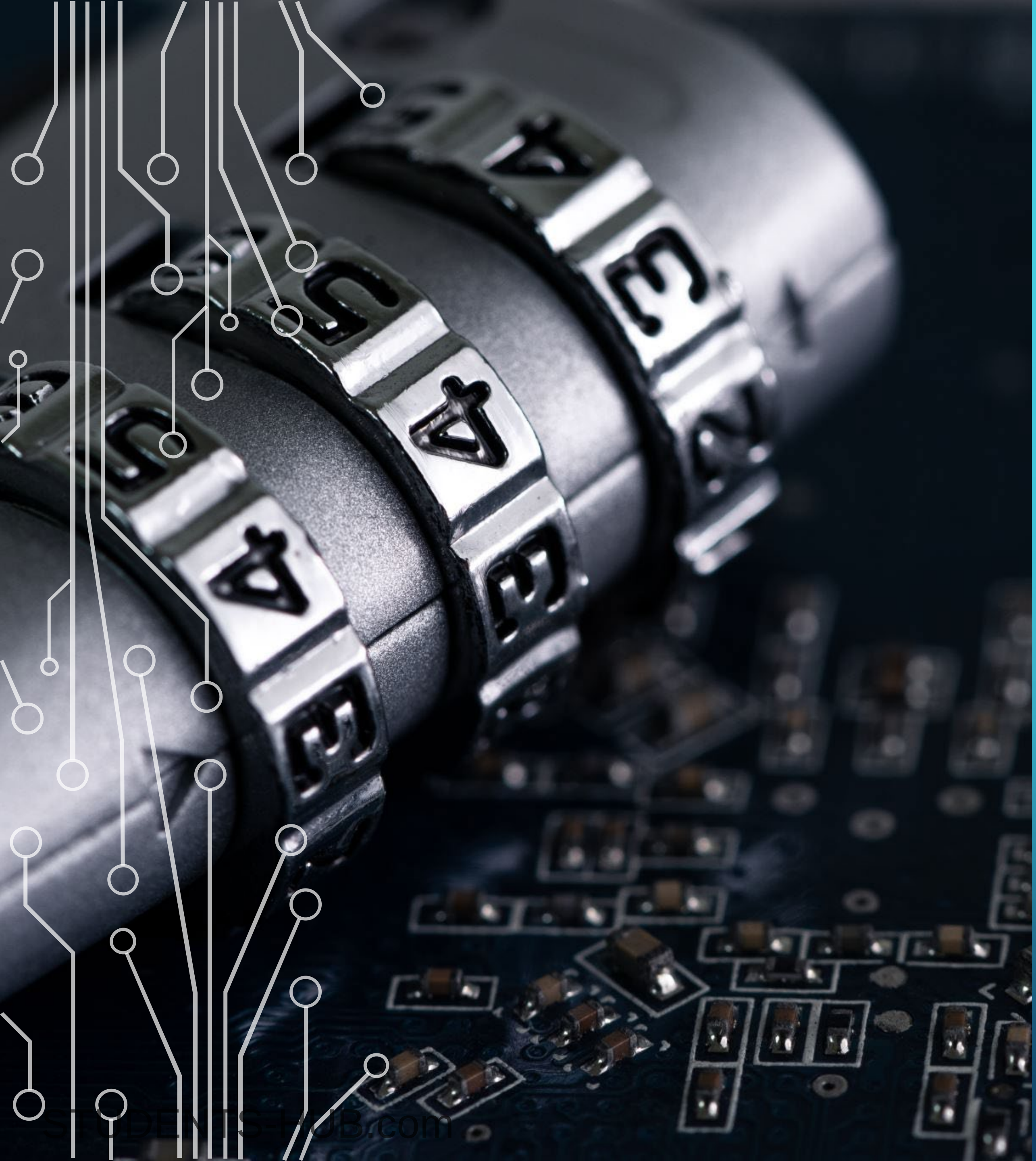


PASSWORD-BASED AUTHENTICATION

DR. ASEM KITANA

AUTHENTICATION MECHANISMS

- Password-based authentication
- Token-based authentication
- Biometric-based authentication

PASSWORDS

- ❑Widely used user authentication mechanism
    - ❑User provides name/identifier (ID) and password.
    - ❑The system compares the password to a previously stored password for that user ID, maintained in a system password file.

PASSWORDS

❖The user ID provides security in the following ways:

- ➢The ID determines whether the user is authorized to gain access to a system. In some systems, only those who already have an ID filed on the system are allowed to gain access.

- ➢The ID determines the privileges accorded to the user. A few users may have administrator or “superuser” status that enables them to perform special protected functions. Some systems have guest or anonymous accounts, and users of these accounts have limited privileges .

- ➢The ID is used in what is referred to as discretionary access control. For example, by listing the IDs of the other users, a user may grant permission to them to read files owned by that user.

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.
- ❖According to a recent survey by PC Magazine, the ten most common passwords used by users, are as follows:

1. password

2. 123456

3. qwerty (which are keys below 123456 on standard keyboards)

4. abc123

5 letmein

6. monkey

7. myspace1

8. password1

9. blink182

10. the user's own first name

SOCIAL ENGINEERING

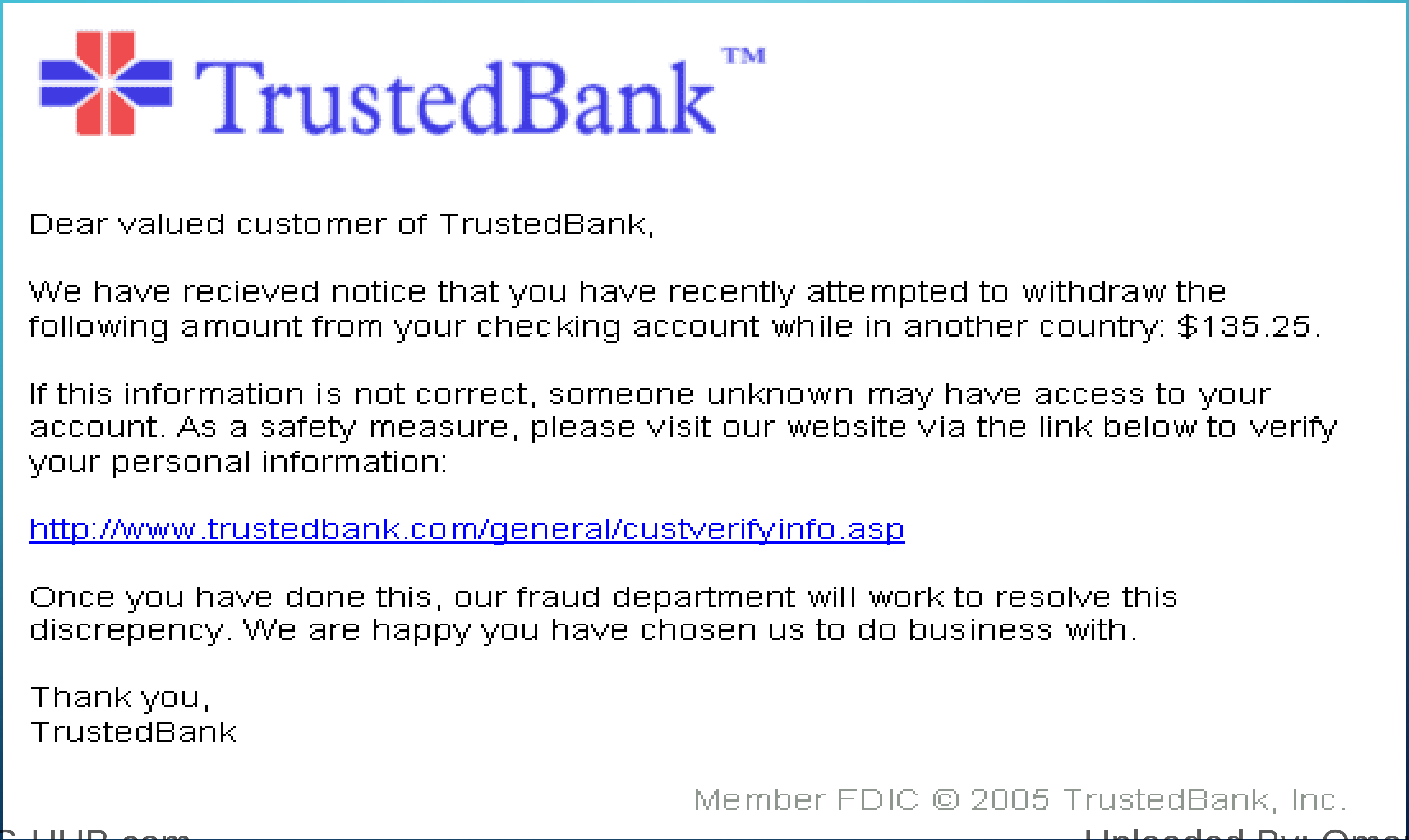
- Social engineering is a method of using social skills to steal secret information from the victims.
- For example, attackers may try to impersonate people with authority or to trick users to reveal sensitive information.
- Impersonation may be carried out either in person or in an electronic form. Phishing is an electronic form of social engineering targeted at a large number of people.
- There are other forms of social engineering attacks. For example, attackers may try to collect recycled papers from the recycle bins in a corporation's office building.

PHISHING

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

- It is type of manipulation to deceive people by performing actions for the purpose of information gathering, fraud, or system access.

PHISHING



TrustedBank™

Dear valued customer of TrustedBank,

We have recieved notice that you have recently attempted to withdraw the following amount from your checking account while in another country: $135.25.

If this information is not correct, someone unknown may have access to your account. As a safety measure, please visit our website via the link below to verify your personal information:

http://www.trustedbank.com/general/custverifyinfo.asp

Once you have done this, our fraud department will work to resolve this discrepency. We are happy you have chosen us to do business with.

Thank you,
TrustedBank

Member FDIC © 2005 TrustedBank, Inc.

PASSWORD SNIFFING

- Password sniffers are software programs, used to capture remote login information such as user names and user passwords.

- Common network applications such as Telnet, FTP, SMTP, and POP3

- Often require users to type their user names and passwords for authentication, making it possible for a password sniffer to intercept useful login information.

COUNTERMEASURES FOR PASSWORD ATTACKS

- Stop unauthorized access to password file
- Intrusion detection measures
- Training & enforcement of policies
- Automatic workstation logout
- Encrypted network links
- Encrypted passwords
- Hashed passwords

PASSWORD HASHING

- The process of converting a password from a plaintext format into unreadable format through deploying a hashing algorithm, such as (MD5 and SHA-256).
- A hashing algorithm is a mathematical function that garbles data and makes it unreadable.
- Hashing algorithms are one-way programs, so the decoding is infeasible (very hard). And that’s the point.
- Hashing protects data at rest, so even if someone gains access to your server, the items stored there remain unreadable.

HASH FUNCTION

- A hash function is a function *h* which maps an input *x* of arbitrary finite bit-length (i.e. variable length string), to an output *h(x)* of fixed bit-length *n*.
- x: password/message string, h: hash function, h(x): digest/hash value
- Three common required security properties of a hash function h with inputs x, x′ and outputs y, y′:

✓Pre-image resistance

✓Second pre-image resistance

✓Collision resistance

PRE-IMAGE RESISTANCE

- Pre-image resistance — for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any pre-image x' such that $h(x') = y$ when given any y for which a corresponding input is not known.
- In other words: given the digest, attacker cannot find the input string.
- a.k.a. one-way property

SECOND PRE-IMAGE RESISTANCE

- Second pre-image resistance — it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a second pre-image $x' \neq x$ such that $h(x) = h(x')$.
- $\boxed{h(x) = h(x')} \longrightarrow$ Collision
- In other words: given one specific input string, attacker cannot find another input string with same digest.
- a.k.a. weak collision resistance.

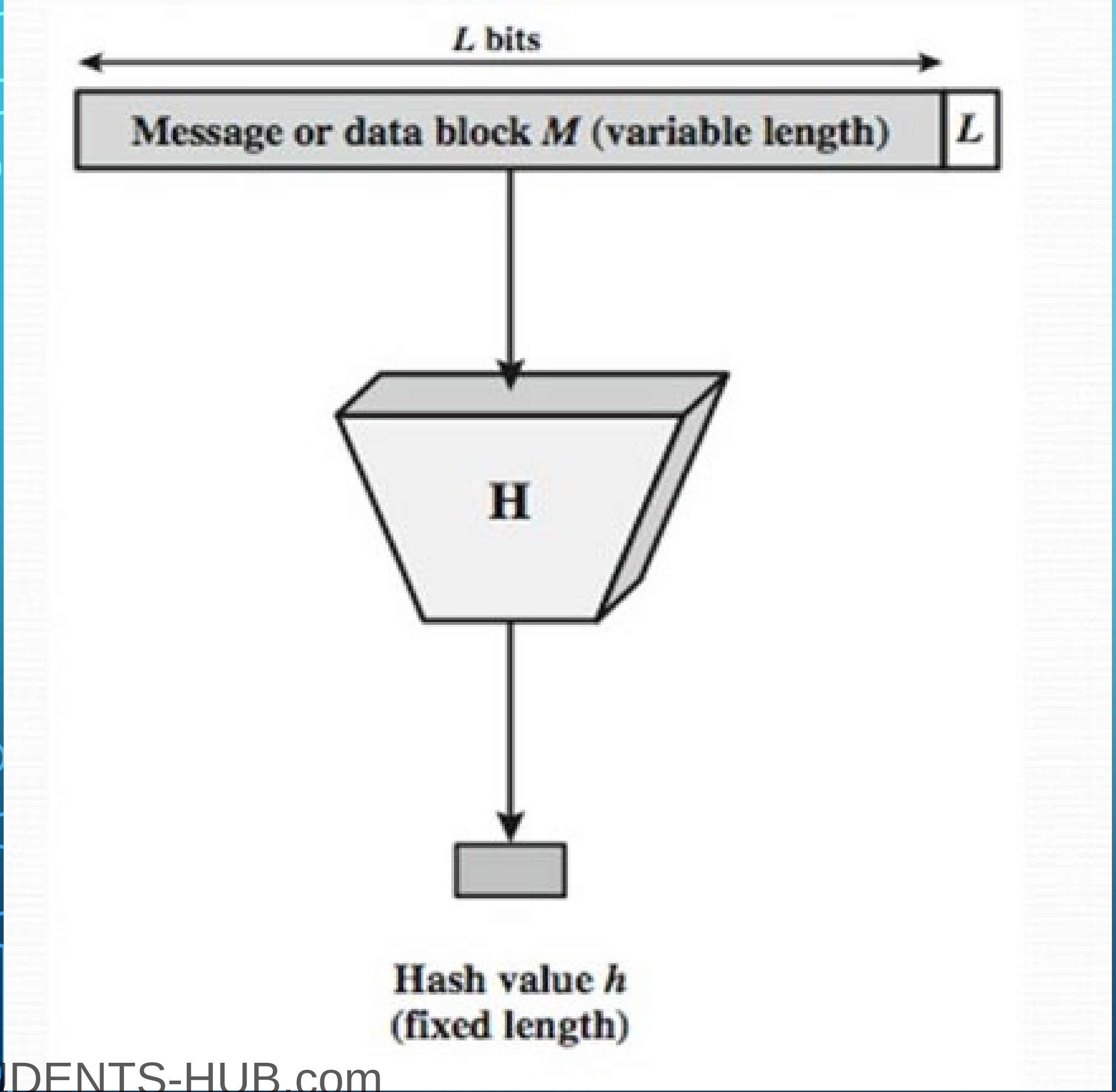
COLLISION RESISTANCE

- Collision resistance — it is computationally infeasible to find any two distinct inputs
  x, x' which hash to the same output, i.e., such that $h(x) = h(x')$. (Note that here there is free choice of both inputs).
- In other words: attacker cannot find any two input strings that produce the same digest.
- a.k.a. strong collision resistance.

HASH FUNCTIONS

- Hash functions
  - Takes message as input and returns unique and random-looking output
  - Different inputs will produce different outputs
  - Also called: Modification Detection Code (MDC), unkeyed hash function
  - Output called: hash value, digital fingerprint, and digest
  - A secure hash function should computationally infeasible to find:
    - x that maps to known h(x) (one-way property)
    - x and x′ that produce the same h(x) (collision-free property)

HASH FUNCTIONS

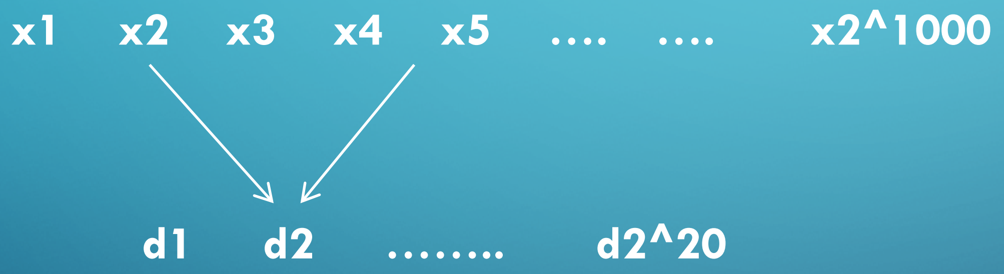


Input is much larger than output, which means repetitions/collisions are possible in theory.
(we accept the fact that two different input strings could have the same digest in theory, but in practice a secure hash algorithm should make it infeasible for hackers to find a collision).

EXAMPLE OF COLLISION

If there is a hash function that maps a variable bit-length input string (x) to an output (i.e. digest) of 20 bits. Let’s say x equals 1000 bits, then:

- There are 2^1000 possible input strings (x values), and
- There are 2^20 possible output values (d values, where d = h(x))



- ➢ There is a collision, which is represented by h(x2) = h(x5) = d2
- ➢ How many collisions are in this example? $\frac{2^{1000}}{2^{20}} = 2^{980}$

EXAMPLE OF COLLISION

- In the previous example, there are 2^{980} possible collisions on average that could happen.
- In other words: there are 2^{980} different input strings that will map to the same digest.
- Despite of the large number of collisions, but it will be hard for hackers to find two input strings that map to the same digest, why?
  - Many of the input strings don't make sense as password in the english language.
  - From hackers perspective, it is not only finding collision, but also finding collisions for input strings that make sense.
  - How many tries does it take for a hacker to guarantee reaching a string that has the same digest? The hacker should try all the possible hash values (2^{20} tries).
  - Therefore, increasing the size of a hash value will make the hacker task much harder.

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.
- **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 $2^{n/2}$.

MESSAGE AUTHENTICATION 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
  - Also called: keyed hash function
  - Output called: tag (t)
  - t = MAC(K,M)

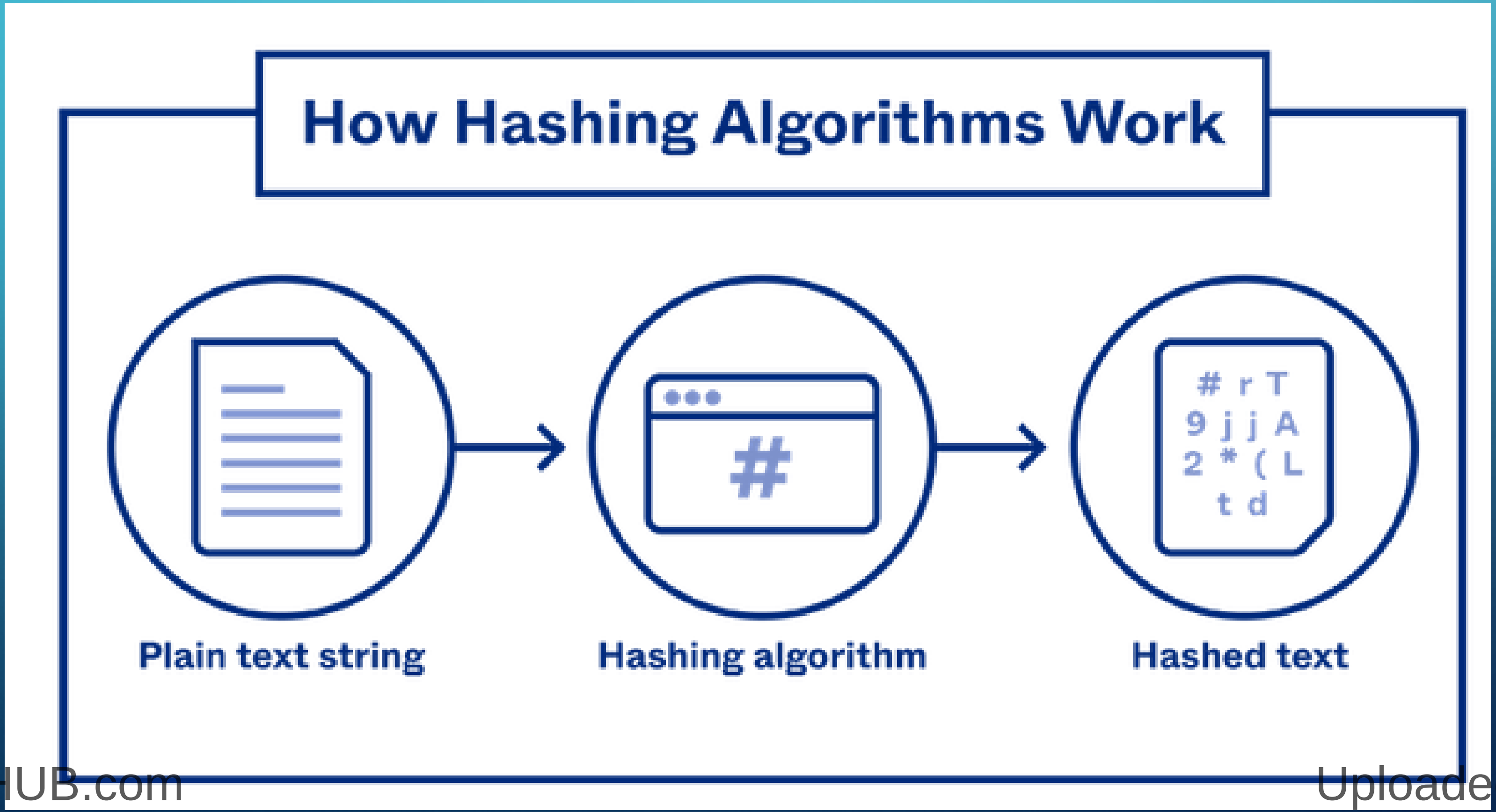
HASHING ALGORITHM

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.

HASHING ALGORITHM

- It’s important to understand that hashing and encryption are ***different*** functions.



HASHING ALGORITHM MECHANISM

- **Create the message:** A user determines what should be hashed.
- **Choose the type:** Dozens of hashing algorithms exist, and the user might decide which works best for this message.
- **Enter the message:** The user taps out the message into a computer running the algorithm.
- **Start the hash:** The system transforms the message, which might be of any length, to a predetermined bit size. Typically, programs break the message into a series of equal-sized blocks, and each one is compressed in sequence.
- **Store or share.** The user sends the hash (also called the "message digest") to the intended recipient, or the hashed data is saved in that form.

HASHING ALGORITHM APPLICATIONS

You might use a hashing algorithm for:

- ✓**Password storage:** You must keep records of passwords, so people can access your resources. Hashing ensures that the data is stored in a scrambled state, so it's harder to steal.
- ✓**Digital signatures:** A tiny bit of data proves that a note wasn't modified from the time it leaves a sender's outbox and reaches receiver inbox.
- ✓**Document management:** Hashing algorithms can be used as a mechanism for data integrity. The writer uses a hash to secure the document when it's complete. The hash works a bit like a seal of approval. A recipient can generate a hash and compare it to the original. If the two are equal, the data is considered genuine.
- ✓**File management:** Some companies also use hashes to index data, identify files, and delete duplicates. If a system has thousands of files, using hashes can save a significant amount of time.

HASHING ALGORITHM EXAMPLES

❑ Hashing the statement “At the top of an apartment building in Queens.“ looks like:

- ➢ **MD5:** 72b003ba1a806c3f94026568ad5c5933
- ➢ **SHA-256:** f6bf870a2a5bb6d26ddbeda8e903f3867f729785a36f89bfae896776777d50af

❑ While, Hashing the statement “Chicago.“ looks like:

- ➢ **MD5:** 9cfa1e69f507d007a516eb3e9f5074e2
- ➢ **SHA-256:** 0f5d983d203189bbffc5f686d01f6680bc6a83718a515fe42639347efc92478e

Notice that the original messages don't have the same number of characters. But the algorithms produce hashes of a consistent length each time. And notice that the hashes are completely scrambled.

HASHING ALGORITHM TOOLS

- MD5 Hash Generator:

https://www.md5hashgenerator.com/

- SHA-256 Hash Generator:

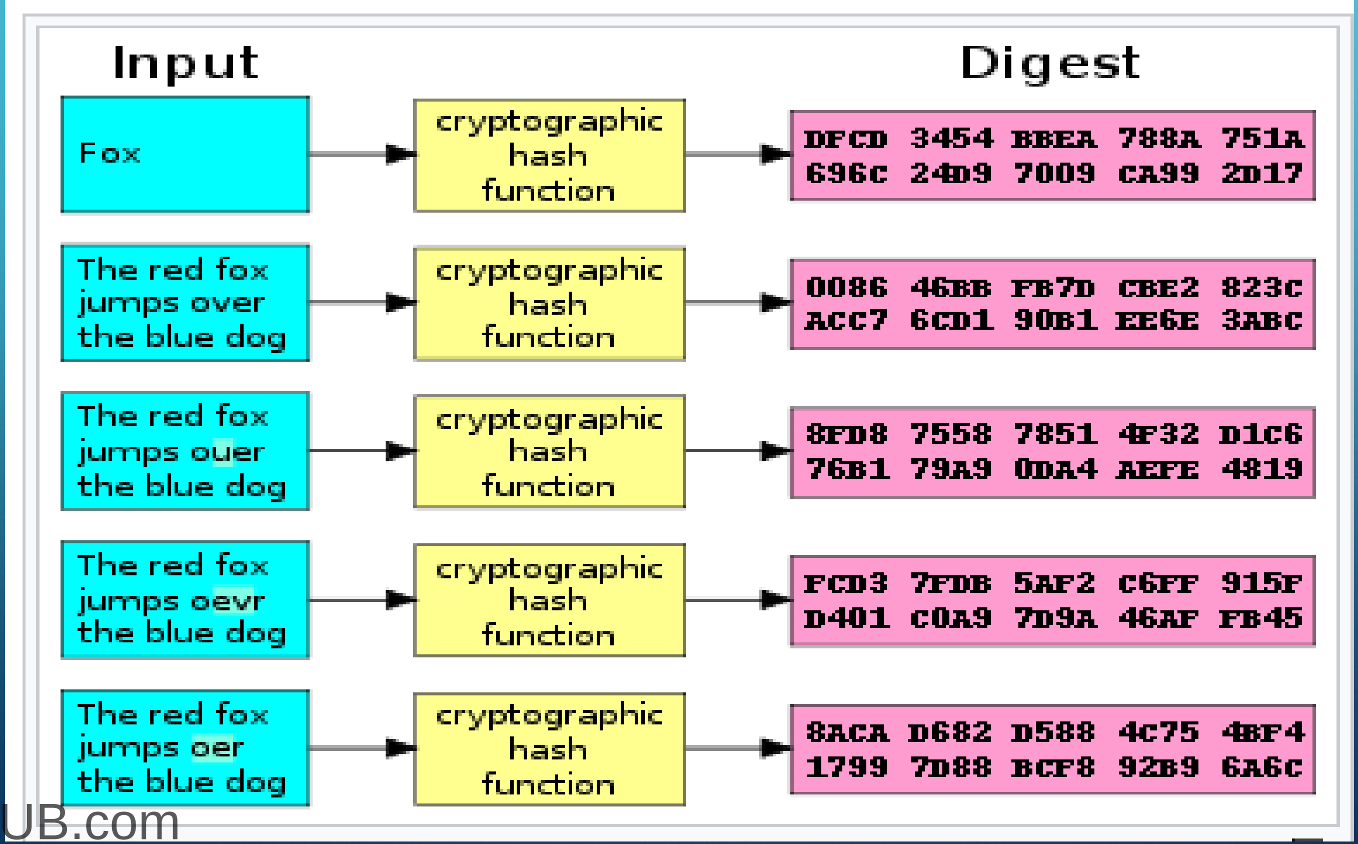
https://www.cleancss.com/sha256-hash-generator/

COMMON HASHING ALGORITHMS

- **MD5:** Message Digest 5 was designed in 1991 to replace an earlier hash algorithm, MD4. MD5 can be broken within seconds which makes the algorithm unsuitable for most use cases. MD5 produces a hash value (digest) of 128 bits.
- **RIPEMD-160:** RACE Integrity Primitives Evaluation Message Digest 160 was developed in 1992, which generates digests of 160 bits.
- **SHA-2:** Secure Hash Algorithm 2 was designed in 2001, which includes hash functions with digests that are 224 bits (**SHA-224**), 256 bits (**SHA-256**), 384 bits (**SHA-384**) or 512 bits (**SHA-512**).
- **Whirlpool:** developed in 2000, which is based on a modified version of the Advanced Encryption Standard (AES). Whirlpool produces a digest of 512 bits.

AVALANCHE EFFECT

- Represents a major property of hashing algorithms, which states that if an input is changed slightly (for example, flipping a single bit), then the output changes significantly (e.g., half the output bits flip).



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

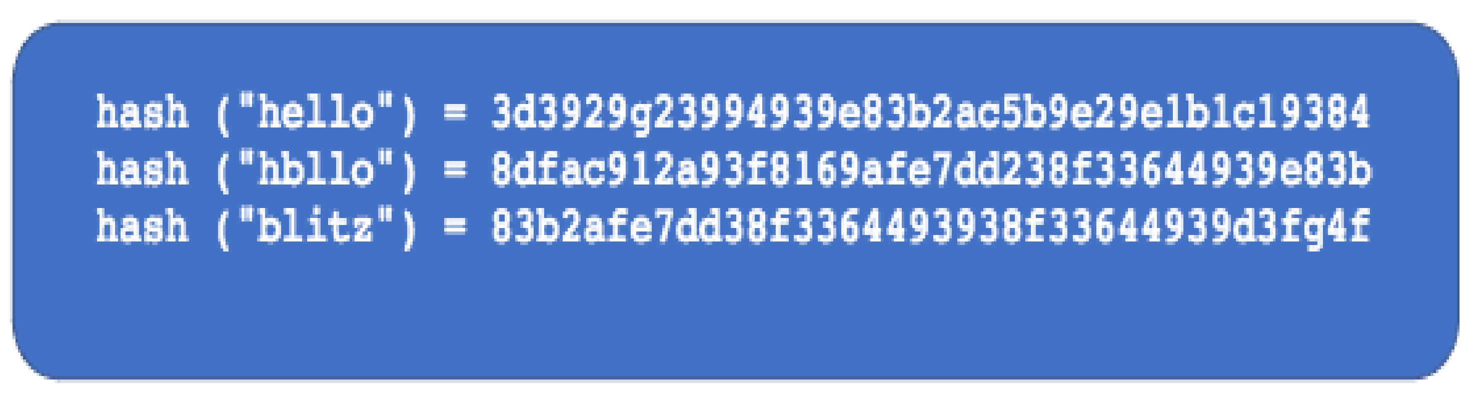
The red fox jumps oer the blue dog → cryptographic hash function → 8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

PASSWORD SALTING

- ❑Password hashing is a key step to protecting users passwords, but it’s not enough because it hashes in a consistent way. This means it is predictable and can be broken by dictionary attack, brute-force attack, or rainbow table attack.
- ❑“Hello”, for example, will always equal to the same combination of letters and numbers, and therefore can be guessed through brute force. One way of protecting against this is by adding password salt or using salted passwords.
- ❑**Salting** is the act of adding a series of random characters to a password before going through the hashing function.

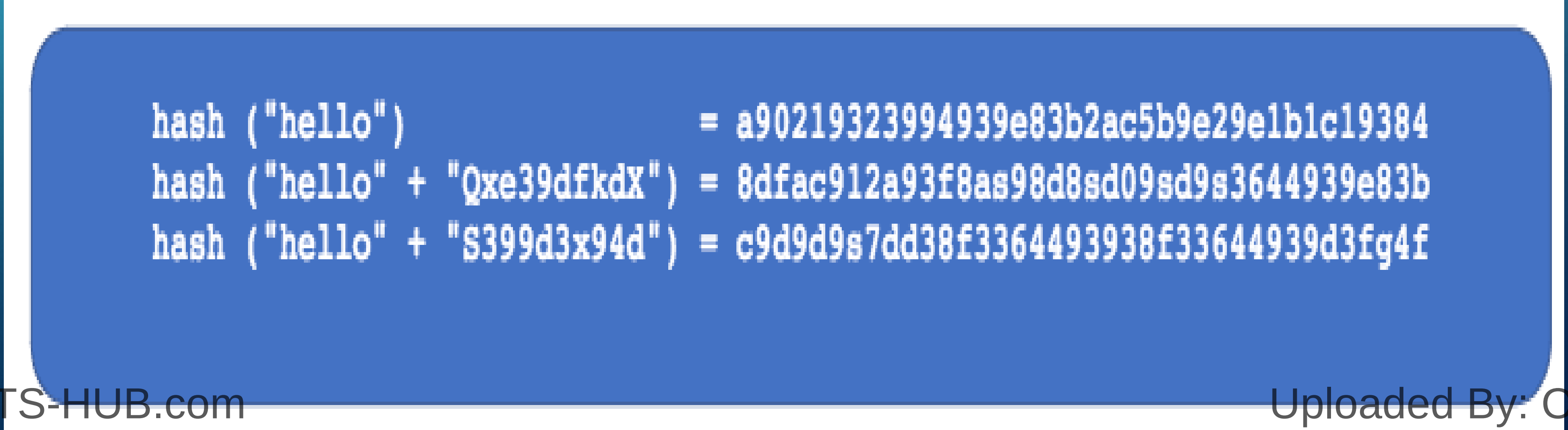
PASSWORD SALTING

- Password hashing without salting



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

- Password hashing with salting

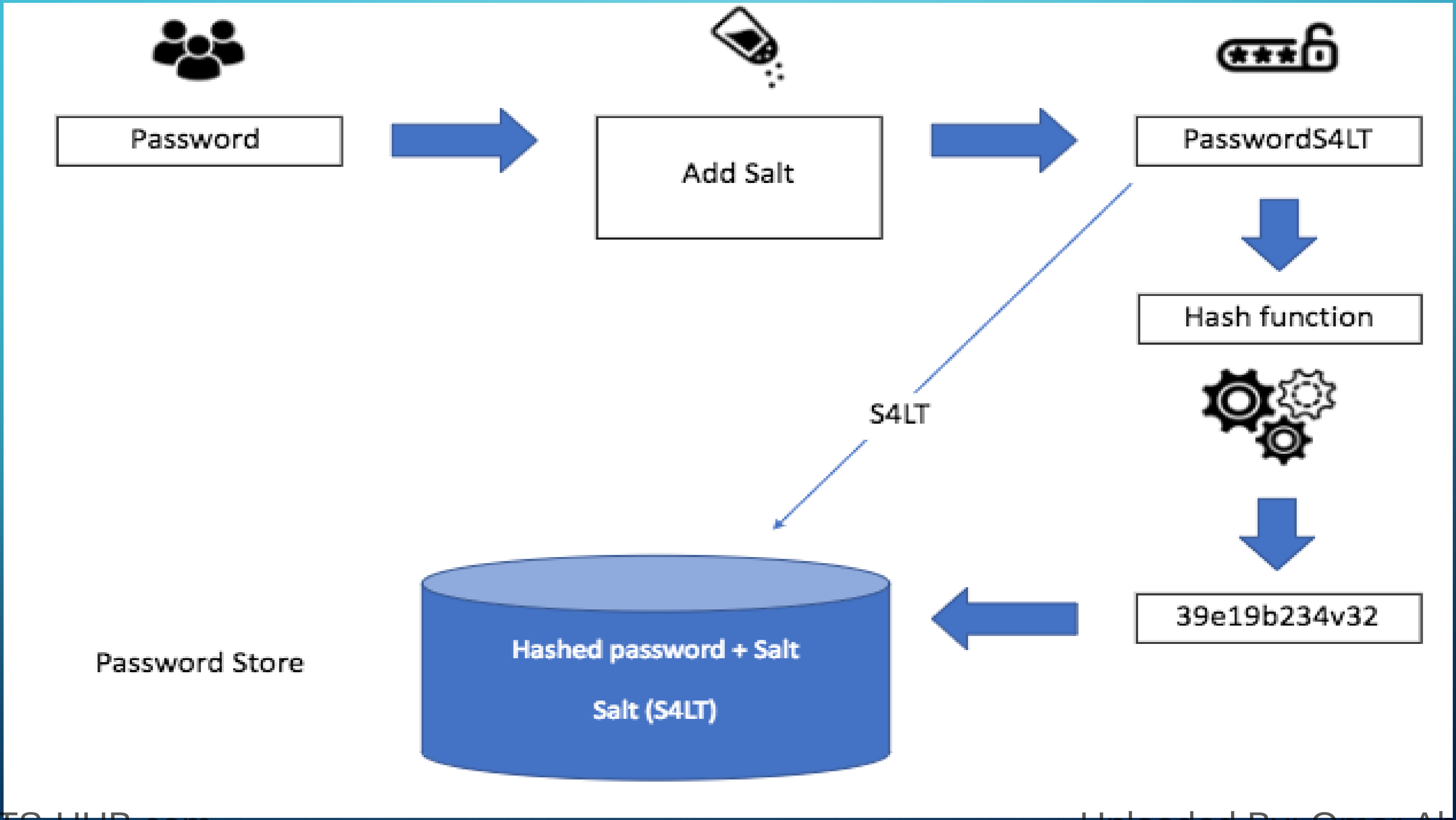


```
hash ("hello")                = a90219323994939e83b2ac5b9e29e1b1c19384
hash ("hello" + "Qxe39dfkdX") = 8dfac912a93f8as98d8sd09sd9s3644939e83b
hash ("hello" + "S399d3x94d") = c9d9d9s7dd38f3364493938f33644939d3fg4f
```

PASSWORD SALTING

- Salt values provide higher level of randomness to the hashed passwords, which leads to different digests each time.
- Therefore, enhancing the protection against different password attacks.

SALTED PASSWORD ARCHITECTURE



Password

Add Salt

PasswordS4LT

Hash function

39e19b234v32

S4LT

Password Store

Hashed password + Salt

Salt (S4LT)