

Encoding, Encryption & Hashing

ahmedbda

- 01** Context
- 02** Encoding
- 03** Encryption
- 04** Hashing
- 05** Comparison
- 06** Real world use cases

Context

01

Security relies on selecting the correct mechanism for the specific data goal

Encoding, encryption and hashing may sound similar to some, but they are very distinct mechanisms and confusing them with each other can cause a lot of security problems

This resource aims to simply differentiate the three concepts, detailing their attributes and proper use cases.

Encoding

02

The purpose of encoding is usability, not security

Encoding means transforming data into a specific format for safe transport across systems

Let's take a look at its key traits,

It requires no key, anyone can reverse it	It uses public algorithms like Base64, ASCII, URL Encoding	It is reversible and designed to be decoded easily
---	--	--

It's like morse code, the secret is not hidden, just formatted differently

`b64encode(b"Hello")`  `b"SGVsbG8="`

Encryption

03

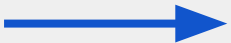
The purpose of encryption is confidentiality

Encryption means obfuscating data using mathematical algorithms so it is unreadable without a secret

Let's take a look at its key traits,

It requires a key and keeps the key secret, not the algorithm	It uses mathematical algorithms like AES or RSA	It is reversible, but only if you possess the correct key
---	---	---

It's like a steel safe, the content exists but is inaccessible without the combination

`fernet.encrypt(b"Hello")`  `b"gAAAAABk9..."`

Hashing

04

The purpose of hashing is integrity

Hashing means mapping data of arbitrary size to a fixed-size string (that is called the "digest")

Let's take a look at its key traits,

It is a one-way function that is mathematically irreversible	It uses algorithms like SHA-256 or MD5	The same input always results in the exact same output
--	--	--

It's like a fingerprint, you can identify a person by it but you cannot recreate them

`sha256(b"Hello").hexdigest()`  `'185f8db32271fe25f561a6fc938b2e264...'`

Comparison

05

Now a comparison table for technical overview to quickly distinguish the different mechanisms based on four attributes,

Method	Encoding	Encryption	Hashing
Goal	Usability (data format)	Confidentiality (secrecy)	Integrity (verification)
Reversible?	Yes (publicly)	Yes (only with key)	No (always one way)
Key Required?	None	Yes (Secret Key)	None
Example	Base64, ASCII	AES, RSA	SHA-256, MD5

Takeaway: encoding is for machines to read data, encryption is for authorized people to read data, hashing is to verify data hasn't changed

Real world use cases

06

The golden rule: never encrypt passwords. If a password is encrypted, it can be decrypted so if the key is stolen, every user account is compromised. Instead use

- Hashing: the hash of the password is stored, not the password itself
- Irreversibility: even the database administrator cannot see the actual data
- Salting: adding random data (that is called "salt") before hashing to prevent rainbow table attacks (pre-computed hash dictionaries)



When there is a padlock icon in a browser (meaning that its a HTTPS connexion), all three mechanisms are active simultaneously

- **Encoding** is used to transport binary certificates over text-based protocols
- **Encryption** is used as asymmetric encryption for safe key exchanges, and as a symmetric encryption to protect the actual data transfer from attackers
- **Hashing** ensures the certificate is valid and data hasn't been tampered with



IETF RFC 4648 (Base64 data encodings) [here](#)

NIST FIPS 197 (AES encryption standard) [here](#)

NIST FIPS 180-4 (secure hash standard - SHA) [here](#)

OWASP password storage cheat sheet [here](#)

CWE-312 (storage of sensitive information) [here](#)