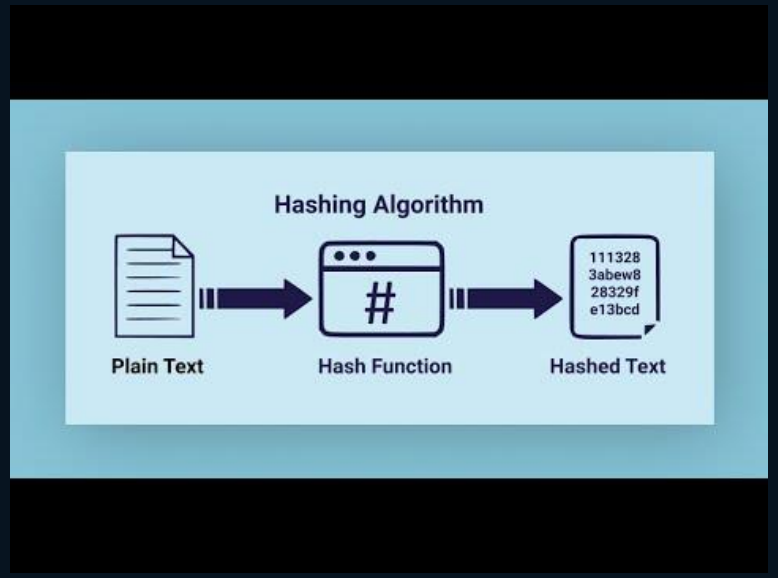
Securing Data

Securing Sensitive Data: Critical Measures & Challenges

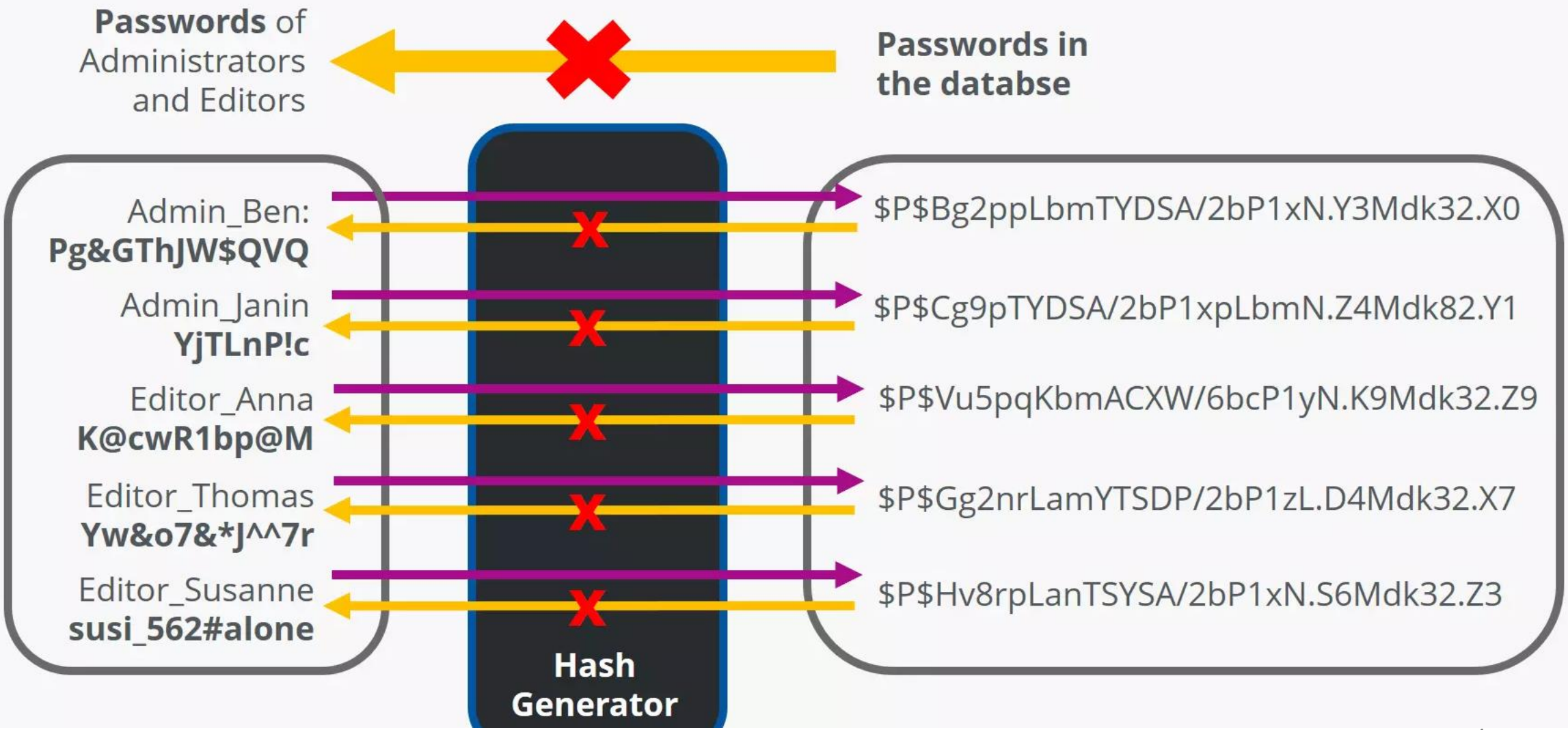
- Protecting digital information (including passwords) from unauthorized access, alteration, or disclosure across all states: at rest, in transit, and in use on servers.
- Prevents data breaches, builds user trust, ensures regulatory compliance (e.g., GDPR), and safeguards business continuity.
- **Key Storage & Security Measures:**
  - **Password Hashing & Salting:** Never store plain text; use strong one-way hashes (e.g., bcrypt) with unique random salts for each password.
  - **Data Encryption:** Encrypt all sensitive data at rest (on disks/databases) and in transit (via TLS/SSL) to prevent unauthorized reading.
  - **Strict Access Controls:** Implement Role-Based Access Control (RBAC) and the Principle of Least Privilege for all data and system access.
  - **Continuous Monitoring & Testing:** Regularly audit logs, conduct penetration tests, and monitor for anomalies to detect and address vulnerabilities.

Hashing Function



- **What is a Hashing Function?**
  - A **one-way** mathematical function that converts input (data) into **a fixed-size string** (hash).
  - **Deterministic**: Same input always produces **the same hash**.
  - Fast computation, but **infeasible to reverse**.
- **Key Properties**
  - Pre-image Resistance – **Hard to retrieve** the original input from the hash.
  - Avalanche Effect – Small input change drastically alters hash.
  - Collision Resistance – Hard to find two inputs with the same hash.
- **Common Hashing Algorithms**
  - MD5 (Deprecated due to vulnerabilities)
  - SHA-1 (Weak, being phased out)
  - SHA-256/SHA-3 (Secure, widely used)

Hash function: Password Encryption



brute force and **dictionary** attacks

Hash (MD5)	Plaintext Password
5f4dcc3b5aa765d61d8327deb882cf99	password
e10adc3949ba59abbe56e057f20f883e	123456
d8578edf8458ce06fbc5bb76a58c5ca4	qwerty
25f9e794323b453885f5181f1b624d0b	123456789

But what about **brute force** and **dictionary** attacks?

What if two passwords are identical

- **Rainbow Table**
  - A **precomputed table** of hashes for common passwords.
  - Allows attackers **to reverse hashes into plaintext quickly**.
- How Rainbow Tables Work
  - Attacker obtains password hashes (e.g., from a **breached DB** or using a **dictionary attack**).
  - Looks up hashes in the **rainbow table**.
  - Finds matches → Passwords cracked in seconds!
- Why Are They Dangerous?
  - Effective: Works against unsalted or weak hashes (MD5, SHA-1).
  - Precomputed: Covers millions of common passwords.
- How to Defend Against Them
  - Salting (Unique random data per password):
    - Renders precomputed tables useless.
    - Slow Hashes (bcrypt, Argon2):
    - Makes bulk hash reversal impractical.

Salt and Pepper

But what about **brute force** and **dictionary** attacks?

What is a Salt?

- Random data is added to each password before hashing.
- Stored openly (e.g., in the database).
- Purpose:
  - Prevents **rainbow table attacks**.
  - Ensures that identical passwords have different hashes.

What is a Pepper?

- A secret global constant (like a "second salt")
- Not stored in DB (hardcoded or in secure **server config**).
- Applied **globally to all passwords** before hashing:
- Purpose:
  - Adds another layer if the database is breached.
  - Requires the attacker to guess both the hash and the pepper.

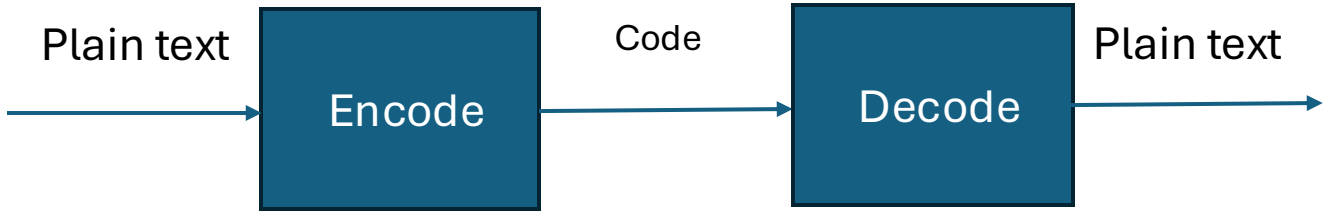
Example

- Password: "hello123"
- Salt: **x3k9vBq7** → Hashed input = hash("hello123x3k9vBq7")
- Pepper: !z$8*fL2 → Final hash = hash("hello123x3k9vBq7!z$8*fL2")

- What if the company send you your password when you forgot your password ?

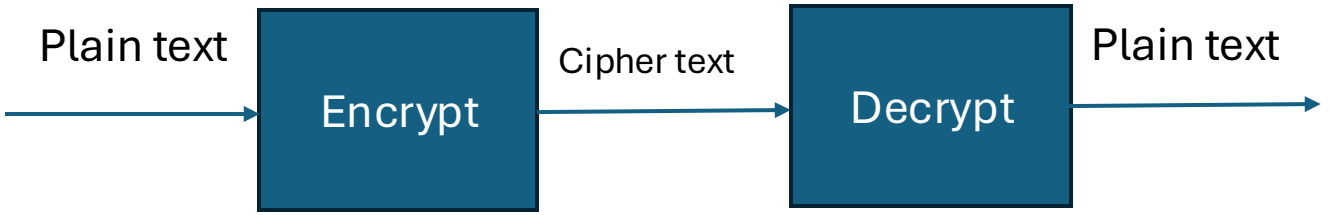
Code Books

- **Physical/digital lookup tables** mapping plaintext to coded phrases.
- Used for secure communication in the military, diplomacy, and commerce.
- **Example:**
  - **Plaintext: "Invade Normandy on June 6" → Code: "The eagle flies at midnight".**
- **Historical Significance**
  - **American Civil War: Union and Confederate armies used codebooks.**
  - **WWI/WWII: Critical for naval/military ops (e.g., Zimmermann Telegram, Japanese JN-25).**
  - **Cold War: KGB used one-time pads (advanced code books).**
- **How Code Books Worked**
  - **Pre-Shared Knowledge: Sender/receiver had identical copies.**
- **Encoding: Plaintext: "Cancel operation" → Code: "BLUE SKY"**
- **Decoding: Reverse lookup in the book.**

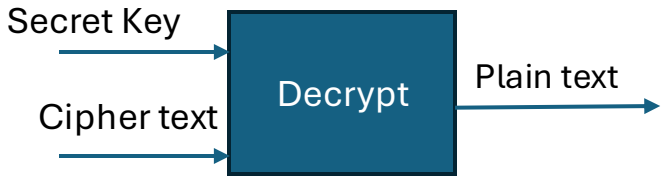


Secrete Key Cryptography: Traditional Methods

- **Traditional Cryptography**: Pre-computer methods of securing messages.
- Relied on substitution, transposition of characters instead of phrases.
- **Examples**: Caesar cipher and Vigenère cipher.
- **Usage**:
  - Military, diplomacy, and espionage (e.g., WWII, Cold War).
  - Required secure distribution of the code book.
- Traditional Cryptography is a **weak method** since they are easy to break, and a physical book is vulnerable to being stolen.

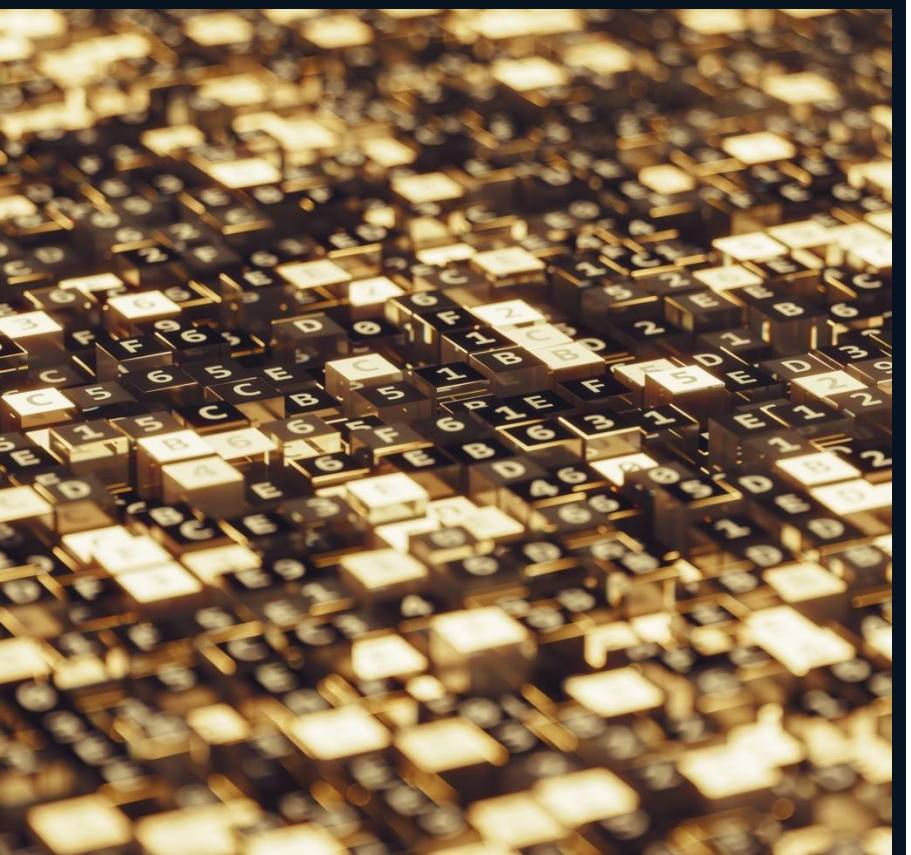


Caesar Cipher: Encryption



Caesar Cipher: Decryption

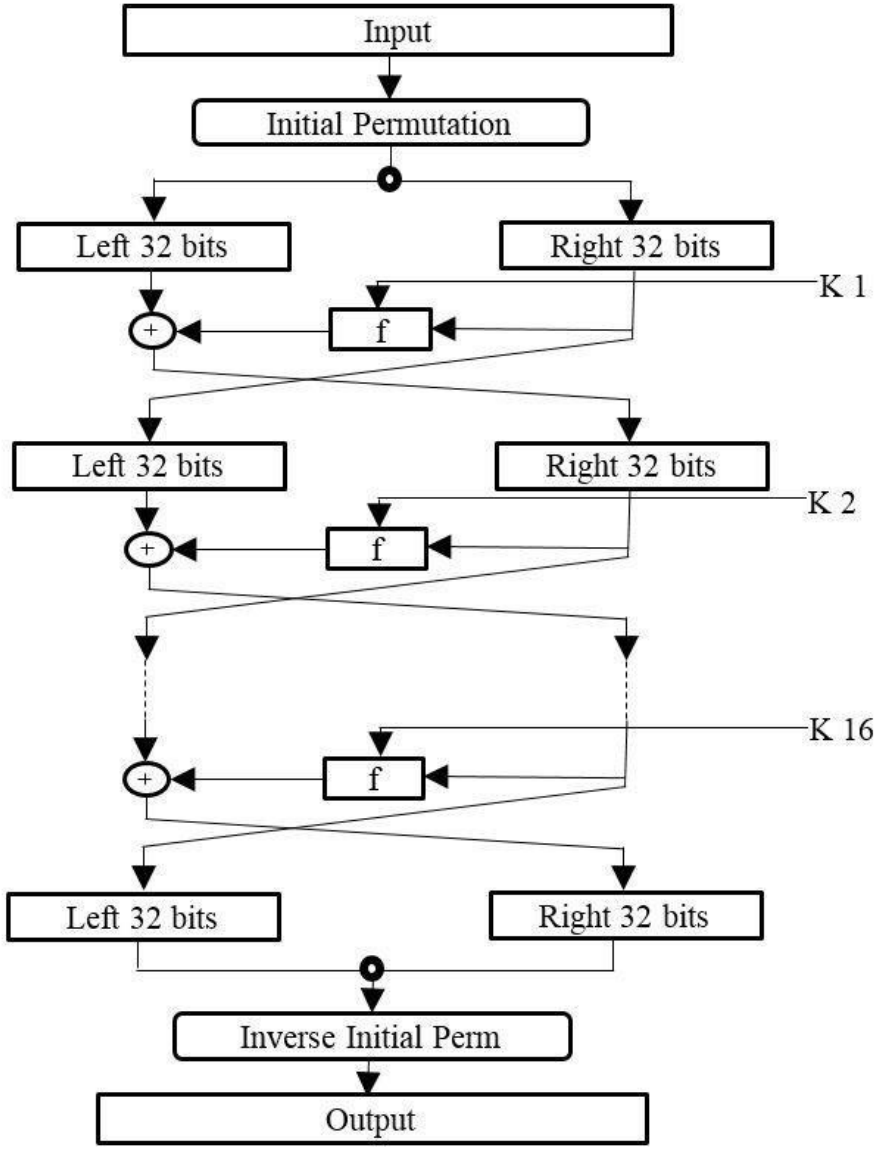
CRYPT ANALYSIS



- What is Cryptanalysis?
  - The science of analyzing and breaking cryptographic systems.
- Goals:
  - Recover plaintext from ciphertext without the key.
  - Find weaknesses in encryption algorithms.
- Breaking the Caesar Cipher
  - Brute Force: Try all 25 possible shifts (easy for Caesar).
  - Frequency Analysis: Match ciphertext letter frequencies to plaintext language (e.g., E=most common in English).
  - Known Plaintext: Use guessed words (e.g., "THE", "AND") to deduce the shift.
- TRY to break the cypher and find the key

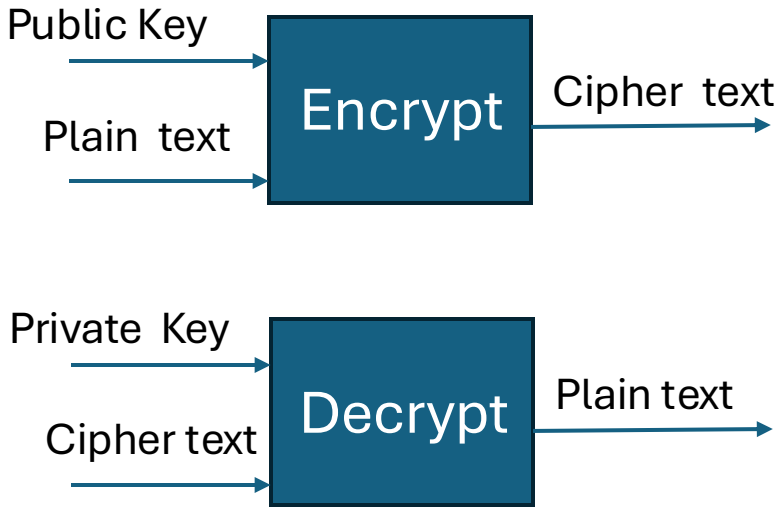
  “OR FHER GB QEVAX LBHE BINEGVAR”

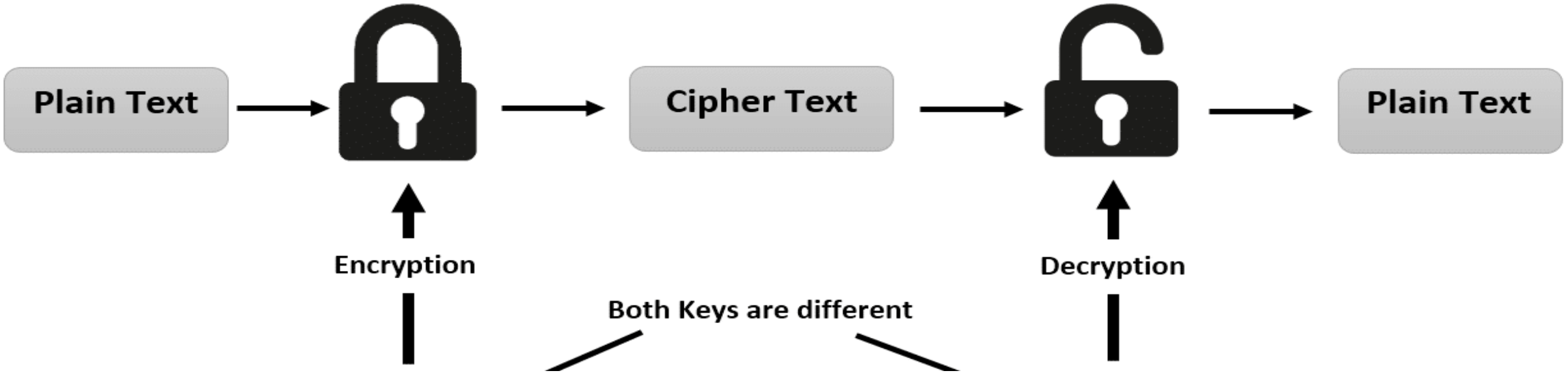
Data Encryption Standard (DES) Cipher



Public Key Encryption

- **Definition**: A cryptographic system using two keys—a public key (shared openly) for encryption and a private key (kept secret) for decryption.
- **Advantage over Secret Key Encryption**: It eliminates the need to securely share a single key, enabling secure communication over untrusted channels (e.g., the Internet).
- **How It Works**:
  - Alice generates a public-private key pair.
  - Bob encrypts a message using Alice's public key.
  - Only Alice can decrypt it with her private key.



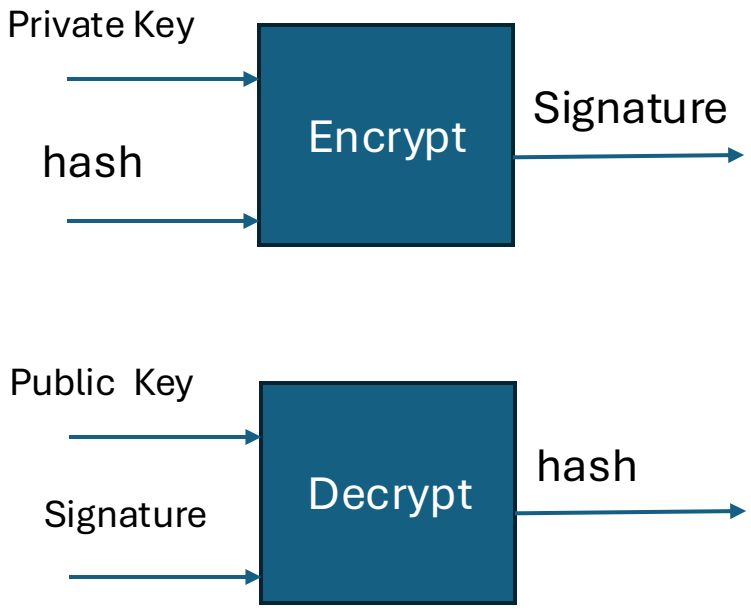


Public Key Cipher: **RSA Method**

- Choose two prime numbers: p=61, q=53,n=p × $q = 61 \times 53 = 3233$, $\lambda(3233) = lcm(p-1, q-1) = lcm(60,52) = 780$
- Choose a **coprime** number e with 780, such that *1*<e<*780*. (e=17)
- ***{Encryption Phase}***: $E(m) = m^e mod\ n \rightarrow E(65) = 65^{17} mod\ 3233 = 2790.$
- Compute d, the modular multiplicative inverse of e (mod $\lambda(n)$): $d = e^{-1} mod\ 780$, $d = 413$
- ***{Decryption Phase}*** $D(m) = m^d mod\ n \rightarrow D(2790) = 2790^{413} mod\ 3233 = 65.$

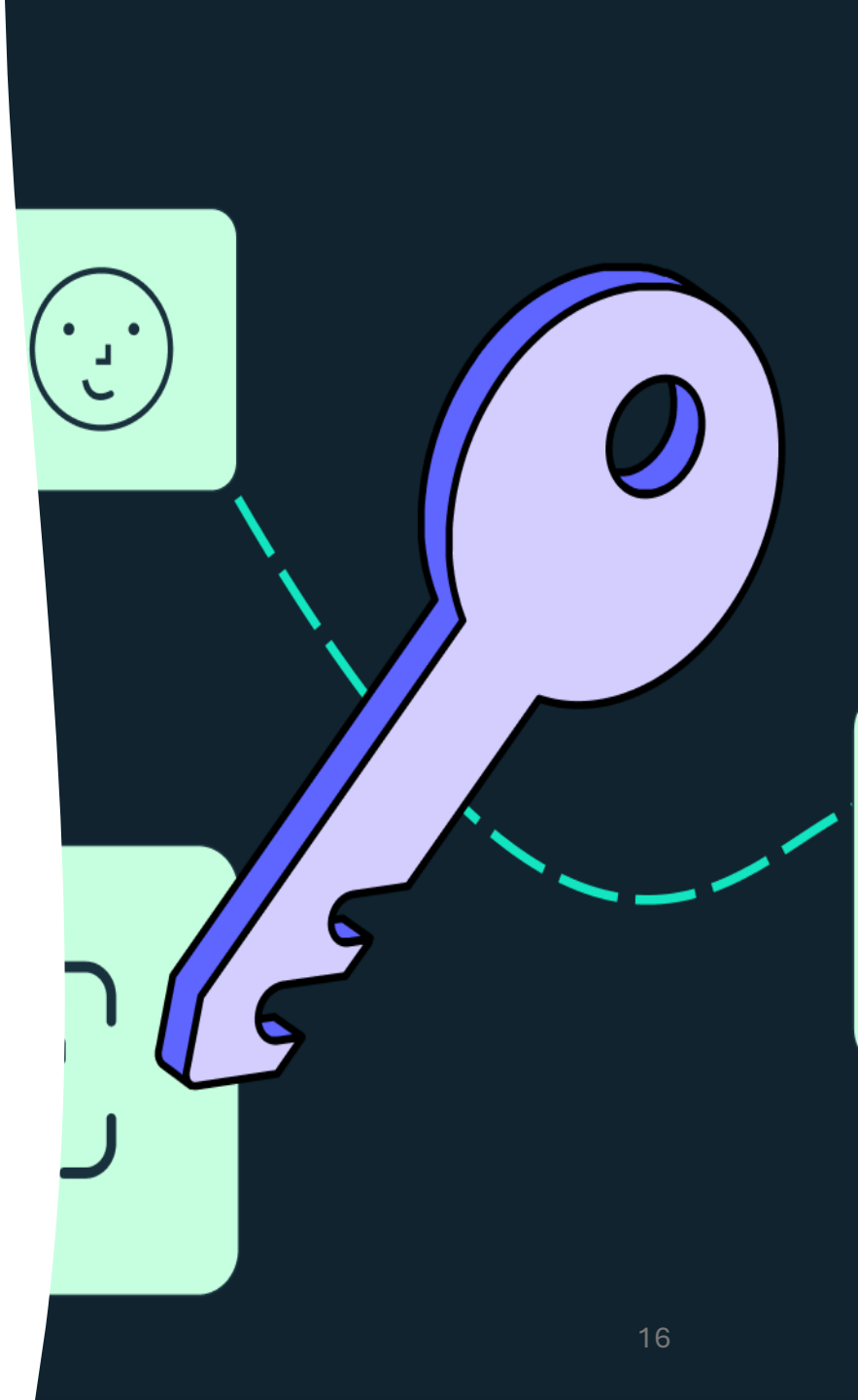
Digital Signature

- **Definition**: A cryptographic technique that ensures the authenticity, integrity, and non-repudiation of a message or document.
- **How It Works**:
  - **Signing**: The sender hashes the message and encrypts the hash with their *private key*, creating the signature.
  - **Verification**: The recipient decrypts the signature using the sender's *public key* and compares it with a freshly computed hash of the message.
- **Advantages Over Simple Encryption**:
  - **Authenticity**: Confirms the sender's identity.
  - **Integrity**: Detects any tampering with the message.
  - **Non-Repudiation**: Prevents the sender from denying they signed it.



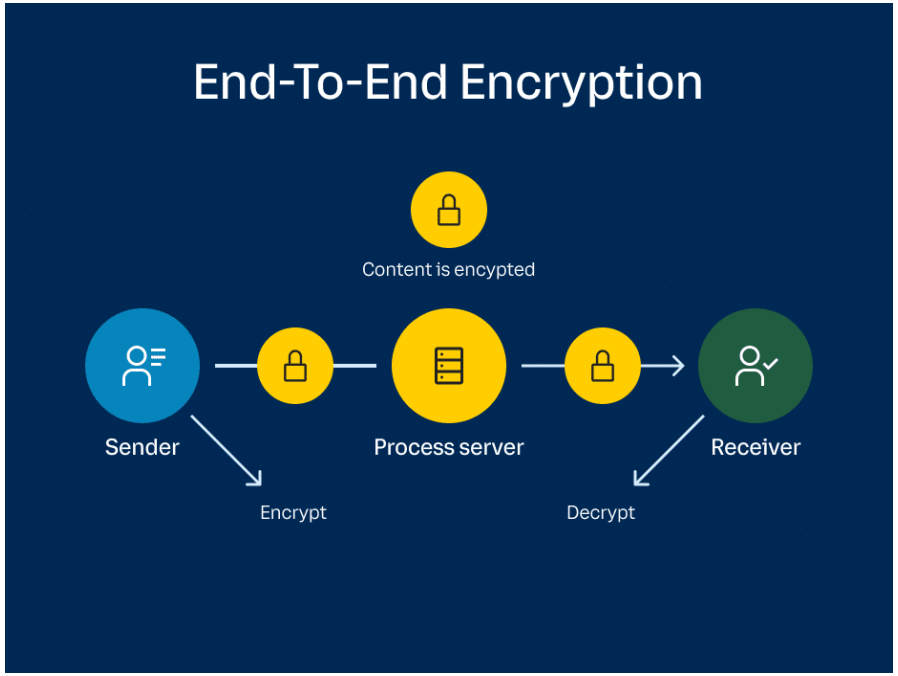
Passkeys

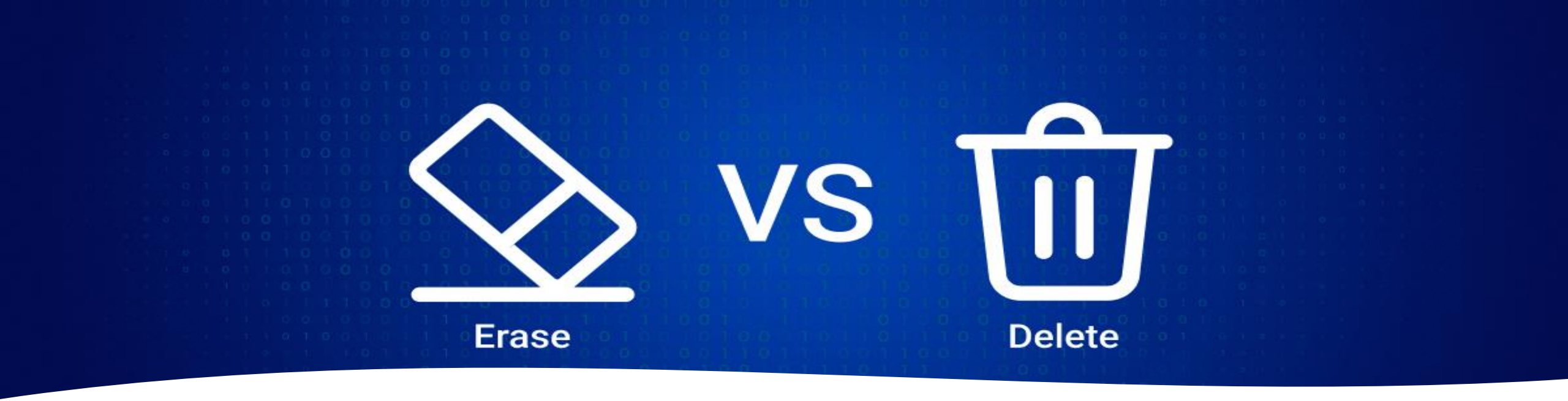
- A passwordless authentication method that uses public-key cryptography to securely log in without traditional passwords.
- **How It Works:**
  - **User Enrollment:**
    - A device (e.g., a phone or laptop) generates a public-private key pair.
    - Public key is stored on the server; private key remains securely on the device.
  - **Authentication:**
    - The server sends a challenge to the user's device.
    - Device signs the challenge with the private key (often via biometrics like fingerprint/face scan).
    - The server verifies the signature using the stored public key.
- **Advantages Over Passwords:**
  - Stronger Security – Resistant to phishing, brute force, and leaks.
  - No Password Management – Eliminates weak and reused passwords.
  - User-Friendly – Fast logins with biometrics or PIN.
- *(Supported by: FIDO Alliance, Apple, Google, Microsoft)*



Encryption in Transit and End-to-End Encryption (E2EE)

- **Encryption in Transit**
  - **Definition**: Protects data **while moving** between devices and servers (e.g., web traffic, emails).
  - **How It Works**:
    - Uses protocols to encrypt data between client and server.
    - Server authenticates via certificates; symmetric keys secure the session.
- **End-to-End Encryption (E2EE)**
  - **Definition**: Encrypts data at the sender's device and only decrypts it at the recipient's device.
  - How It Works:
    - Sender encrypts with the recipient's public key.
    - Only the recipient's private key can decrypt it.
    - No third party (including servers) can access plaintext.





Regular data deletion vs Secure data deletion

- **Regular Data Deletion:**
  - What Happens: Files are marked as "deleted" but remain recoverable until overwritten.
- **Risks:**
  - Recovery Tools (e.g., Recuva, PhotoRec) can restore files.
  - Sensitive data leaks if devices are resold/recycled.
- **Secure Deletion (Erasure):**
  - **Methods**:
    - Overwriting: **Fills** storage with **random data** (e.g., shred, DBAN).
    - Crypto-shredding: Deletes encryption keys after Full disk encryption, rendering data unreadable.
    - Physical Destruction: Drilling, degaussing (HDDs), or shredding!

Full Disk Encryption (FDE)

- "Protecting Data Where It Lives"
- What FDE does:
  - Encrypts entire storage drives (OS, apps, files)
  - Requires authentication (password/PIN) to boot
- Tools:
  - BitLocker (Windows), FileVault (macOS), LUKS (Linux)
- Its importance
  - Stops thieves from accessing data on stolen devices
- Encryption at Rest
  - What it does:
    - Encrypts individual files/databases while stored
    - Works even if the disk is already encrypted with FDE
- **With this encryption in place, your data remains inaccessible without your password or biometric authentication, even if your device is lost or stolen.**

How Ransomware Weaponizes Full Disk Encryption When Your Protection Becomes the Attackers' Tool

- FDE's Normal Role is to protect data at rest from physical theft
  - Requires authentication (password/biometrics) to decrypt the drive
- Ransomware's Twist:
  - Exploits Unlocked Systems:
    - Once the user logs in, FDE decrypts data – ransomware encrypts live files
    - Attackers mimic FDE behavior to make encryption appear "official"

Questions