DataCrypt : Client Side File Encryption for Secure Cloud Storage

Under the guidance of Dr. Bindu Verma
(Department of Information Technology, DTU)

Presented By: Rajnish Kumar (2K21 / IT / 139)
Ritik (2K21 / IT / 145)
Rugung Daimary (2K21 / IT / 151)

INTRODUCTION

The Problem

- In today's digital world, organizations increasingly rely on cloud services for data storage and collaboration.
- However, cybersecurity threats like data breaches and unauthorized access during file transfers make it risky to store sensitive information on the cloud without protection.
- Organizations face challenges balancing ease of use, compliance with regulations like GDPR and HIPAA, and ensuring absolute security.

The Solution

- A system that enables encryption of data locally before it's uploaded to the cloud.
- Ensures sensitive information remains secure and private, even if the cloud environment is compromised.
- Accessible through an open source repository for easy deployment within an organization's own system.

Use Case: Client-Side Encryption (CSE)

1. **Key Management**
   - CSE: Full user control over encryption keys.
   - Others: Cloud provider has some level of access to keys (e.g., Cloud-Managed Keys).
2. **Security**
   - CSE: Maximum security—cloud never sees plaintext or keys.
   - Others: Provider’s infrastructure introduces risks of breaches or insider attacks.
3. **Privacy**
   - CSE: Complete privacy—only the user can decrypt data.
   - Others: Cloud providers can potentially access data.
4. **Compliance**
   - CSE: Fully compliant with strict regulations (e.g., GDPR, HIPAA).
   - Others: Compliance depends on the provider's practices.
5. **Key Loss Risk**
   - CSE: Key loss means permanent data loss—ensures no unauthorized recovery.
   - Others: Providers can assist in recovery but increase risks.

KEY FEATURES

1) Advanced Encryption Techniques

- AES (Advanced Encryption Standard): For encrypting files, ensures that the encryption is both fast and secure.
- ECC (Elliptic Curve Cryptography): To manage encryption keys securely,allows efficient key exchange while reducing computational cost

2) Local Processing

- Encryption happens entirely within the organization’s environment using Java-based tools and servlets.

3) Cloud Compatibility

- Data encrypted by our system can be securely stored on any cloud service provider.

4) **Open-Source and Flexible**

- Accessible through our GitHub repository, allowing for customization to meet specific requirements.

ECC (Elliptic Curve Cryptography)

1) Introduction to ECC:

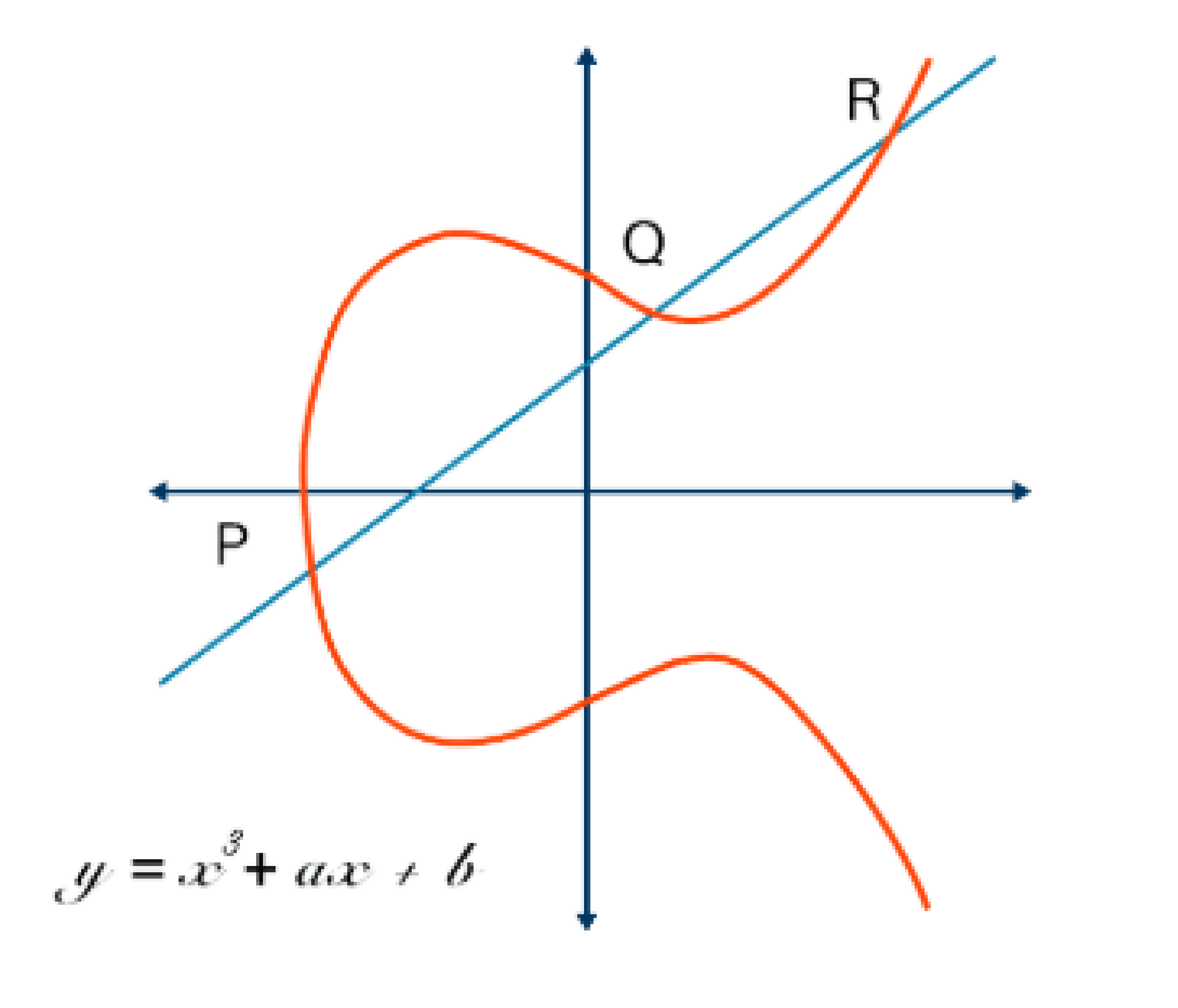
- ECC is a modern public-key cryptography system using elliptic curves.
- Provides strong security with smaller key sizes compared to RSA and DSA.

2) Key Features of ECC:

- Public-Key Cryptography
- Smaller Key Sizes
- Efficiency

3) How ECC Works:

- ECC uses elliptic curves over finite fields.
- Equation: $y^2 = x^3 + ax + b$
  where a and b define the curve.
- Valid curves must meet security properties.

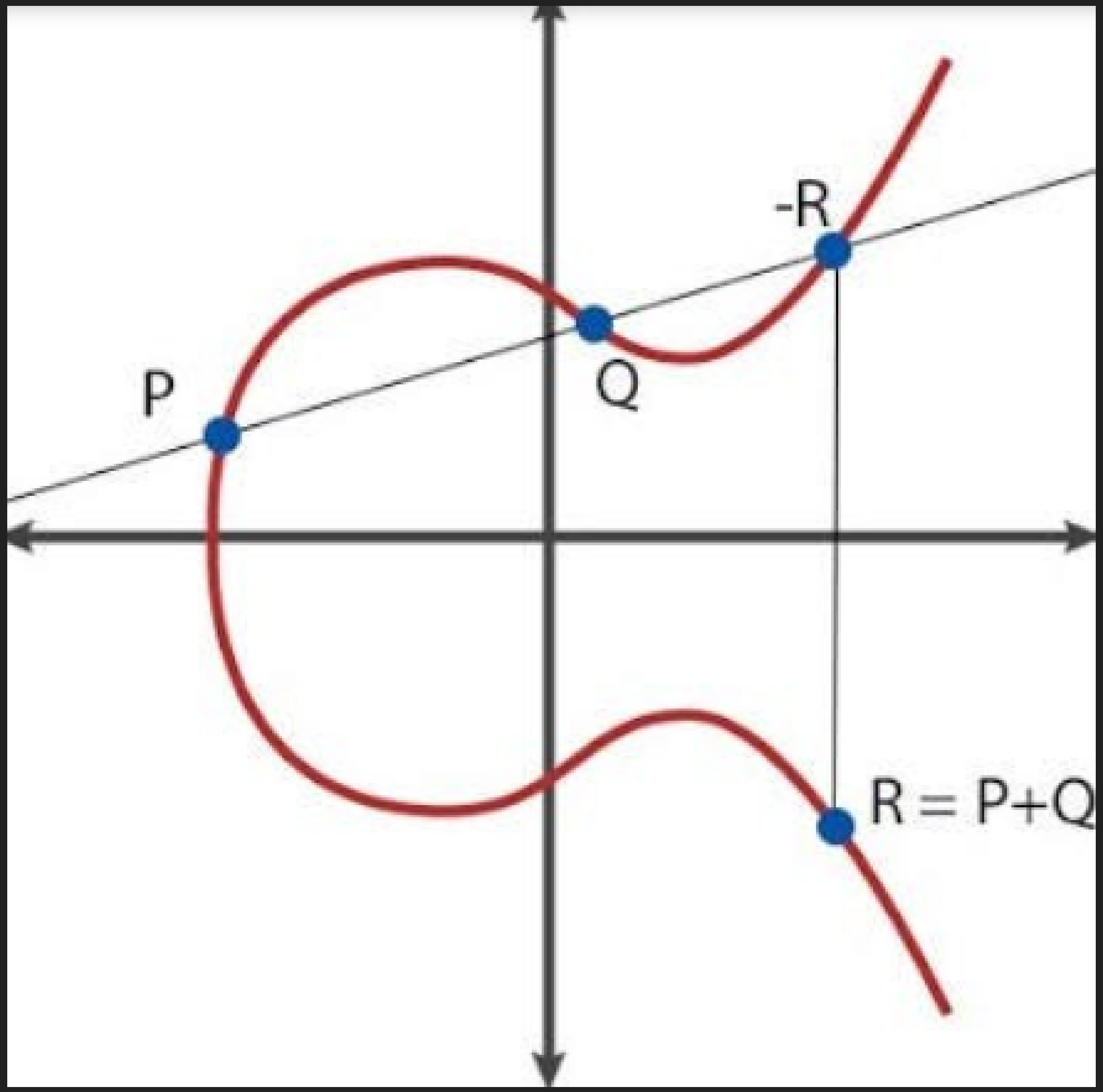


Point Addition and Point Multiplication:

- Point Addition:
  - Combine two points P and Q on the curve to compute R=P+Q.
  - Well-defined operation with properties useful for cryptography.
- Point Multiplication:
  - Multiply a point P by a scalar k to compute Q=kP.
  - Computationally easy to perform but reversing it (finding k from P and Q) is infeasible, ensuring security.
  - Security Basis: Relies on the Elliptic Curve Discrete Logarithm Problem (ECDLP).

4) ECC Key Generation:

- Private Key: A random integer k within a specified range.
- Public Key: Computed as Q=kG, where:
  - G is a generator point on the curve (publicly known).
  - Q is the public key derived from the private key k.



5. ECC Cryptographic Operations:
ECC can be used for various cryptographic operations, including:

ECC Key Exchange (ECDH - Elliptic Curve Diffie-Hellman):
This is used to securely exchange cryptographic keys over a public channel. Both parties generate their own private and public keys. Using their private key and the other party's public key, they can both compute the same shared secret.

$$\text{Shared Secret} = kA \times P_B = kB \times P_A$$

Where P_A and P_B are the public keys of Alice and Bob, and kA and kB are their respective private keys.

ECDH - Elliptic Curve Diffie-Hellman

ECDH is used to create a shared key. Bob will generate a public key and a private key by taking a point on the curve. The private key is a random number (d_B) and the Bob's public key (Q_B) will be:

$$Q_B = d_B \times G$$

Alice will do the same and generate her public key (Q_A) from her private key (d_A):

$$Q_A = d_A \times G$$

They then exchange their public keys. Alice will then use Bob's public key and her private key to calculate:

SharekeyAlice$= d_A \times Q_B$

This will be the same as:

SharekeyAlice$= d_A \times d_B \times G$

Bob will then use Alice's public key and his private key to determine:

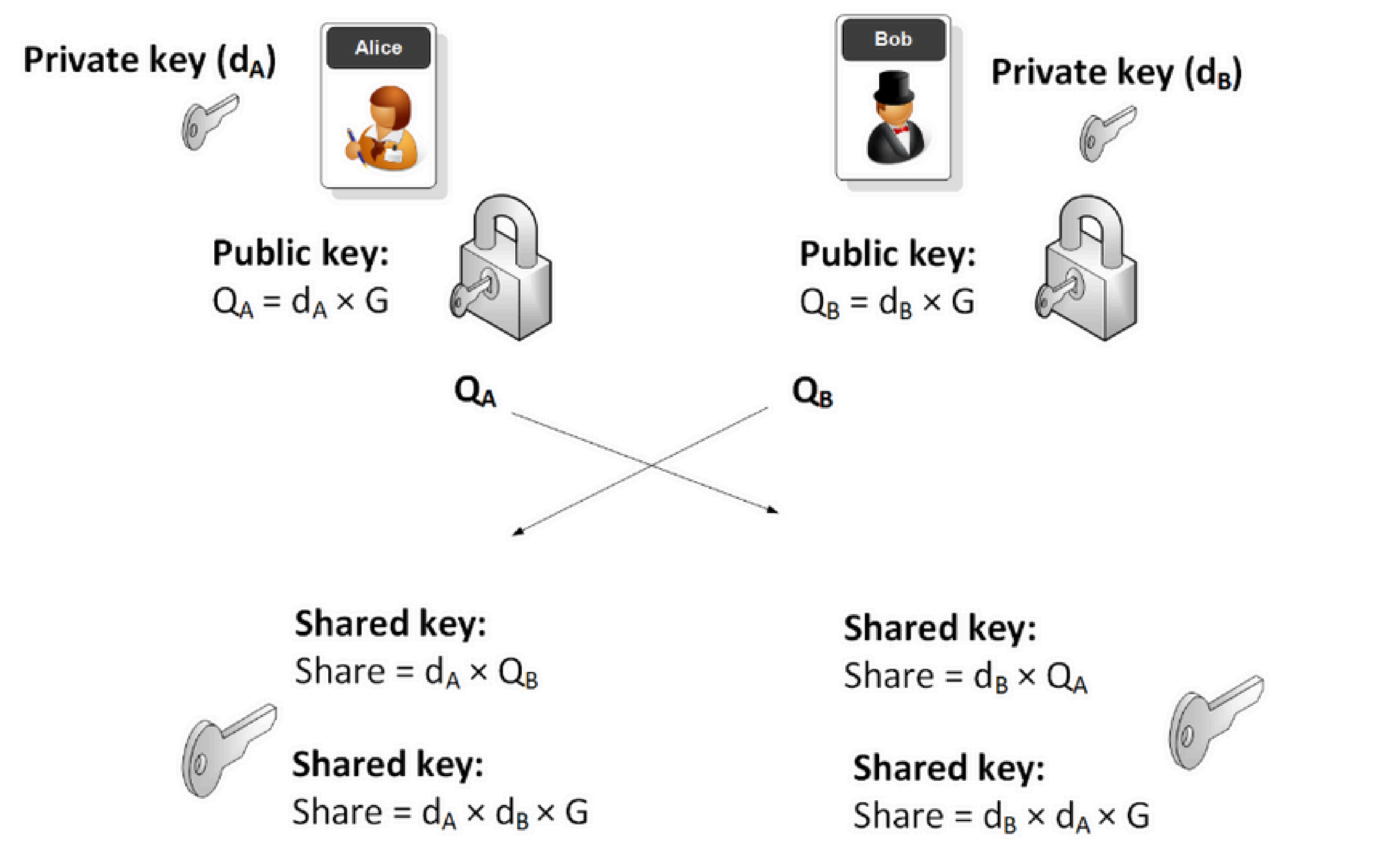
SharekeyBob $= d_B \times Q_A$

This will be the same as:

SharekeyBob$= d_B \times d_A \times G$

And the keys will thus match.

The following illustrates the process:



AES (Advanced Encryption Standard)

1) Introduction to AES:

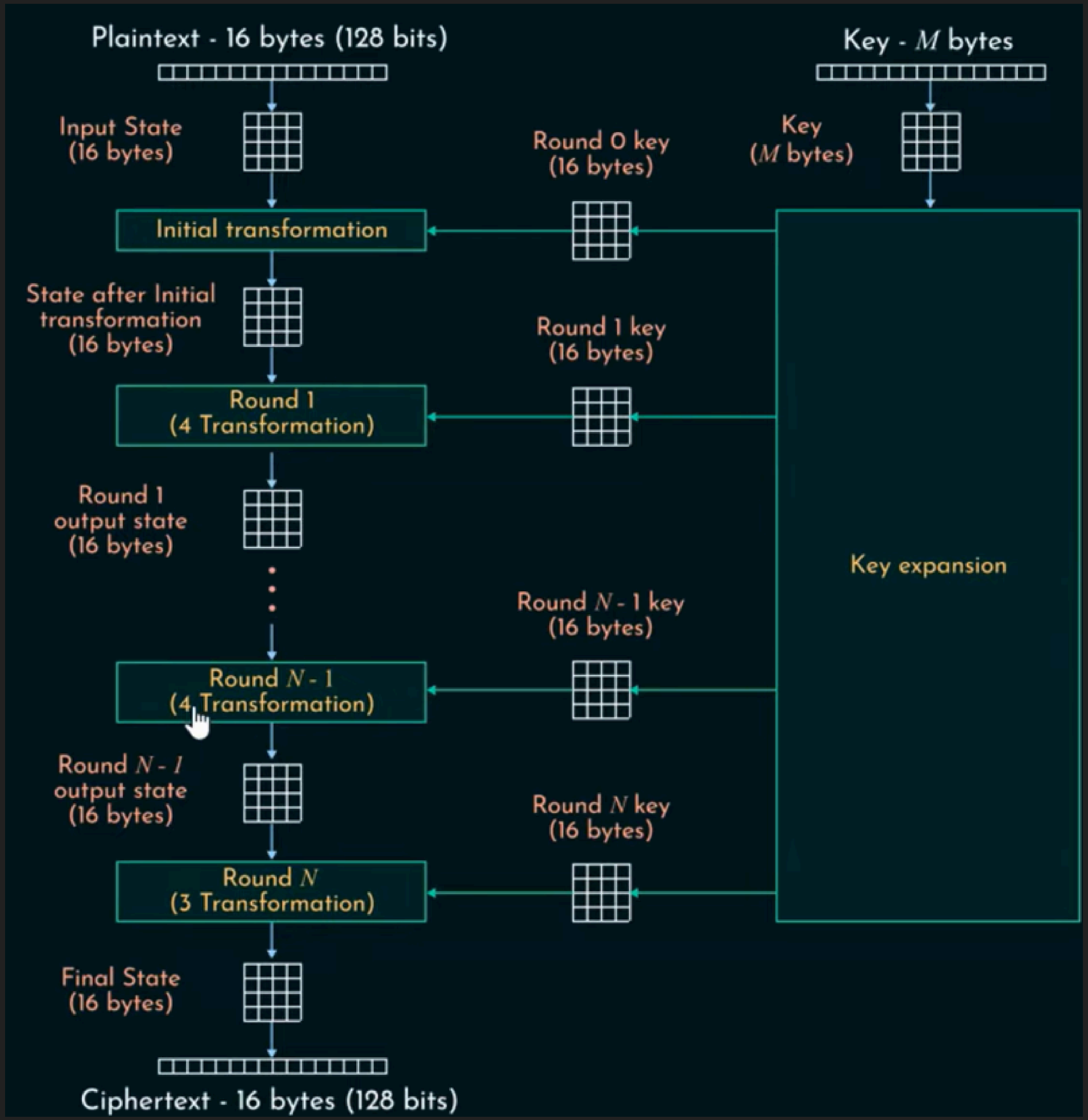
- AES is a symmetric encryption algorithm (same key for encryption and decryption).
- Established by NIST in 2001 to replace DES.
- Known for efficiency, security, and adaptability.

2) Key Features of AES:

- Symmetric Encryption: Same key for both encryption and decryption.
- Block Cipher: Operates on 128-bit data blocks.
- Key Sizes:
  - AES-128 (128-bit key)
  - AES-192 (192-bit key)
  - AES-256 (256-bit key)

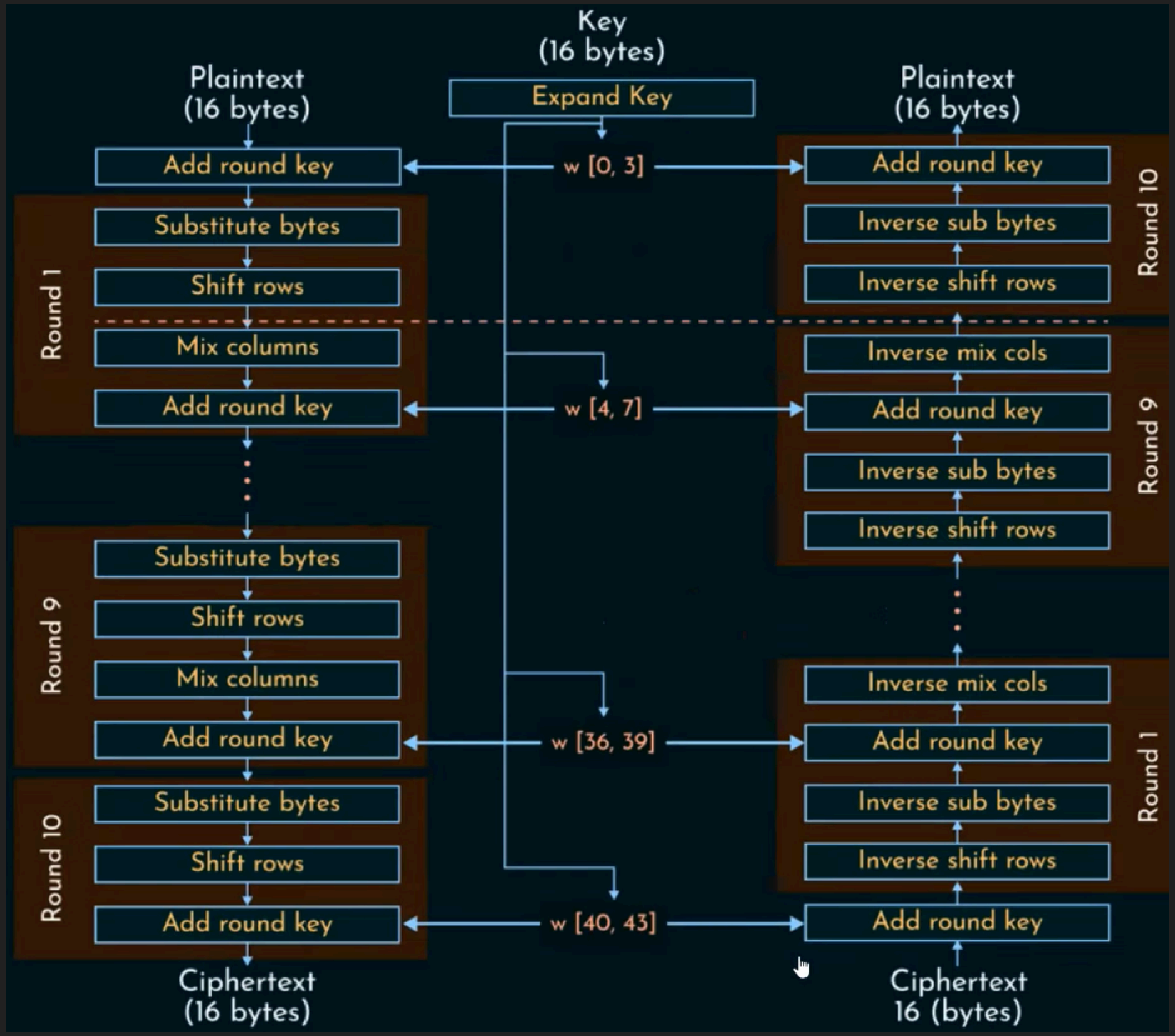
3) AES Structure:

- Data undergoes multiple transformation steps.
- Rounds:
  - AES-128: 10 rounds
  - AES-192: 12 rounds
  - AES-256: 14 rounds



4) AES Encryption Process:

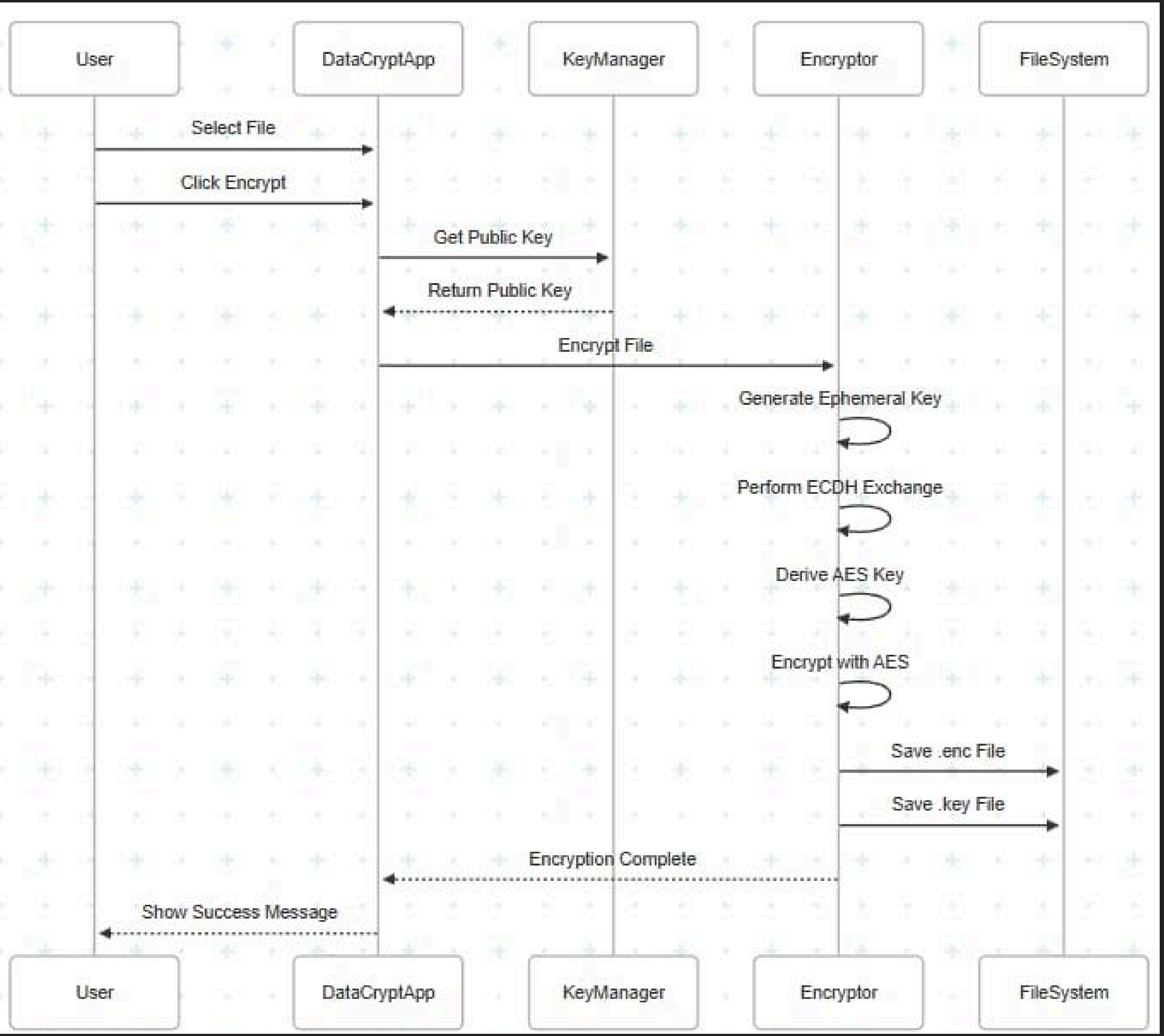
- **Key Expansion:** Encryption key is expanded into round keys.
- **Initial Round:** Add Round Key (XOR the data block with round key).
- **Main Rounds:**
  - **Sub Bytes:** Substitute each byte using the S-box.
  - **Shift Rows:** Shift rows of the data block.
  - **Mix Columns:** Mix each column for diffusion.
  - **Add Round Key:** XOR the state with the round key.
- **Final Round:** Similar to main rounds but without Mix Columns. Involves Sub Bytes, Shift Rows, and Add Round Key.



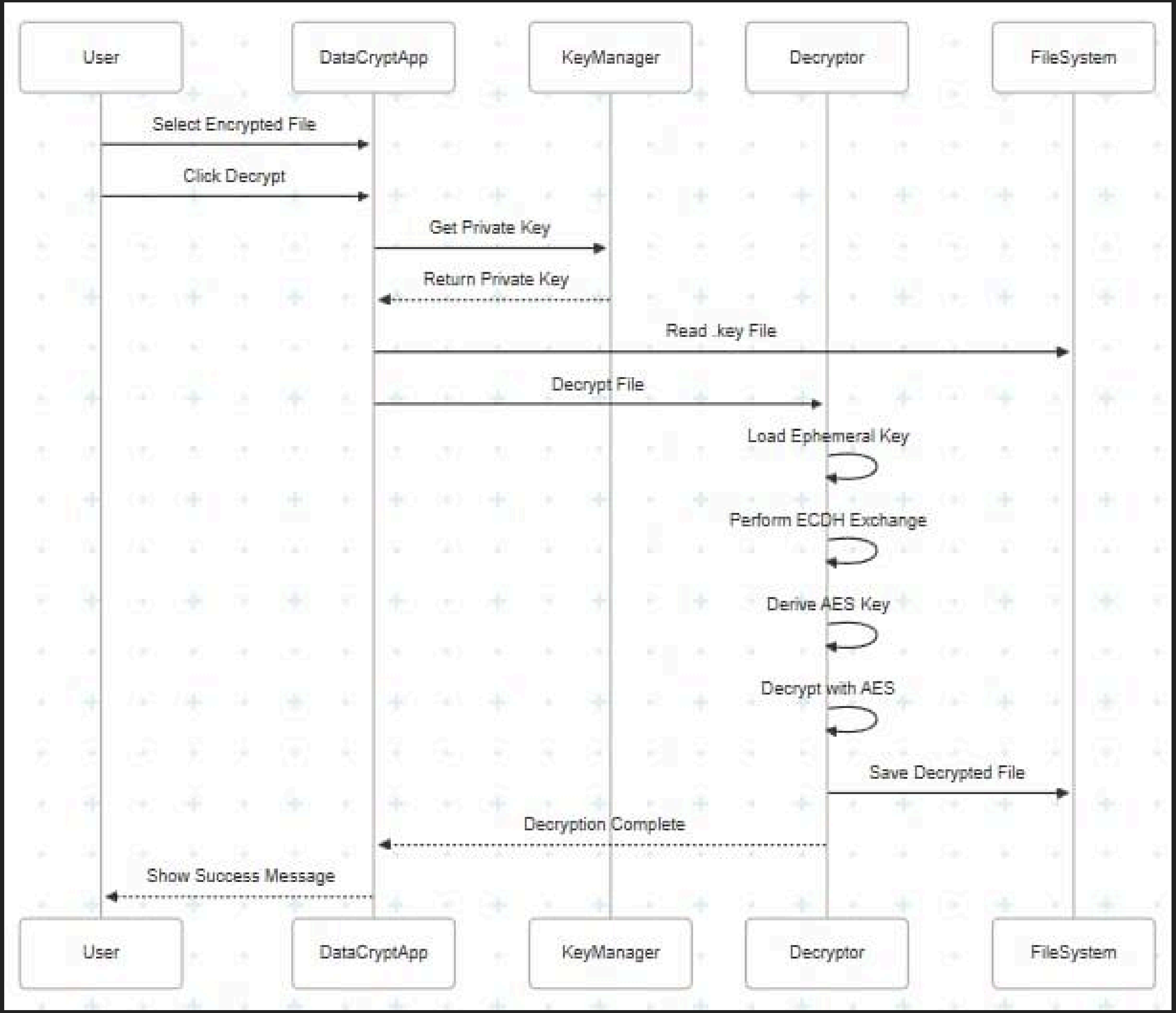
5) AES Decryption Process:

- Works in reverse order of encryption.
- Apply round keys in reverse.
- Use inverse operations:
  - Inverse SubBytes
  - Inverse ShiftRows
  - Inverse MixColumns
  - AddRoundKey

FILE ENCRYPTION



FILE DECRYPTION



LIVE DEMONSTRATION

THANK YOU!!!!