

Data Security

CloudSiksha

Cryptography



Integrity



Confidentiality



Authenticity

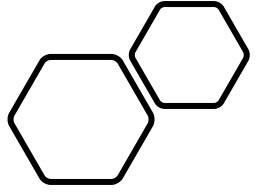


Non-Repudiation

Encryption

- Encryption is the process of scrambling or enciphering data so it can be read only by someone with the means to return it to its original state





Types of Cryptography



Symmetric Key

Stream Ciphers

Block Ciphers



Asymmetric Key

Symmetric Encryption

From: Computer Security Principles & Practice (3rd Edition)

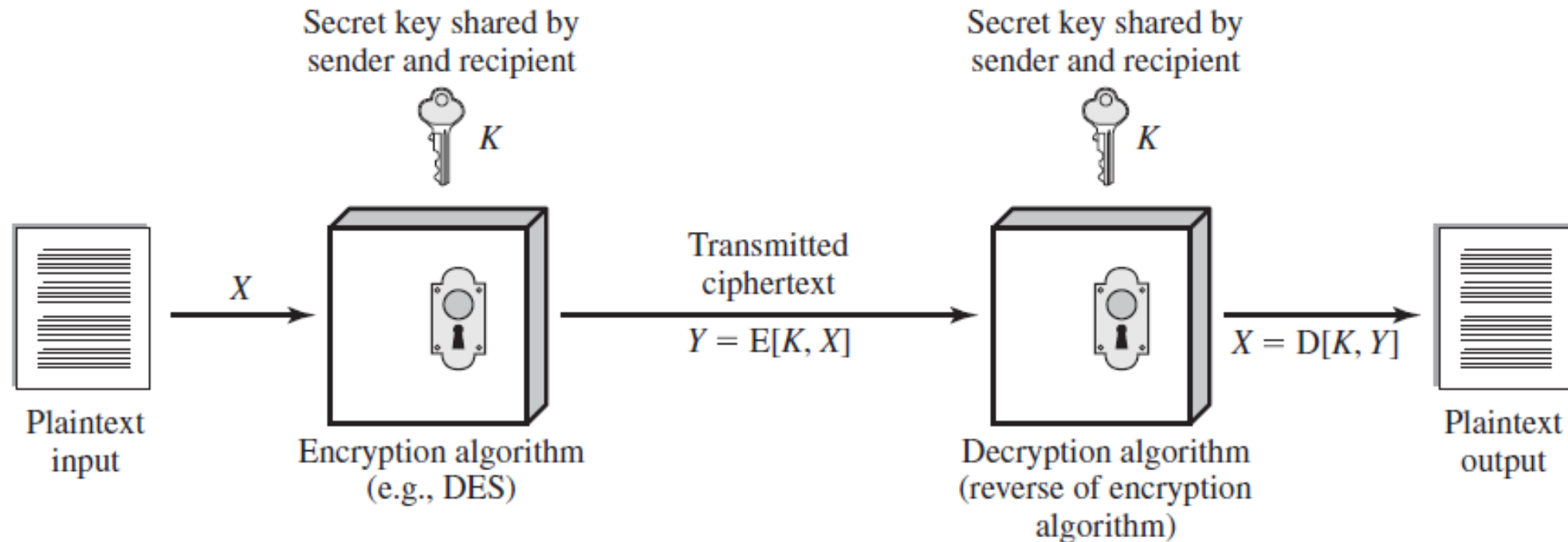


Figure 2.1 Simplified Model of Symmetric Encryption

Symmetric Encryption Algorithms

- *From: Computer Security Principles & Practice (3rd Edition)*

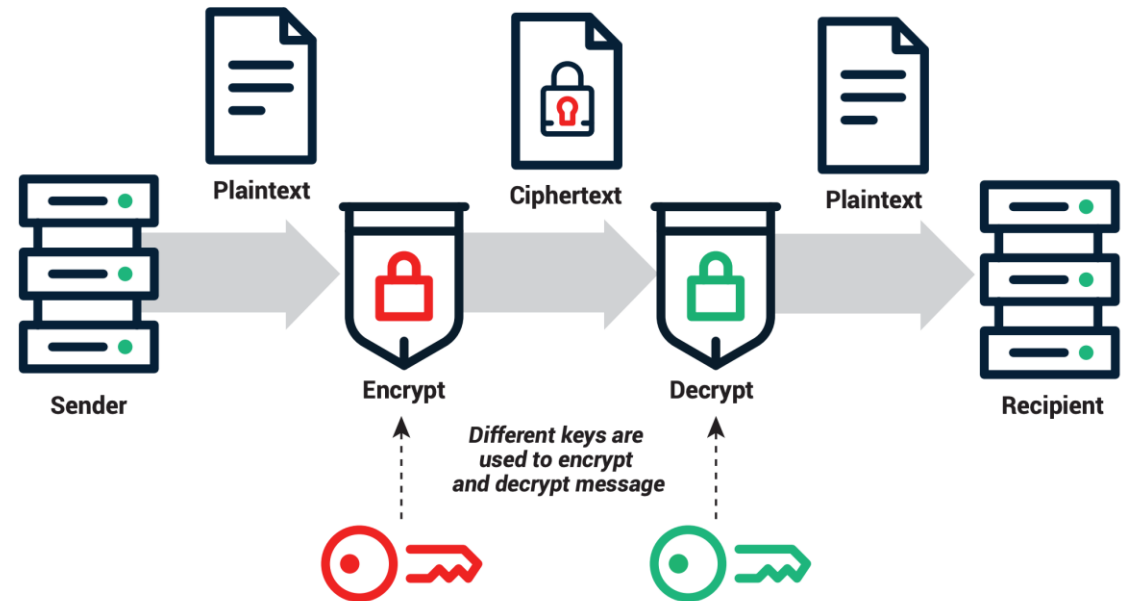
Table 2.1 Comparison of Three Popular Symmetric Encryption Algorithms

	DES	Triple DES	AES
Plaintext block size (bits)	64	64	128
Ciphertext block size (bits)	64	64	128
Key size (bits)	56	112 or 168	128, 192, or 256

DES = Data Encryption Standard

AES = Advanced Encryption Standard

Asymmetric Key Encryption / Decryption



RSA Algorithm

- RSA (Rivest–Shamir–Adleman)
- RSA is a widely-used public key cryptographic algorithm.
- It relies on the mathematical difficulty of factoring large integers.
- RSA involves a public key for encryption and a private key for decryption.
- Key generation includes choosing two large prime numbers.
- RSA supports secure data transmission and digital signatures.



Elliptic Curve Cryptography (ECC)

- ECC is a public key cryptography approach using elliptic curves over finite fields.
- It provides strong security with shorter key lengths compared to traditional methods.
- ECC is widely used for securing mobile devices and constrained systems.
- Key operations include point addition and scalar multiplication on elliptic curves.
- It supports encryption, digital signatures, and key exchange protocols like ECDSA and ECDH.

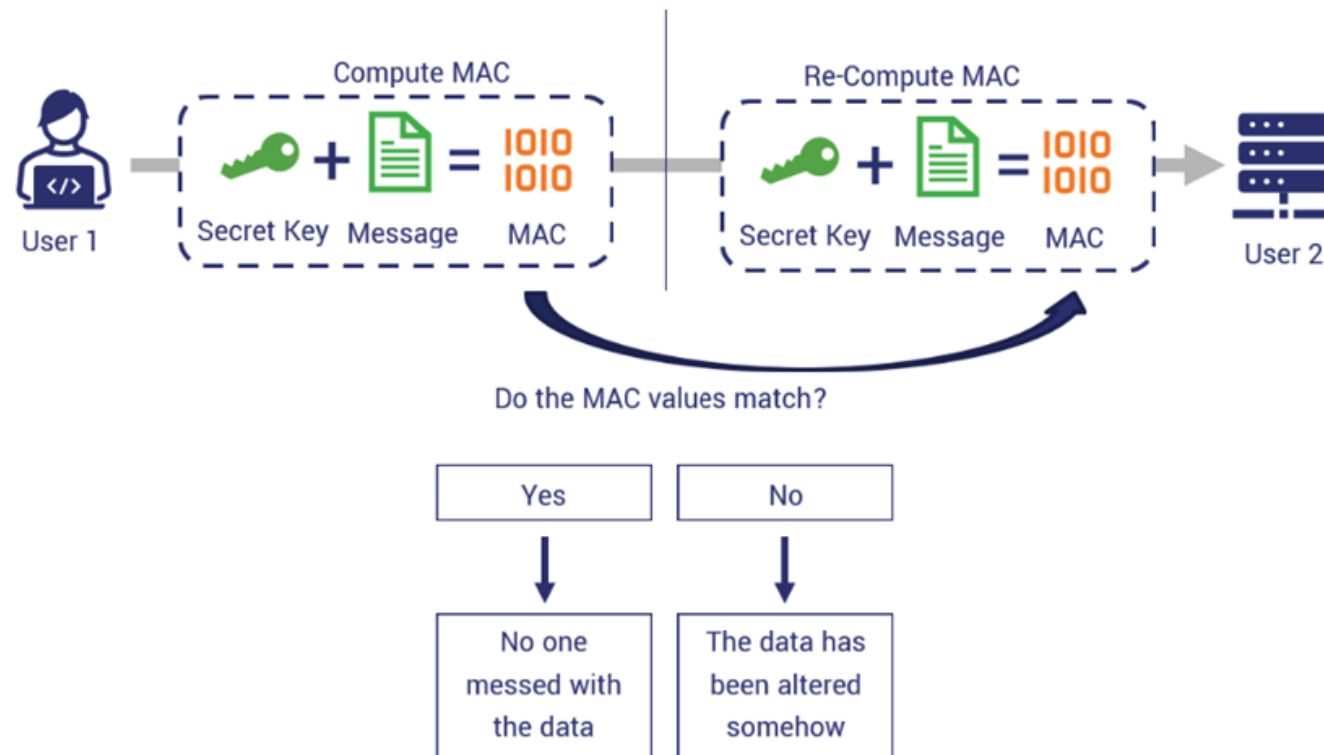


RSA vs ECC

Feature	RSA	ECC
Security basis	Factoring large integers	Elliptic Curve Discrete Logarithm Problem
Key size for equivalent security	Much larger (e.g., 2048-bit)	Much smaller (e.g., 256-bit ECC \approx 3072-bit RSA)
Performance	Slower (esp. for signing & key exchange)	Faster, more efficient (smaller computations)
Bandwidth / Storage	Larger keys, larger signatures	Smaller keys & signatures (good for constrained devices)
Adoption	Very widely used historically	Increasingly used (TLS, cryptocurrencies, mobile, IoT)
Quantum resistance	Not resistant (Shor's algorithm breaks it)	Also not resistant (quantum breaks ECC too)

Message Authentication Code (MAC)

How Message Authentication Codes Work



Digital Signature



SIGNING



VERIFICATION



MAC vs Digital Signatures

Feature	MAC	Digital Signature
Keys	Shared secret	Public/private pair
Verifier	Only secret holders	Anyone with public key
Authentication	Yes	Yes
Integrity	Yes	Yes
Non-repudiation	 No	 Yes
Speed	Faster	Slower
Use case	Internal trust (e.g., API calls, VPNs)	Public verification (e.g., contracts, certificates)

Public vs Private Keys



Scenario	Operation	Who Uses It	Which Key is Used	Purpose
Encryption (confidentiality)	Encrypt a message	Sender	Receiver's Public Key	Ensures only the receiver can decrypt (since only they have the matching private key).
	Decrypt a message	Receiver	Receiver's Private Key	Recovers the original message securely.
Digital Signature (authenticity & integrity)	Sign a message	Sender	Sender's Private Key	Generates a signature only the sender could have made.
	Verify a signature	Receiver / Anyone	Sender's Public Key	Confirms the message was signed by the sender and not altered.

Public Key Certificates

From: Computer Security Principles & Practice (3rd Edition)

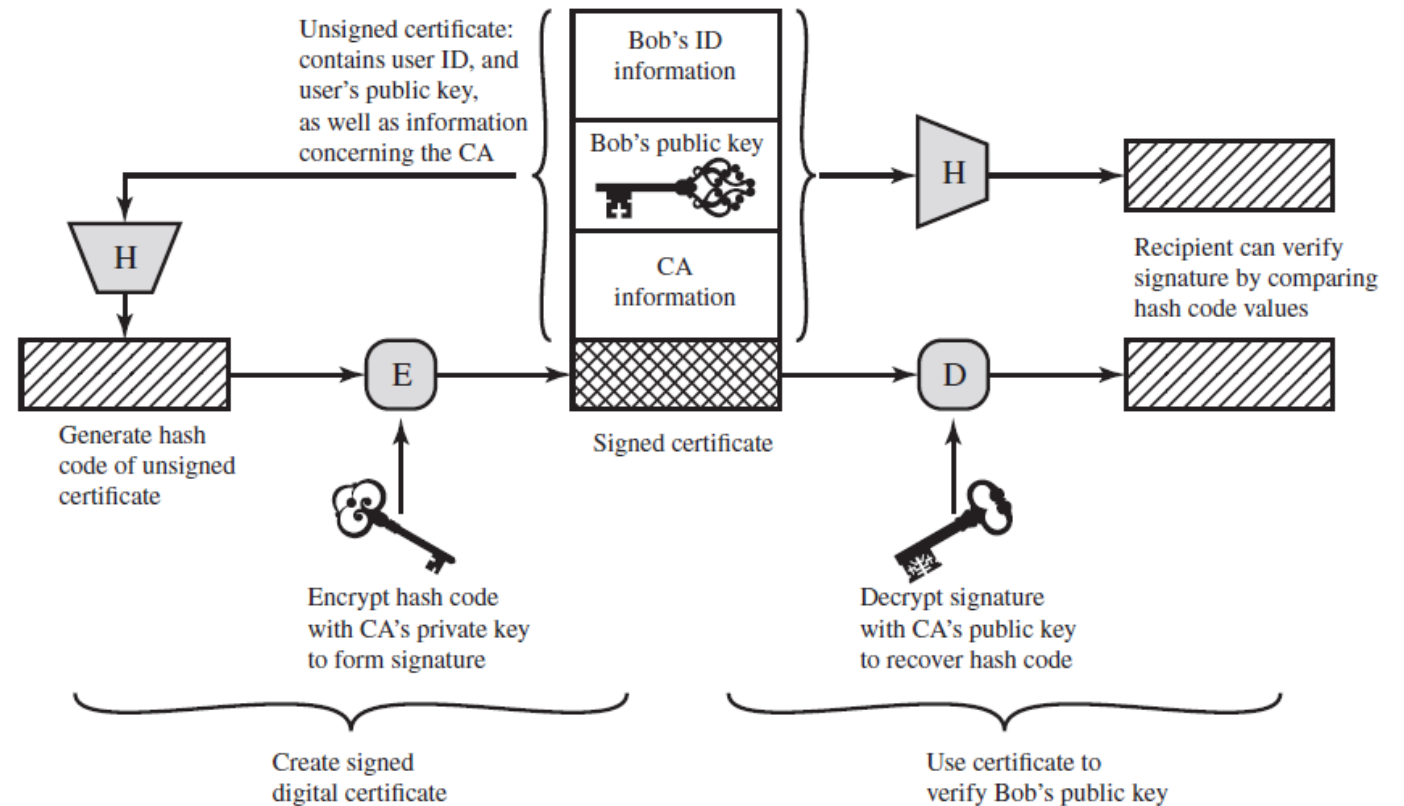


Figure 2.7 Public-Key Certificate Use

HSM



Hardware Security Modules



Designed to store secret keys safely



Tamper proof



Certified against International Standards, ex: FIPS 140



For Cloud based infrastructure, Cloud based HSM provide better speed

AWS CloudHSM

- AWS CloudHSM is a cloud-based hardware security module (HSM) that enables you to easily generate and use your own encryption keys on the AWS Cloud.
- With CloudHSM, you can manage your own encryption keys using FIPS 140-2 Level 3 validated HSMs.
- CloudHSM offers you the flexibility to integrate with your applications using industry-standard APIs
- It is a fully-managed service that automates time-consuming administrative tasks for you, such as hardware provisioning, software patching, high-availability, and backups

Post Quantum Cryptography

- Shor's algorithm
 - **Factors integers and solves discrete logarithms fast**
 - **RSA and ECC can both be broken**
- Grover's algorithm
 - **Speeds up brute force search**
 - **AES 128 not safe. AES 256 still safe**
- Quantum resistant algorithms being developed

KMS

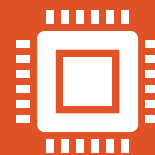


Key Management
System



Create and
Control
Encryption Keys

These keys
are used to
encrypt our
data



Integrated with
other AWS
Services

S3, EBS,
RDS, RedShift
etc

KMS Features



Create Master
Keys

These
cannot be
exported
out of the
service



Encrypt, Decrypt, Re-
Encrypt using Master
Keys



Generate Data
Encryption
Keys

Both plain
text and
encrypted

Used to
encrypt
data

Terminology

Customer Master Key (CMK)

- Can protect upto 4KB of data
- CMK does not leave KMS unencrypted

Data Keys

- Used to encrypt data
- Plain text and encrypted data key
- Plain text used to encrypt data.
Encrypted data key stored with data

Envelope Encryption

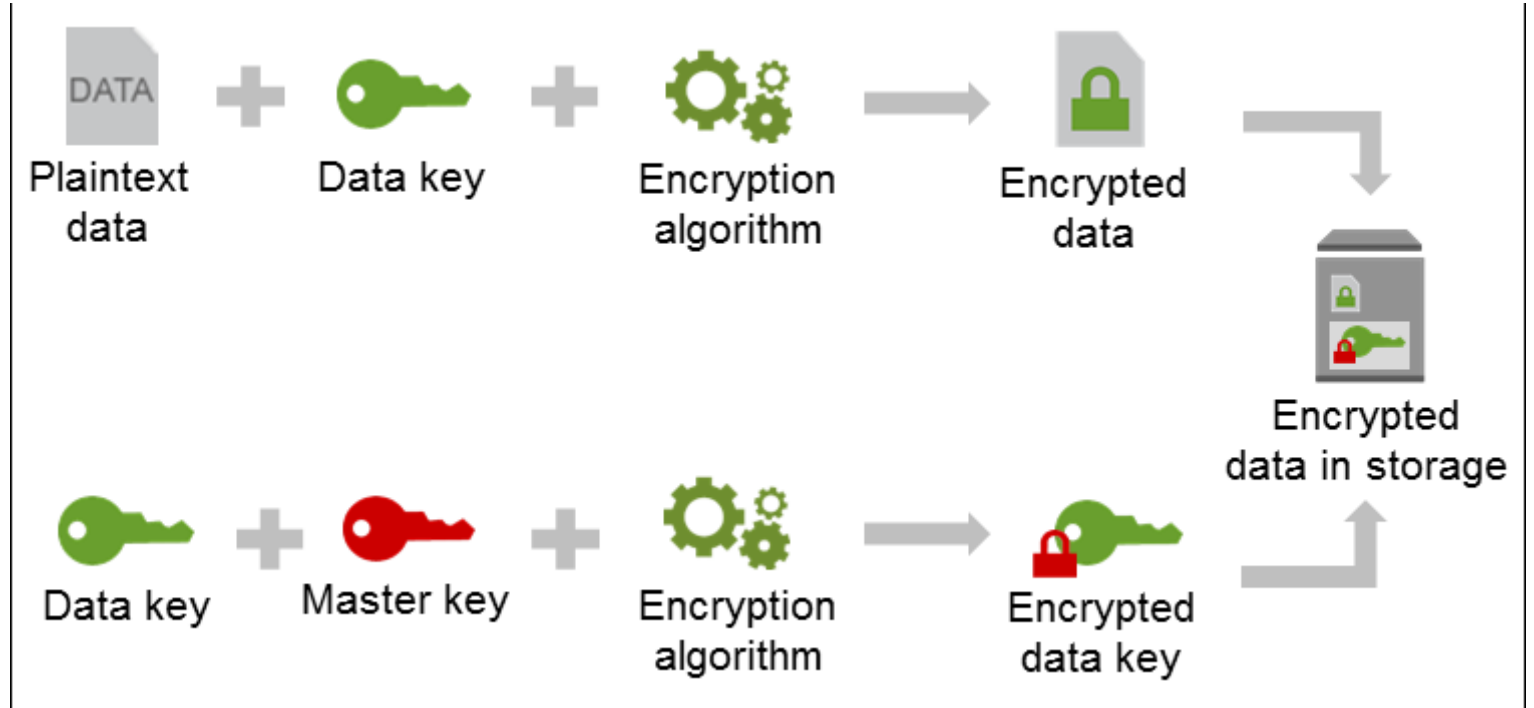
Envelope
Encryption used
to protect data

Plain text is
encrypted using
unique data key

Data key is then
encrypted and
stored with data

Envelope Encryption

Image Source: AWS
Documentation



Encryption in S3

Server-Side Encryption (SSE)

- SSE – S3
 - S3 Manages Data and Master keys
- SSE – C
 - Customers manage their own keys
- SSE – KMS
 - Keys managed by KMS

S3 Encryption Client

- Encrypt in application and upload to S3

Other
services
that can
use KMS

CloudTrail

EBS

DynamoDB

SQS

Glacier

Kinesis

Thanks



CloudSiksha

Comparing RSA and ECC Cryptography

Feature	RSA	ECC
Security basis	Factoring large integers	Elliptic Curve Discrete Logarithm Problem
Key size for equivalent security	Much larger (e.g., 2048-bit)	Much smaller (e.g., 256-bit ECC \approx 3072-bit RSA)
Performance	Slower (esp. for signing & key exchange)	Faster, more efficient (smaller computations)
Bandwidth / Storage	Larger keys, larger signatures	Smaller keys & signatures (good for constrained devices)
Adoption	Very widely used historically	Increasingly used (TLS, cryptocurrencies, mobile, IoT)
Quantum resistance	Not resistant (Shor's algorithm breaks it)	Also not resistant (quantum breaks ECC too)



KMS Key Rotation

Cryptographic best practices discourage extensive reuse of encryption keys

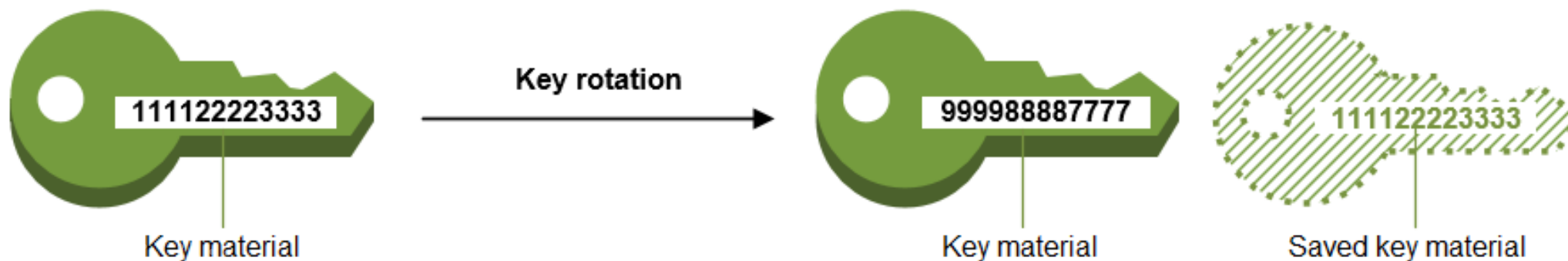
When you enable automatic key rotation for a KMS key, AWS KMS generates new cryptographic material for the KMS key every year

AWS KMS saves all previous versions of the cryptographic material in perpetuity so you can decrypt any data encrypted with that KMS key

AWS KMS does not delete any rotated key material until you delete the KMS key

KMS Key Rotation

- AWS KMS supports automatic key rotation only for symmetric KMS keys with key material that AWS KMS creates
 - **Automatic rotation is optional for customer managed KMS keys**
- Key rotation changes only the KMS key's key material, which is the cryptographic material that is used in encryption operations
- The KMS key is the same logical resource, regardless of whether or how many times its key material changes



Key ID = 1234abcd-12ab-34cd-56ef-1234567890ab

Key ID = 1234abcd-12ab-34cd-56ef-1234567890ab