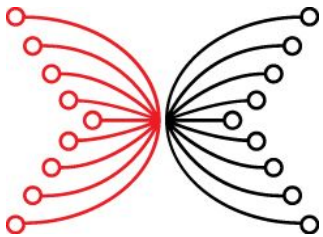
Lesson 7
Key management

A. Sorokin (Input | Output)

Self paced course

July 31, 2025



What is key management?

Key management discipline that encompasses all policies, procedures, and technologies used to protect cryptographic keys throughout their existence. It includes:

(i) **governance**: who can access keys, compliance requirement;

(ii) **security controls**: HSMs, access policies, auditing;

(iii) **key lifecycle**: how keys are generated, exchanged, stored, used, and destroyed.

Reasons for key management

A proper key management in a cryptosystem ensures

confidentiality, integrity, and availability

of encrypted data and protects a cryptosystem from:

(i) security breaches;

(ii) data leaks;

(iii) system compromises.

Poor key management

(i) If key management doesn't follow strict policies then keys are weak, exposed, or misused.

(ii) If there are no audits, then hackers can steal keys unnoticed.

(iii) If keys never expire, then cryptosystem is vulnerable to long-term attacks.

(iv) If there is no secure destruction, then deleted keys and data can be recovered by an adversary.

Real life aspects

Key management goes beyond pure mathematical practices, as it involves aspects of social engineering such as

- system policy;
- user training;
- organizational and departmental interactions;
- coordination between all of the above.

Key lifecycle

The **key lifecycle** is a subset of key management which defines the stages how a cryptographic key goes through from creation to destruction and ensures keys are handled securely at every phase.

Key lifecycle includes the following activities:

(1) generation
(2) exchange
(3) storage
(4) usage
(5) rotation
(6) revocation
(7) backup
(8) destruction

Key management vs key lifecycle

Key management answers the "what" and "why" questions

$$\textit{Key management} = \textit{Governance} + \textit{Technology} + \textit{Policies}$$

Key lifecycle answers the "how" and "when" questions

$$\textit{Key lifecycle} = \textit{Stages} + \textit{Execution}$$

Key management vs key lifecycle

The key lifecycle as a subset of key management is related to the latter in the following way.

Key Management	Key Lifecycle
Defines policies	Implements those policies in each phase
Chooses technology	Applies the tech at the storage phase
Sets compliance rules	Executes secure destruction phase

Key management vs key lifecycle

Enterprise Encryption System

Key management decision: we need to comply with PCI DSS, which requires key rotation every 90 days.

Key lifecycle execution:

- *generation*: automatically create new keys in an HSM;
- *rotation*: replace old keys every 90 days;
- *revocation*: disable compromised keys immediately;
- *destruction*: securely erase retired keys per NIST SP 800-88.

Types of keys

Different keys serve different purposes in cryptographic systems. Below are the most important types.

(1) **Symmetric keys** are used in symmetric cryptosystems, i.e. cryptosystems wherer the same key is used for encryption and decryption.

(2) **Asymmetric keys (public-private key pairs)** are key pairs, where one key encrypts and can be shared openly, and the other decrypts and is kept secret.

Types of keys

(3) **Session keys** are temporary keys used for a single communication session.

(4) **Ephemeral keys** are short-lived keys used for a single operation.

(5) **Master keys** are high-level keys used to derive or encrypt other keys.

(6) **Derived keys** are keys generated from a master key or password.

1. Key generation

The security of cryptographic systems depends on strong key generation, which is ensured by

(i) **randomness requirements**: keys must be generated using *cryptographically secure pseudo-random number generators* (CSPRNGs), since weak randomness leads to predictable keys;

(ii) **key length**: key length of 128 bits or more for symmetric encryption and at least 2048 bits for asymmetric one;

(iii) **following standards and best practices** such as key generation guidelines.

2. Key exchange

Key exchange (distribution) is a sharing keys securely with authorized parties. The most common methods are:

- *manual distribution* (e.g., pre-shared keys (PSK) for Wi-Fi);
- *symmetric key exchange*:
  (i) key encapsulation mechanisms (KEM) (e.g., RSA-KEM);
  (ii) key wrapping (e.g., RSA key wrapping);
- *asymmetric key exchange*:
  (i) key exchange protocols (e.g., Diffie-Hellman, ECDH);
  (ii) public key infrastructure (PKI) (e.g. certificate X.509).

Secure key exchange reqires to use authenticated channels and avoid key exposure in logs or memory.

3. Key storage

Key storage refers to the secure safekeeping of cryptographic keys when they are not in use. Among the most popular options are

- *hardware security modules (HSM)* are tamper-resistant devices for key storage and cryptographic operations (e.g. Luna PCIe, Azure Dedicated HSM, Google Cloud HSM);
- *key management services (KMS)* are cloud-based solutions (e.g. AWS KMS);
- *secure enclaves (trusted execution environments (TEE))* are isolated CPU environments where keys are protected even if OS is compromised (e.g. Intel SGX, ARM TrustZone);
- *software-based key vaults* provide access control, auditing, and key lifecycle management (e.g. HashiCorp Vault, Azure Key Vault);
- *key splitting* is a process of dividing a key into multiple shares (e.g. 3-of-5 recovery scheme via Shamir's secret sharing).

4. Key usage

Key usage refers to the controlled and secure employment of cryptographic keys for specific operations, such as:

- encryption and decryption (e.g. AES-GCM, RSA-OAEP, ECC);
- digital signatures (e.g. RSA-PSS, ECDSA);
- key wrapping (e.g. RFC 3394, ECIES);
- key derivation (e.g. HKDF, PBKDF2) ;
- authentication (e.g. SSH Key Pairs).

4. Key usage

Key usage prioritaze the following principles:

(i) *separation of duties*: different keys serve different purposes;

(ii) *least privilege*: keys are available to users according to their access level;

(iii) *auditability*: logging and monitoring actions to detect misuse.

5. Key rotation

Key rotation (replacement) is a periodical keys replacement to prevent long-term key exposure. Here are common methods:

- *time-based rotation* (e.g. every 90 days);
- *event-based rotation* (e.g. suspected compromise, employee departure);
- *usage-based rotation* (e.g. rotate after a certain number of encryptions);
- *key rolling* (e.g. old and new keys coexist temporarily to avoid service disruption);
- *forward secrecy* (e.g. use ephemeral keys to ensure the past sessions remain secure if a key is compromised);
- *automated rotation* (e.g. AWS KMS rotate keys automatically once per year).

6. Key revocation

Key revocation is the process of invalidating a cryptographic key before its expiration date to prevent misuse (e.g., after a breach or employee departure). Common revocation methods are:

- *certificate revocation lists (CRL)* are publicly distributed lists of revoked certificates (e.g. HTTPS certificates revoked due to private key leaks);
- *online certificate status protocol (OCSP)* perform real-time checks to verify if a certificate is revoked (e.g. web browsers use OCSP to validate TLS certificates);

6. Key revocation

• *key invalidation in HSMs/KMS* is realized as an immediate disablement of keys in hardware/cloud systems (e.g. AWS KMS marks keys as "disabled");

• *short-lived credentials* set short expiration times to auto-revoke (e.g. JWT, API Keys);

• *blocklisting in identity systems* explicitly denies access to specific keys, tokens, or credentials in identity and access management (IAM) systems (e.g. HSM key deactivation).

7. Key backup and recovery

Key backup ensures cryptographic keys are securely stored and can be restored if lost due to hardware failure or human error, while **key recovery** refers to the process of retrieving and reinstating those keys when needed.

7. Key backup and recovery

Key Backup Methods:

(i) *secure storage in HSMs and key vaults* offers automated backups (e.g., Thales Luna, AWS CloudHSM, Google Cloud KMS);

(ii) *Shamir's secret sharing* splits a key into multiple shares (e.g. enterprise key escrow for disaster recovery via 3-of-5 scheme);

(iii) in *encrypted backups* keys are encrypted with a strong passphrase or another key (e.g. PGP);

(iv) *paper/metal backups (cold storage)*: printed QR codes or seed phrases, steel plates for fire and water resistance (e.g., cryptocurrency wallets).

7. Key backup and recovery

Key Recovery Methods:

(i) *HSM key restoration* provides secure key recovery (e.g., Thales SafeNet);

(ii) *cloud KMS* allows to retain deleted keys for a grace period (e.g., 30 days);

(iii) *multi-party recovery (MPR)* requires multiple authorized personnel to reconstruct a key (e.g. corporate master keys);

(iv) *recovery from splits* in Shamir's secret sharing (e.g., 3 out of 5 admins must collaborate).

8. Key destruction

Key destruction (crypto-shredding) ensures that keys (and encrypted data) are irrecoverable when no longer needed.

(i) *secure deletion methods* like overwriting storage via multiple passes or HSM key zeroization (physically erases keys from hardware);

(ii) *cryptographic erasure* means deleting the encryption key which makes data permanently unrecoverable (e.g., AWS S3 bucket key deletion);

Key lifecycle example

TLS certificate lifecycle

1. Generation: CA creates a private key and CSR.
2. Distribution: issues a signed certificate.
3. Storage: the private key stored in HSM.
4. Usage: deployed on a web server for HTTPS.
5. Rotation: renewed every 1-2 years (Let's Encrypt auto-renewal).
6. Revocation: if the private key is leaked, revoke via CRL/OCSP.
7. Backup: recover the private key stored in HSM.
8. Destruction: securely delete expired keys.