Core Cryptography and Key Management

Course Overview

Domains Covered: Security Architecture and Engineering

Course Objective: The art of protecting data using cryptographic techniques and learn how to manage cryptographic keys throughout their entire lifecycle. This course provides practical, hands-on skills for security professionals and IT practitioners working with encryption systems in real-world environments.

Part A: Understanding Cryptographic Fundamentals ⁉️

1 Choosing Strong Cryptography

What This Means

you shouldn't use weak cryptography to protect important data. Strong cryptography means using battle-tested, industry-approved algorithms with keys that are long enough to resist attacks from both current and emerging computational threats.

How to Implement

Step 1: Select Proven Algorithms

- Use algorithms that security experts have tested
- Examples include AES, RSA, and SHA-256
- Avoid proprietary or obscure algorithms without peer review

Step 2: Choose Appropriate Key Lengths

- **Minimum requirement:** 112-bits of effective key strength for existing systems
- **Recommended for new projects:** 128 bits or higher

Step 3: Stay Current

- Review and update your cryptographic standards
- Retire outdated algorithms
- Monitor industry advisories
- Monitor vulnerability disclosures

Even if attackers break through your firewalls, steal your backups, or compromise your servers, properly encrypted data remains completely unreadable without the correct keys.

2 Using Vetted Modules and Services

Leverage Operating System Features

- Use built-in encryption services from the OS
- Implement platform-provided identity management system
- Benefit from security updates and patches

Use Established Frameworks

- Web applications: TLS/SSL libraries like OpenSSL
- Password hashing: bcrypt, Argon2, or PBKDF2
- Encryption: Bouncy Castle or Cryptography.io

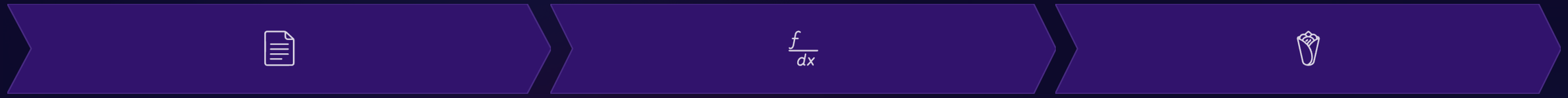
⚠ Avoid Custom Implementations

- Never create your own encryption algorithms
- Don't modify existing algorithms unless you're an expert
- Stick to standardised, peer-reviewed solutions

Security experts spend years perfecting cryptographic implementations. Using their work reduces your workload dramatically and prevents costly mistakes that could leave your data vulnerable. Even minor implementation errors can compromise security completely.

3 Understanding Cryptographic Constraints

Not all data needs the same level of protection. Match your security effort to the value and lifespan of the data you're protecting. This practical approach ensures efficient resource allocation whilst maintaining robust security posture.



Step 1: Classify Your Data

Identify how long data remains valuable (days, months, years, decades) and determine the sensitivity level (public, internal, confidential, top secret).

Step 2: Calculate Protection Requirements

Short-term data (valid for hours/days): Standard encryption may suffice. Long-term data (valuable for years): Use stronger encryption and longer keys.

Step 3: Perform Cost-Benefit Analysis

Protection cost shouldn't exceed data value, but protection must last until data becomes worthless to potential adversaries.

Part B: Cryptographic Key Lifecycle Management

Restricting Access to Cryptographic Keys

Keys are the crown jewels of your security system. Limit who can access them to an absolute minimum. This principle of least privilege is fundamental to maintaining cryptographic security across your organisation.

01

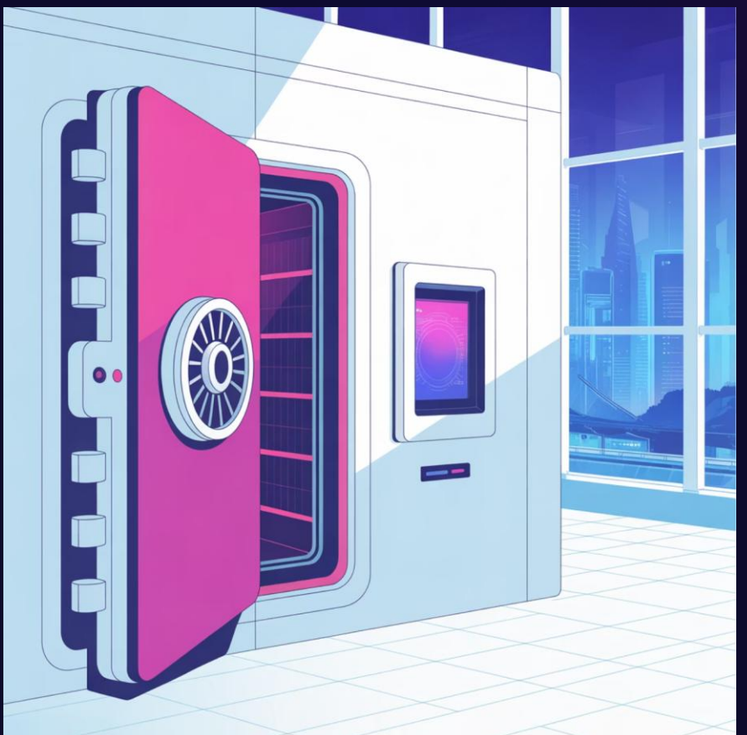
Identify Key Custodians

Designate specific, trusted individuals as key custodians. Keep this list as short as possible (need-to-know basis) and document who has access and why.

02

Secure Key Storage

Store keys in Hardware Security Modules (HSMs) when possible. Use key vaults or key management services. Never store keys in source code, configuration files, or databases without encryption.



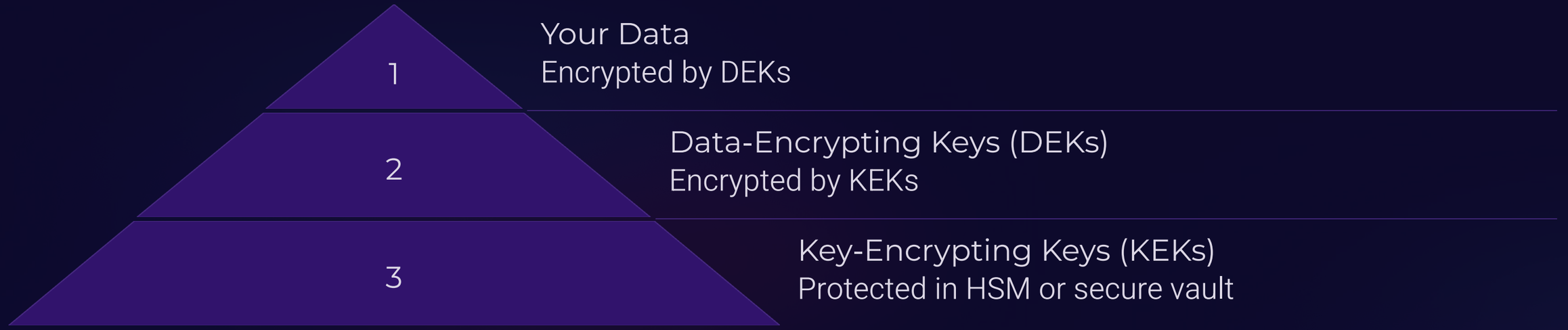
Implement Monitoring

Log all key access attempts. Set up alerts for unauthorised access. Regularly audit key usage patterns and investigate anomalies promptly.

Keys are the master passwords to your encrypted data. If an attacker gets your keys, all encryption becomes useless. Key-encrypting keys (the keys that protect other keys) are especially critical—they're like master keys that open all doors in your security infrastructure.

5 Implementing Key Separation and Strength

Use a layered approach: protect your encryption keys with even stronger keys, and keep them separated. This defence-in-depth strategy significantly reduces the risk of complete cryptographic compromise.



Step 1: Establish Key Hierarchy

- **Data-Encrypting Keys (DEKs):** Encrypt your data
- **Key-Encrypting Keys (KEKs):** Encrypt your DEKs
- Ensure KEKs are at least as strong (preferably stronger) than DEKs

Step 2: Physical/Logical Separation

- Store KEKs in different locations than DEKs
- Use different systems or HSMs for each key type
- Never store KEKs alongside the data they protect

Step 3: Access Control Separation

- Different personnel, Different KEKs versus DEKs
- Require additional authentication for KEK access
- Implement time-based access restrictions

This creates a layered defence. Even if attackers compromise your data storage and steal the DEKs, they still can't decrypt anything without the separately stored KEKs. It's like having a locked box inside another locked safe—double protection that significantly increases the difficulty of successful attacks.

Controlling Cleartext Key Handling

When keys must be handled manually in unencrypted form, use split knowledge and dual control to prevent any single person from having complete access. This is one of the most critical controls in cryptographic key management.

1

Implement Split Knowledge

Divide key components amongst multiple people. No single person should possess all components. Typically split into 2-3 parts minimum to ensure adequate separation of duties.

2

Establish Dual Control

Require at least two authorised individuals to be present. Both must perform their part of the operation simultaneously. Neither should have access to the other's component at any time.

3

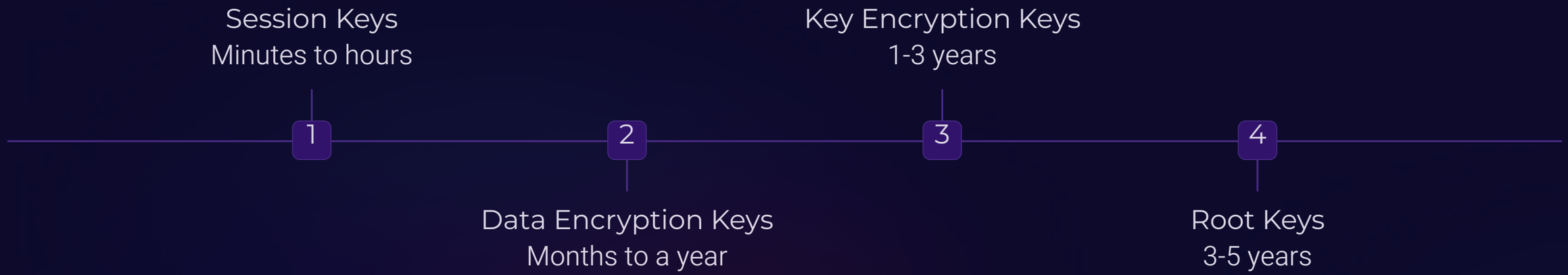
Document Procedures

Create detailed procedures for key generation, storage, and use. Require signatures from both parties. Maintain comprehensive audit logs of all dual-control operations.

This prevents insider threats and accidental key exposure. No single individual can steal, misuse, or accidentally expose the complete key. It's the security equivalent of requiring two keys to launch nuclear missiles—a tried and tested approach to preventing unauthorised actions.

7 Managing Cryptoperiods and Key Changes

Keys shouldn't last forever. Regularly rotate them to reduce the risk of compromise and stay ahead of evolving attacks. Understanding and implementing proper key rotation schedules is essential for maintaining long-term cryptographic security.



Implement Rotation Procedures

- Set calendar reminders for key rotation
- Automate rotation when possible using key management systems
- Test rotation procedures regularly in non-production environments
- Re-encrypt data with new keys before destroying old keys
- Maintain key version tracking for audit purposes

Handle Compromise Situations

- Create an emergency key replacement procedure
- Immediately revoke compromised keys across all systems
- Investigate the scope and impact of the compromise
- Re-encrypt all affected data with new keys
- Notify stakeholders if required by law or regulation

The longer a key is used, the more opportunities attackers have to capture and analyse encrypted data. Regular rotation limits the damage from any single key compromise. Additionally, as computers become more powerful, older keys become easier to crack—rotation keeps you ahead of the technological curve and emerging threats.

Documenting Cryptographic Architecture

For service providers especially, maintain comprehensive documentation of your entire cryptographic setup and keep production and test environments completely separate. Good documentation is not just best practice—it's essential for security incident response and compliance.

Create Documentation Inventory

Document the following:

- All encryption algorithms in use (e.g., AES-256, RSA-2048)
- Protocols employed (e.g., TLS 1.3, IPsec)
- Key strengths and types for each system
- Key usage purposes and data classifications
- Key storage locations and backup procedures
- Key custodians and access controls with responsibilities

Environment Separation

- Never use the same keys in production and test/development
- Generate separate keys for each environment
- Clearly label and colour-code environments
- Implement technical controls to prevent key migration between environments
- Use different key management systems when possible

Regular Reviews

- Quarterly review of cryptographic inventory
- Annual comprehensive audit by security team
- Update documentation immediately after changes
- Track deprecated algorithms and plan migrations
- Review access logs and custodian assignments

> Imagine discovering a vulnerability in an algorithm you use. With good documentation, you can immediately identify every place it's used and create a remediation plan. Without documentation, you're flying blind and putting your organisation at serious risk.

Part C: Applied Data Protection Using Cryptography

Encrypting Data at Rest

Protect stored data by encrypting it on servers, databases, applications, and end-user devices. Data at rest encryption is your last line of defence when physical or logical security controls fail.

Server and Database Encryption

- Enable Transparent Data Encryption (TDE) on databases
- Use full-disk encryption on servers
- Encrypt sensitive columns or fields at application layer
- Implement encryption for backups and archives

End-User Device Encryption

- Windows: Enable BitLocker
- macOS: Enable FileVault
- Linux: Use LUKS or dm-crypt
- Mobile devices: Enable built-in encryption (usually default)

Cloud Storage Encryption

- Use server-side encryption for cloud storage
- Consider client-side encryption for extra sensitive data
- Ensure encryption keys are properly managed
- Verify provider's encryption practices

CUI Protection

- Follow government-specific guidelines
- Maintain compliance documentation

This protects your data even if physical security fails. If someone steals a laptop, server, or backup tape, they can't read the encrypted data without the keys. Application-layer encryption ensures data remains protected even if the database or filesystem is compromised through SQL injection or other attacks.

10 Encrypting Data in Transit

Protect data whilst it travels across networks using secure protocols. Network traffic encryption is essential in today's interconnected world where data constantly moves between systems, users, and organisations.

Web Traffic

- Use TLS 1.2 or higher (preferably TLS 1.3)
- Implement HTTPS for all websites and web applications
- Use strong cipher suites and disable weak ones
- Obtain valid SSL/TLS certificates from trusted authorities

Remote Access

- Use SSH (Secure Shell) for remote server management
- Implement VPNs for remote user access
- Use SFTP/SCP instead of FTP for file transfers
- Disable legacy protocols like Telnet

Email Security

- Use TLS for SMTP connections
- Consider S/MIME or PGP for end-to-end email encryption
- Encrypt email attachments containing sensitive data
- Implement DMARC, SPF, and DKIM

Payment Card Data

- Always encrypt Primary Account Numbers (PAN) during transmission
- Use point-to-point encryption (P2PE) when possible
- Never transmit cardholder data over unsecured channels

Internal Networks

- Don't assume internal networks are safe
- Encrypt sensitive traffic even within your organisation
- Use IPsec or TLS for internal application communications
- Implement network segmentation with encryption

Attackers can intercept unencrypted traffic and steal credentials, payment data, or confidential information. Encryption makes intercepted data useless to attackers—they capture gibberish instead of valuable information that could compromise your organisation.

1 Protecting Authentication Factors

Ensure passwords, PINs, biometric data, and other authentication credentials are always encrypted during transmission and storage. Authentication factors are the gateway to your systems—their protection is paramount.

Password Storage

Never store passwords in plain text. Use strong hashing algorithms (bcrypt, Argon2, PBKDF2). Add unique salts to each password. Consider using adaptive hashing that increases work factor over time.

Transmission Protection

Always use TLS/HTTPS when transmitting credentials. Never send passwords via email or unencrypted chat. Use POST requests (not GET) for submitting credentials to prevent logging in URLs.

Multi-Factor Authentication (MFA)

Encrypt token seeds and backup codes. Protect biometric templates with strong encryption. Secure authentication app data on devices using device encryption capabilities.

Session Management

Encrypt session tokens. Use secure, HttpOnly, and SameSite cookies. Implement secure session storage. Enforce session timeouts and re-authentication for sensitive operations.

Privileged Account Protection

Use privileged access management (PAM) solutions. Encrypt privileged credentials at rest. Require additional authentication for privileged access. Implement just-in-time access provisioning.

Many breaches occur because stolen password databases contained plain-text or weakly hashed passwords. Proper cryptographic protection of authentication factors is your last line of defence against credential theft and unauthorised access.