

Encryption Cybersecurity: Concepts, Purpose, and Examples



Explain what encryption is and why it matters in cybersecurity



See how encryption shows up in real systems and daily workflows



Audience: IT staff, security beginners, developers, managers, educators



No math proofs required — focus on concepts and decisions



Three anchors: core concepts, strategic purpose, practical examples



Scope: encryption is one data-protection control, not all of cybersecurity



Path: vocabulary → hands-on examples → crypto-agility roadmap



Background: Why Encryption Is a Baseline Control

Encryption



Hashing



Encoding



Tokenization



Digital Signatures



- Encryption, hashing, encoding, and tokenization solve different problems
- Confidentiality first; integrity via AEAD tags and digital signatures
- Frameworks require appropriate encryption — no single mandated cipher
- GDPR Art. 32, ISO 27001 A.8.24, SOC 2 CC6.1/CC6.7, HIPAA, PCI DSS Req. 3/4
- "Encrypted" is a system property: key management decides if it protects

Compliance Frameworks

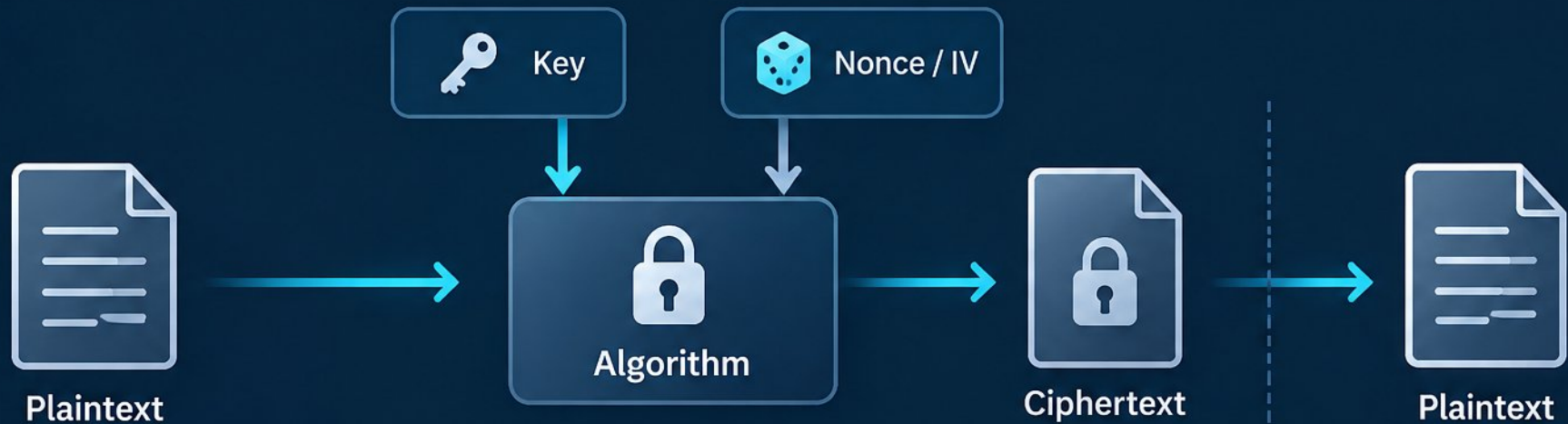
GDPR

ISO 27001

HIPAA

PCI DSS

Core Concepts: Keys, Ciphers, and Modes



Symmetric (AES)

AES-128/192/256



Asymmetric (RSA / ECDSA)

RSA ≥ 2048 -bit,
ECDSA P-256/P-384



Hashing (SHA-2 / SHA-3)

SHA-2/SHA-3,
Argon2id

- Plaintext, ciphertext, key, algorithm, cipher suite — the course vocabulary
- Symmetric: AES-128/192/256; asymmetric: RSA ≥ 2048 -bit, ECDSA P-256/P-384
- Hashing (SHA-2/SHA-3, Argon2id) is one-way, not encryption
- Block vs. stream ciphers; modes change security: GCM/CCM/XTS-AES accepted
- ECB retired as a confidentiality mode; CBC conditional; FF3 disallowed
- Key strength depends on generation randomness, nonces, and IVs — not algorithm alone

Purpose: Confidentiality, Compliance, and Business Value

CONFIDENTIALITY, INTEGRITY, AND AUTHENTICITY



Confidentiality

Keep data private



Integrity

Help ensure data hasn't been altered



Authenticity

Verify source and trust

COMPLIANCE DRIVERS



HIPAA

GDPR

BUSINESS VALUE

IBM 2025:
global average breach

\$4.44M

Encryption mitigates
roughly \$208K



Protects confidentiality at rest, in transit, and in use; AEAD adds integrity



Regulators expect "appropriate" encryption, not a named cipher



HIPAA and GDPR safe harbors can cut breach-notification duties when keys are safe



Encryption doesn't stop endpoint compromise, misconfiguration, or exposed keys



IBM 2025: global average breach \$4.44M; encryption mitigates roughly \$208K

Encryption in Transit: TLS 1.3 and Modern Protocol Hygiene

SECURE COMMUNICATIONS FOR A MORE RESILIENT TOMORROW



Use Modern,
Secure Suites

- ✓ AEAD
- ✓ ECDHE
- ✓ SHA-256



Disable Insecure
and Retired Protocols

- ✗ TLS 1.0/1.1
- ✗ SSLv2/v3
- ✗ RC4
- ✗ 3DES
- ✗ NULL
- ✗ Anonymous suites

- TLS delivers confidentiality, integrity, and server authentication; mTLS authenticates the client
- Default to TLS 1.3; allow TLS 1.2 only with AEAD suites and ECDHE forward secrecy
- Disable TLS 1.0/1.1, SSLv2/v3, RC4, 3DES, NULL, and anonymous suites
- RFC 9852: new protocols must require TLS 1.3; TLS 1.2 is in feature freeze
- Use HSTS; SHA-256 certificate signatures, 2048-bit RSA minimum, and OCSP stapling
- Disable 0-RTT unless the application handles replay risk

Encryption at Rest:

Disk, Database, Object Storage, and Backups

Full-disk encryption



Full-disk encryption (BitLocker, FileVault, LUKS) is mandatory for laptops, mobiles, backups

Databases



Databases: TDE at storage layer plus field-level encryption for sensitive columns

Cloud storage



Cloud storage: server-side encryption is on by default; CMEK/BYOK adds control when needed

Backups



Envelope encryption: unique data keys wrapped by a KMS/HSM key encryption key



Recovery reality: disabled or destroyed KMS keys can make data and backups unreadable.

Encryption in Use and Key Management Systems



Data in use is hardest:
confidential computing,
enclaves, homomorphic
encryption



Full key lifecycle:
generation, distribution,
storage, rotation, revocation,
archival, destruction



Envelope encryption:
DEKs wrap data, KEKs stay
in KMS/HSM



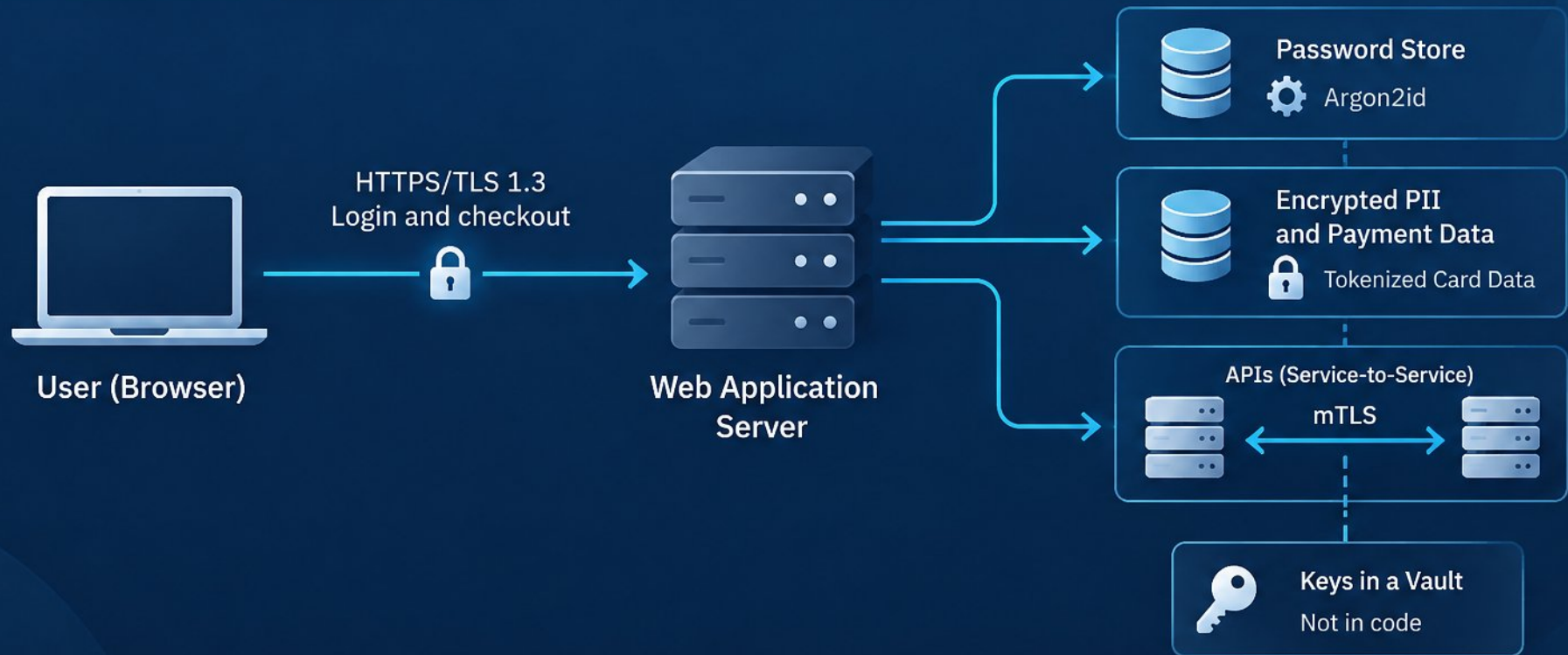
Choose by threat model:
HSM vs. TPM vs. software
keystore



**Separation of duties, dual
control, immutable audit logs
for every key operation**



Practical Example: Securing a Web Application End to End



- HTTPS/TLS 1.3 for login and checkout; Secure cookies and HSTS
- Passwords hashed, never encrypted: Argon2id 19 MiB, t=2, p=1
- Salts per user; optional pepper for defense in depth
- SHA-256 banned for passwords — GPUs guess ~22 billion per second
- Encrypt PII and payment data; tokenize card data to cut PCI DSS scope
- APIs: mTLS service-to-service; keys in a vault, not in code

Practical Example: Enterprise Data-at-Rest and Cloud KMS

1



- Enforce full-disk encryption on laptops via MDM; verify with BitLocker and Intune logs

2



- Use cloud KMS (AWS, Azure, Google) with keys in a separate project or account
- Set a 30-day minimum destruction window; alert on key deletion and policy changes

3



- Rotate keys annually minimum; more often for sensitive workloads or after compromise



Do this



- Stolen encrypted laptop earns safe harbor



Never do this



- Backup with co-located key does not

Practical Example: Secure Communication and Encrypted Messaging

TLS (Hop-by-Hop)

Protects the connection between each hop



Provider-Mediated Messaging

Messages encrypted with the provider



End-to-End Encryption (MLS)

Message encrypted end to end



Email Encryption in Practice

Different approaches for different needs



S/MIME

Dominates enterprise email



PGP/OpenPGP

Persists in technical communities



Password Portal

Option for recipients outside encrypted paths



Metadata still leaks who talked to whom, and when

Key Takeaways

- TLS protects the hop; E2EE protects the message end to end
- RCS uses MLS, giving forward secrecy and post-compromise security
- S/MIME dominates enterprise email; PGP/OpenPGP persists in technical communities
- Gateway decryption reintroduces EFAIL-class attacks and exposes plaintext
- Choose by recipient path: same provider, PGP, S/MIME, or password portal
- Metadata still leaks who talked to whom, and when



Implementation Pitfalls and Encryption Misconfigurations

SECURE
DESIGN
STRONGER
TOMORROW

Reused Nonce

Never do this



Do this



ECB Pattern Leakage

Never do this



Do this



Hardcoded Secret in Source Code

Never do this

```
const  
API_KEY =  
"hardcoded_secret"
```



Key Stored Beside Data

Never do this



Do this



Downgrade Attack

Never do this



Do this



TLS 1.3 0-RTT Replay

Never do this



Do this



Cryptographic Inventory (CBOM)

You cannot migrate what you have not inventoried



Validate with TLS scanners (testssl.sh, sslyze), secret detection, and CI/CD baselines

- Weak keys, reused nonces, and retired modes like ECB break approved algorithms
- Hardcoded secrets or keys stored beside data defeat encryption entirely
- Downgrade attacks, disabled certificate validation, missing HSTS, 0-RTT replay risk
- Cryptographic inventory (CBOM): you cannot migrate what you have not inventoried
- Validate with TLS scanners (testssl.sh, sslyze), secret detection, and CI/CD baselines



FEWER
MISTAKES
STRONGER
SYSTEMS

Key Management and Operational Practices



Governing references:

NIST SP 800-57 Part 1 (draft Rev. 6),
SP 800-133, SP 800-131A



Cryptoperiods: define and document them;
PCI DSS requires rotation at end of cryptoperiod



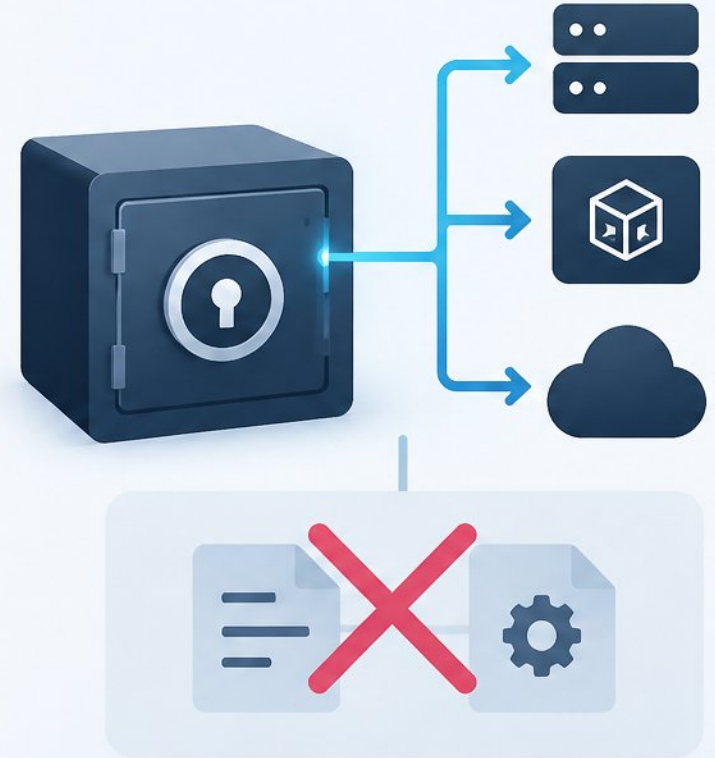
Use vaults and short-lived credential injection —
not environment variables or config files



Audit evidence: key inventory, access control
records, FIPS certificates, lifecycle runbooks



“We use AES” is not an answer — prove custody,
rotation, and control to auditors



Crypto Agility and Post-Quantum Migration Planning



-  “Harvest now, decrypt later”: adversaries collect encrypted traffic for future quantum decryption
-  Mosca's theorem: migrate before X (data lifetime) + Y (migration time) exceeds Z (CRQC arrival)
-  Plan around FIPS 203 (ML-KEM), 204 (ML-DSA), 205 (SLH-DSA); HQC added as future KEM
-  US federal deadlines: PQC key establishment by Dec 31, 2030; PQC signatures by Dec 31, 2031
-  TLS 1.3 required by Jan 2, 2030 — foundational for hybrid key exchange
-  Crypto agility: no hardcoded algorithms, carry key identifiers, run hybrid/dual-stack deployments



Assessing, Teaching, and Applying Encryption by Role



Self-assessment

- Self-assessment: inventory crypto, classify data, verify TLS, confirm FDE, test recovery



Developers

- Developers: Argon2id for passwords, maintained libraries, never roll your own, verify before use



Managers

- Managers: encryption cuts breach costs ~\$208K — enables safe cloud and AI adoption



Educators

- Educators: teach with real artifacts — TLS scans, KMS policies, OWASP cheat sheets



- Stay current: NIST CSRC, OWASP Cheat Sheets, vendor advisories, CISA/NSA KMS guidance

Closing: Key Takeaways, Resources, and Next Steps

Stronger cryptography. Safer systems. A more resilient tomorrow.

Five Takeaways

-  **name the property**
-  **use current algorithms**
-  **protect keys**
-  **prove controls**
-  **plan agility**

Common traps

-  **ECB**
-  **MD5/SHA-1**
-  **TLS 1.0/1.1**
-  **RSA under 2048 bits**
-  **hardcoded secrets**

Key resources

-  **NIST SP 800-57**
-  **SP 800-131A**
-  **SP 800-52 Rev. 2**
-  **FIPS 203/204/205**
-  **OWASP Cheat Sheets**

Next steps

-  **Developers**
ship Argon2id and TLS 1.3
-  **Managers**
commission a crypto inventory and agility roadmap



Open Q&A — revalidate this training as standards and threats evolve

PersonWise.

PersonWise • AI digital-human course platform



Scan the code or click anywhere to watch this course live, with voice Q&A

personwise.ai/t/6f08be3f/public