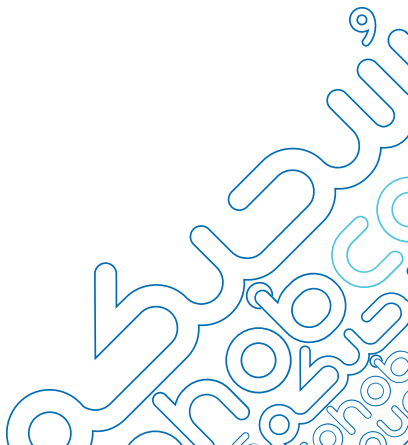
Encryption in Cloud Environments

Securing Data Across Cloud Infrastructures

BUTHAINA AL-MASHRAQI | Security Engineer

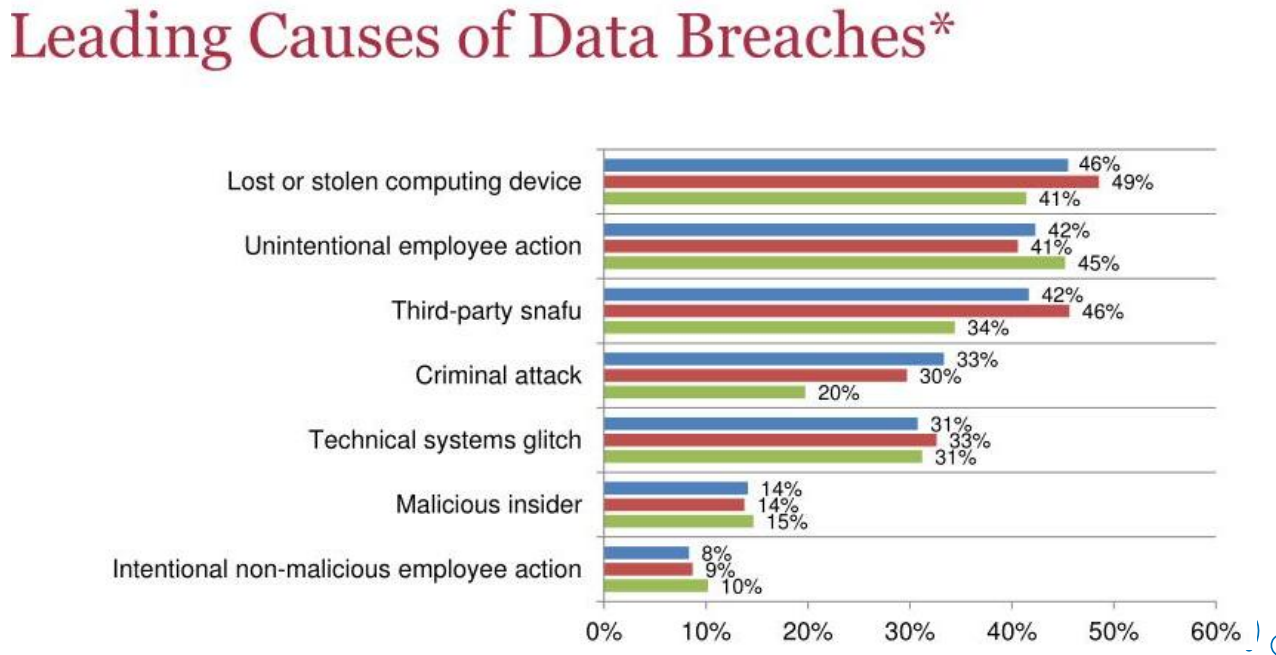
Agenda

- Why Encrypt in the Cloud?
- The Three States of Data
- Core Encryption Technologies
- Cloud Provider Models (AWS, Azure, GCP)
- Bring Your Own Key (BYOK)
- Transparent Data Encryption (TDE)
- Tokenization & Format-Preserving Encryption
- Runtime Encryption
- Post-Quantum Cryptography (PQC)
- Emerging Trends
- Best Practices & Roadmap



Why Encrypt in the Cloud?

- 85% of organizations use multiple clouds.
- Data is everywhere: cloud, edge, mobile
- Shared responsibility model: You own the data
- Risks:
  - Misconfigured storage
  - Insider threats
  - Compliance failures (GDPR, HIPAA, PCI DSS)



The Three States of Data

State	Protection
At Rest	AES-256, TDE, HSM
In Transit	TLS 1.3, IPsec
In Use	Runtime Encryption, Confidential Computing

! Traditional encryption stops at "at rest" — but data in use is most vulnerable.

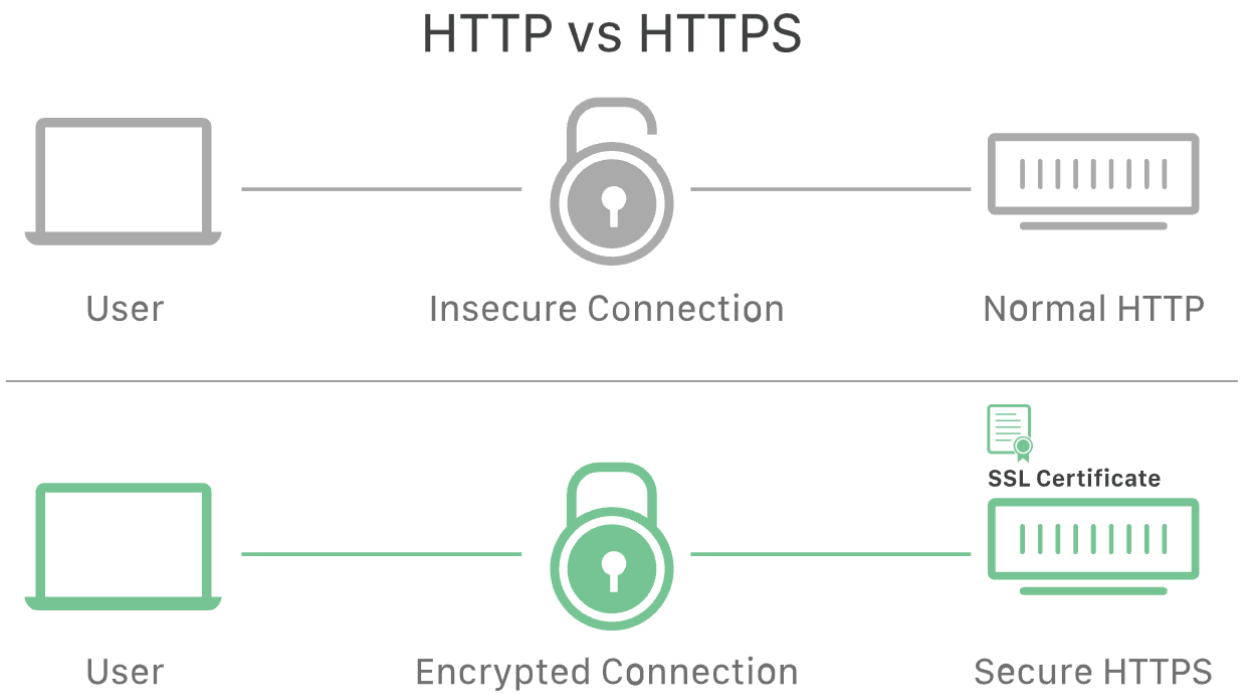
Encryption at Rest

- Protects: Databases, VM disks, backups, object storage
- Methods:
  - Full Disk Encryption (FDE)
  - Transparent Data Encryption (TDE)
  - Object Storage Encryption (S3, Blob)

Encryption in Transit

- Protocols:
  - TLS 1.3 (HTTPS, APIs)
  - IPsec (VPNs)
  - MACsec (layer 2)
- Use Cases:
  - Web traffic
  - Microservices
  - Cloud-to-on-prem

Pro Tip: Use mTLS for zero-trust service communication



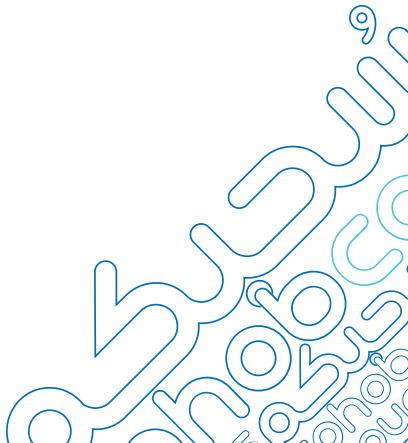
The Challenge of Data in Use

- When data is processed:
- It's decrypted in memory
- Vulnerable to:
  - Memory scraping
  - Hypervisor attacks
  - Malware
  - Insider access

- Traditional encryption does not protect data in use

Runtime Encryption & Confidential Computing

- Runtime Encryption: Keeps data encrypted during processing
- Confidential Computing: Uses Trusted Execution Environments (TEEs)
  - Intel SGX
  - Azure Confidential VMs
- Data decrypted only in secure enclave
- Even OS or cloud provider cannot see it

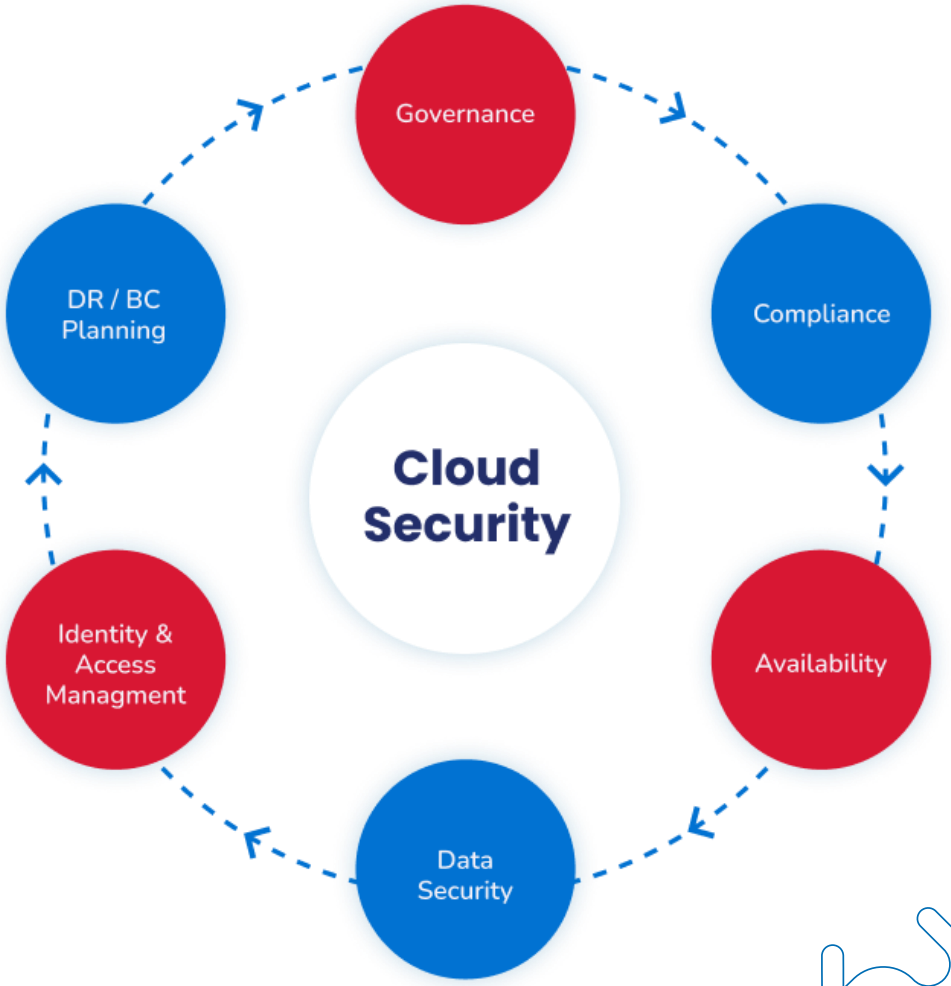


Core Encryption Technologies

Technology	Purpose	Use Case
AES-256	Symmetric encryption	Disk, file, DB
RSA / ECC	Asymmetric	Key exchange, signing
HSM	Secure key storage	FIPS 140-2 Level 3
KMS	Key lifecycle	Centralized control
PKI	Certificates	TLS, identity
Tokenization	Replace data	PCI DSS, PII

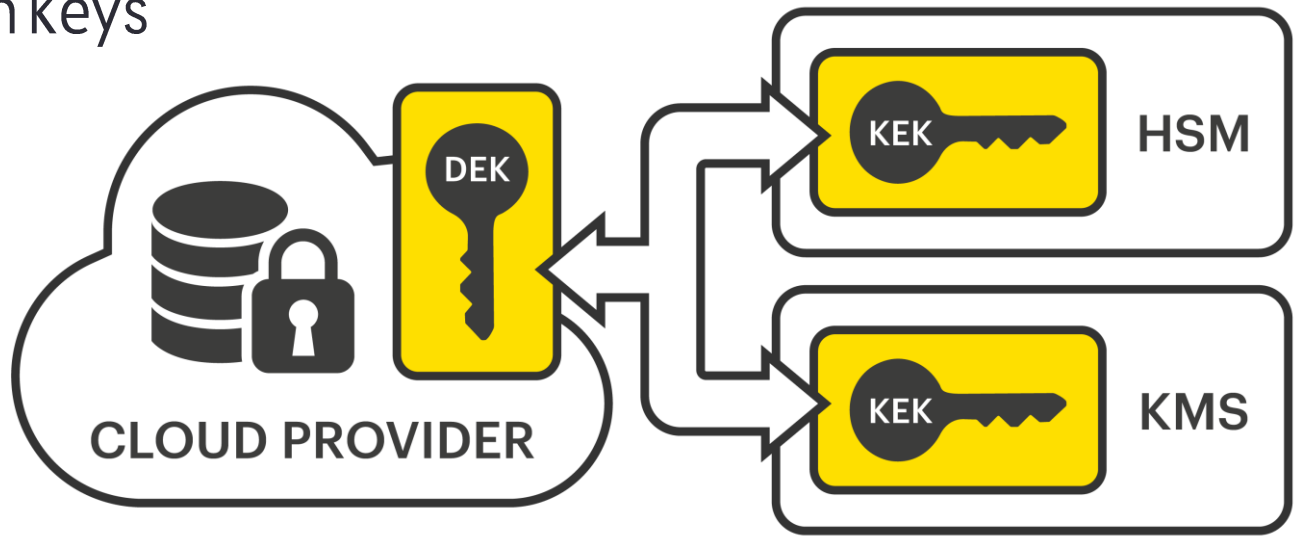
Cloud Provider Encryption Models

- Default encryption at rest
- Customer-managed keys via KMS
- HSM integration for high security
- Client-side encryption libraries for maximum control



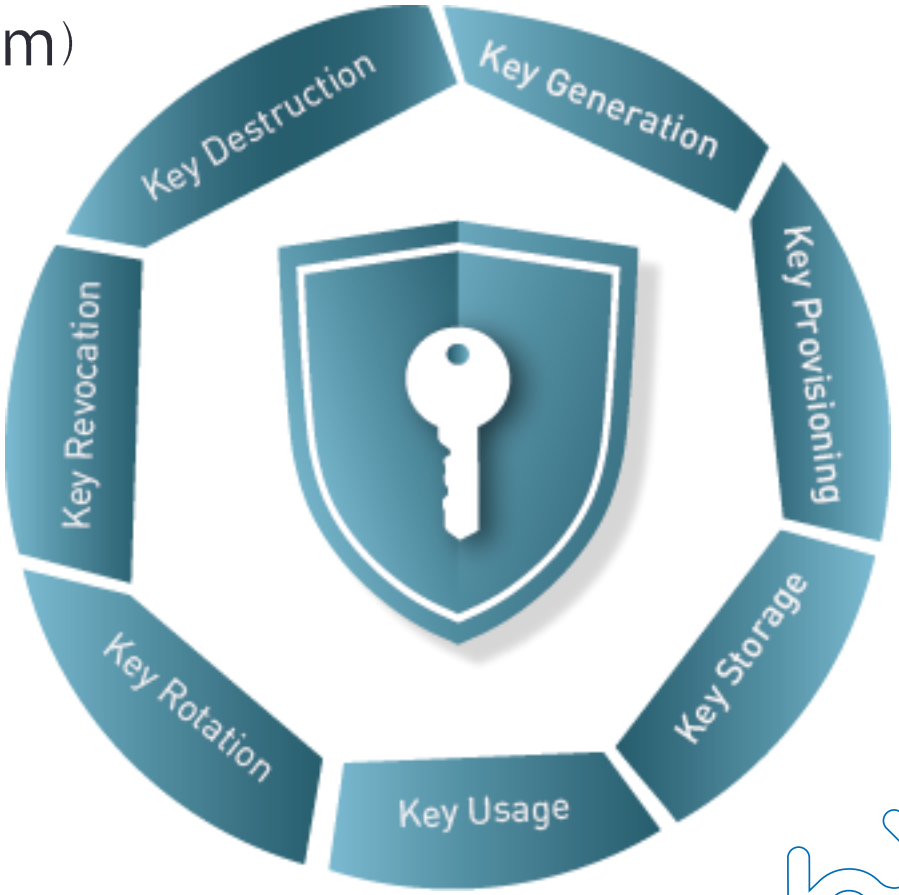
Bring Your Own Key (BYOK)

- Problem: Cloud providers manage keys → risk
- Solution: BYOK — import your own keys
- Benefits:
  - Full control
  - Prevents provider access
  - Meets compliance
- Supported by Sohobcom



Key Management Best Practices

1. Use a central KMS（A service provided by Sohobcom）
2. Enable key rotation
3. Apply least privilege
4. Audit all key usage
5. Never store keys in code or config



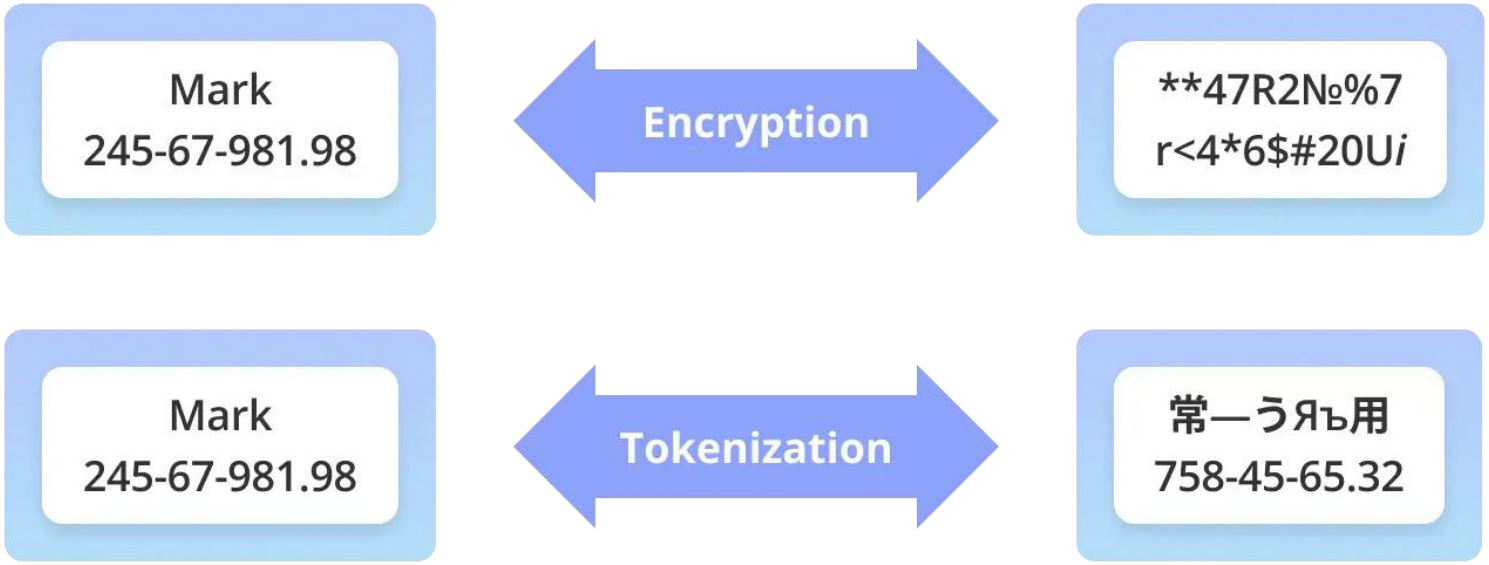
🔑 Golden Rule: Keys should never leave the vault.

Transparent Data Encryption (TDE)

- Encrypts DB files, logs, backups automatically
- No app changes needed
- Supported by: SQL Server, Oracle, MySQL, RDS, Azure SQL
- Pros: Easy to deploy
- Cons: Doesn't protect data in use

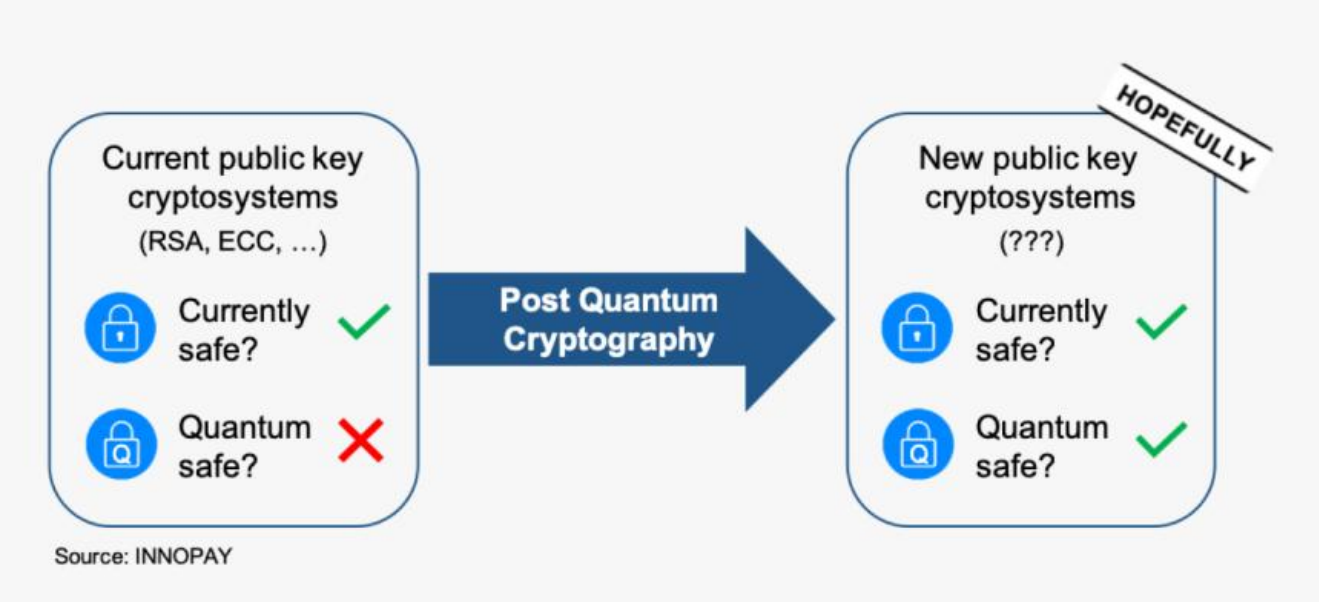
Tokenization

- Replace real data with fake tokens
- Token looks real but has no value
- Use Cases:
  - Credit cards (PCI DSS)
  - PII (SSN, email)
  - Test environments
- Benefit: Reduces compliance scope



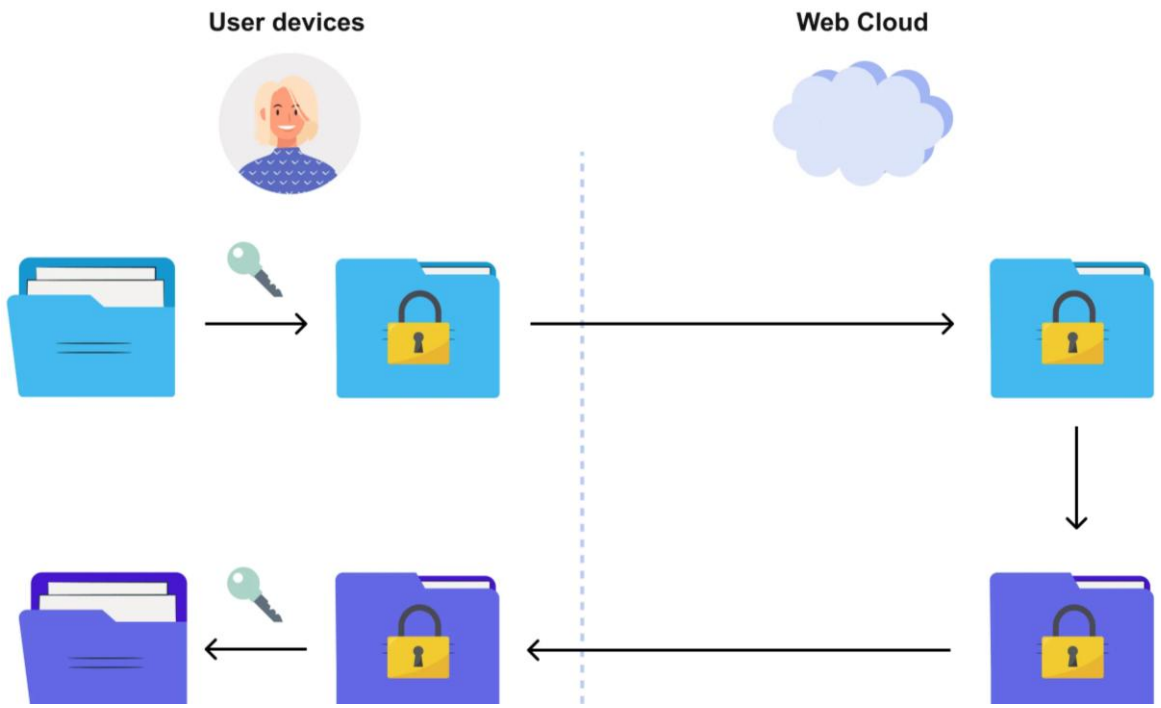
Post-Quantum Cryptography (PQC)

- Quantum computers can break RSA/ECC
- NIST standards:
  - CRYSTALS-Kyber (encryption)
  - Dilithium (signatures)
- What to do:
  - Inventory long-lived data
  - Test PQC
  - Plan hybrid transition



Homomorphic Encryption

- Compute on encrypted data without decryption
- Example: Analyze encrypted sales data
- Challenges:
  - Extremely slow
  - Experimental
- Future Use:
  - Secure AI/ML
  - Privacy-preserving analytics



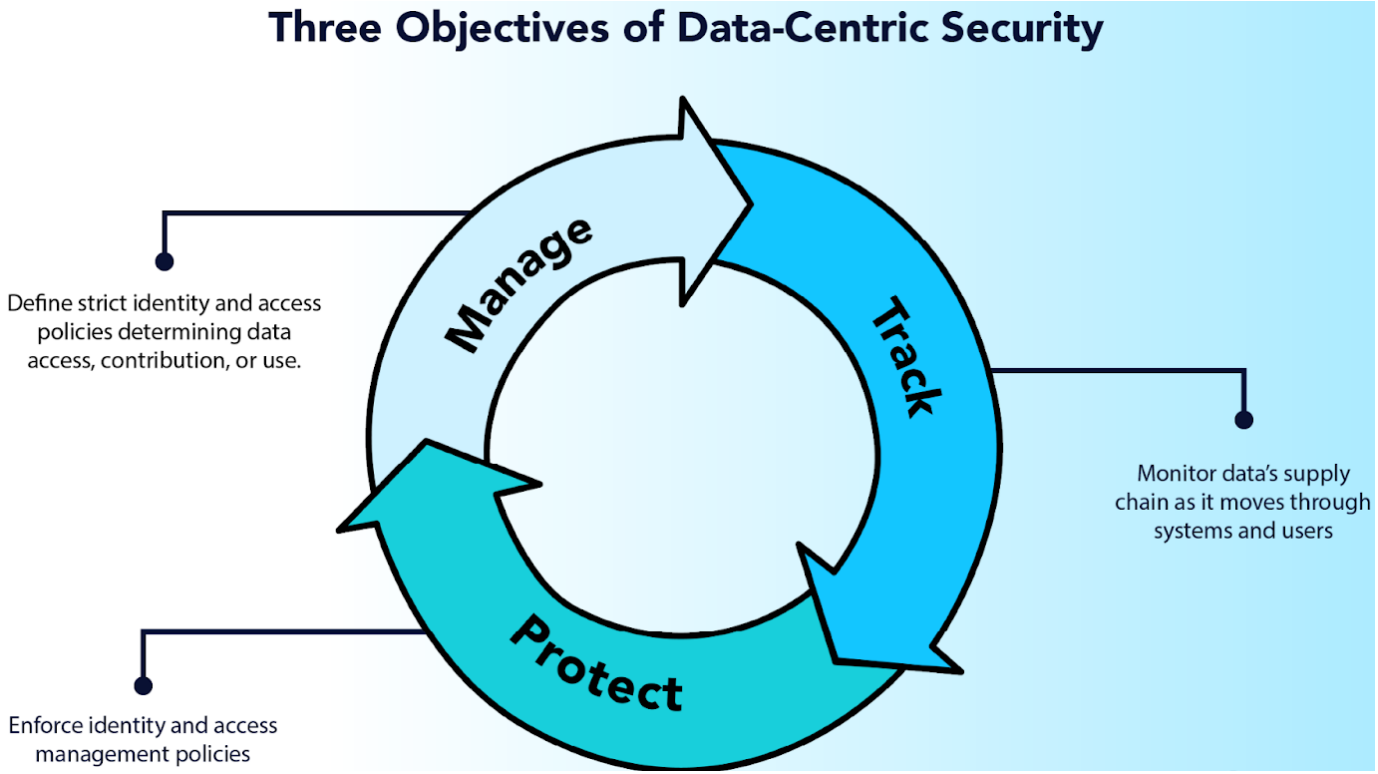
🔮 Not ready for production — but watch this space

Data-Centric Security

- Traditional:
  Firewall → Network → Host → App → Data
- Modern:
  Data → Encrypted → Labeled → Tracked → Protected

Tools:

- DLP
- Classification
- Encryption
- Tokenization

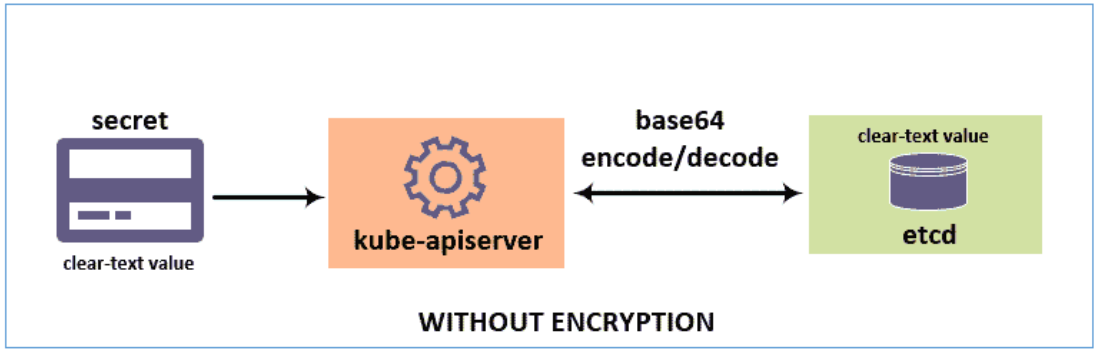


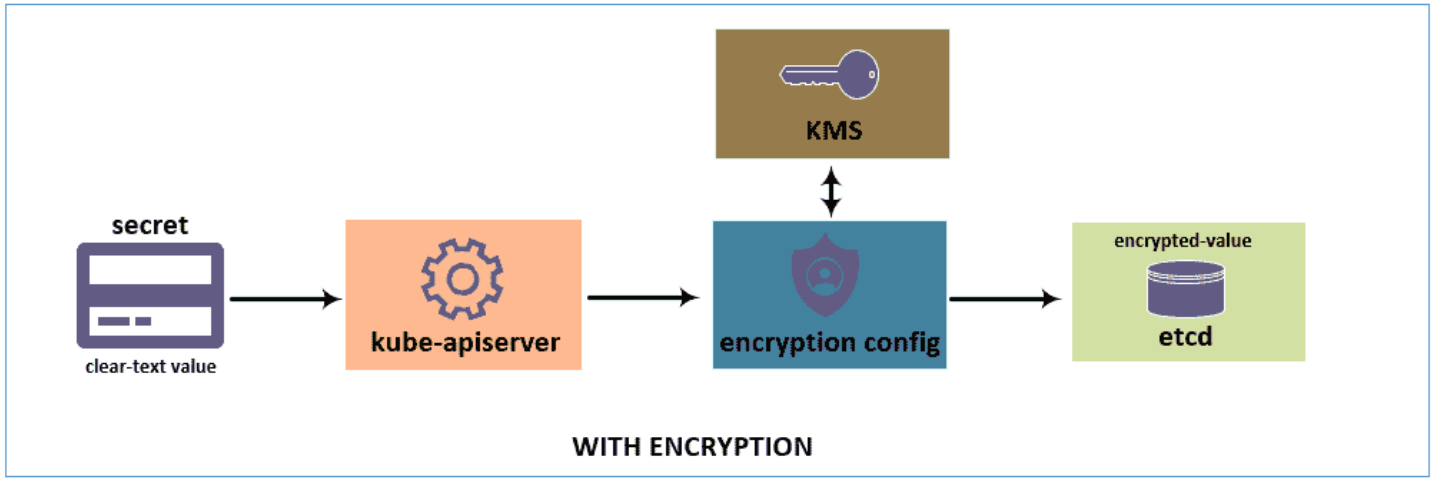
Encryption in Kubernetes

Challenges:

- Short-lived pods
- Secrets management

Encrypt secrets, not just storage





File Share & Email Encryption

- File Shares: Encrypt before saving
- Email: Use S/MIME, IRM, Purview
- Outlook: Encrypt attachments automatically
- Ensure classification labels trigger encryption

Common Pitfalls to Avoid

1. Assuming cloud encryption is enough → Use BYOK
2. Storing keys in code → Use KMS
3. Ignoring data in use → Use confidential computing
4. No classification → Can't protect what you can't see
5. No monitoring → Enable audit logs

Best Practices Summary

- [x] Enable encryption at rest & in transit
- [x] Use BYOK and central KMS
- [x] Classify data and label files
- [x] Encrypt data in use
- [x] Train teams
- [x] Audit key usage
- [x] Plan for PQC

Future Trends

- Confidential Computing
- PQC Adoption
- AI-Driven DLP
- Zero Trust Encryption
- Homomorphic Encryption
- Automated Key Rotation

The future is encrypted, private, and automated


Q&A

Secure Today, Safe Tomorrow