**Control Adoption and Compensating Controls - Data Encryption at Rest - Reference Implementation Proposal**

**Background**

The protection of sensitive data stored within organizational systems is critical to ensuring confidentiality, integrity, and compliance with regulatory standards. Challenges in implementing data encryption at rest have resulted in vulnerabilities and audit findings:

* Sensitive data stored without encryption in production and non-production environments is susceptible to unauthorized access.
* Inconsistent encryption practices across on-premises and cloud environments increase the risk of data breaches.
* Poor management of cryptographic keys compromises the security of encrypted data.

This document proposes a reference implementation for data encryption at rest to mitigate these risks, safeguard sensitive data, and support operational and development needs while aligning with firmwide policies and standards such as DP30\*\*.

**Definitions**

* **Sensitive Data** - Data classified as DP20 or above per the Firmwide Policy for Data Classification, including personally identifiable information (PII), financial records, and proprietary data.
* **Encryption at Rest** - The cryptographic protection of data stored on disks, databases, or other media to prevent unauthorized access, as mandated by the firm’s cryptography standards.
* **Environment** - A segregated set of computing resources designated for a specific purpose.
* **Production** - Current classification for environments that support the firm’s business processes, where infrastructure, systems, and applications conduct, process, or record firm business.
* **Non-Production** - Current classification for environments used for development and testing purposes.
* **Secure Environment** - Subject to the control requirements and governance mandated for handling sensitive data.
* **Data Read Sensitive Environment** - Contains or processes data that is not publicly available.
* **Data Write Sensitive Environment** - Contains or modifies data whose correctness must be guaranteed by modification through approved processes.
* **Operationally Sensitive Environment** - Critical to end-user activities, where failure would disrupt business or testing operations.
* **Business Sensitive Environment** - Produces output directly used in business processes.

**Considerations when working with data encryption at rest**

* Sensitive data must be protected from unauthorized access when stored, aligning with the firm’s cryptography standards.
* Operational integrity of production environments must be maintained during encryption processes.
* Productivity of developers, strategists, and business users requiring access to encrypted data must be preserved.

**Risks and Threat Vectors**

| **Class** | **Risk** | **Threat Vector** | **Examples** |
| --- | --- | --- | --- |
| Data | Unauthorized Access | Physical theft of unencrypted storage media | - A stolen unencrypted hard drive from a data center exposes customer PII and financial data. |
|  |  | Insider extraction of unencrypted data | - A privileged user copies unencrypted database backups to an external device. |
|  | Key Compromise | Exposure of cryptographic keys | - Keys stored in a misconfigured repository are accessed by an unauthorized user. |
|  |  | Insufficient key rotation | - Stale keys are exploited to decrypt archived sensitive data. |
|  | Data Leakage | Unencrypted data transfer | - Unencrypted backups are uploaded to an insecure cloud service. |
|  |  | Misconfigured encryption settings | - A database is deployed without enabling encryption at rest, exposing sensitive records. |
|  | Data Loss | Loss of encryption keys | - Inaccessible keys render encrypted data unrecoverable, disrupting business operations. |
| Operational | Resource Starvation | Encryption processing overhead | - CPU-intensive encryption delays transaction processing, impacting service-level objectives (SLOs). |
|  | System Downtime | Key management system failure | - A corrupted key vault prevents data decryption, halting critical applications. |
|  |  | Encryption misconfiguration | - Incorrect encryption settings cause application failures, leading to downtime. |

**SECTION TO BE REMOVED ONCE COMMENTS HAVE BEEN RESOLVED**

* Data
  + Exposure of unencrypted sensitive data
  + Compromise of cryptographic keys due to poor storage practices
  + Leakage of sensitive data through unencrypted backups or snapshots
  + Loss of encrypted data due to unavailable or corrupted keys
* Audit
  + Failure to maintain encryption audit trails
  + Non-compliance with data protection regulations and policies (e.g., DP30\*\*)
* Operational
  + Performance degradation from encryption overhead
  + System downtime due to key unavailability
  + Application failures from encryption misconfigurations

**Requirements**

* Sensitive data when stored or accessed must be encrypted at rest using approved cryptographic keys and mechanisms that comply with the firm’s cryptography standard, as per DP30\*\*.
* Encryption keys must be managed in a secure system with access controls, rotation policies, and audit logging, ensuring protection across on-premises and cloud environments.
* Data used for business processes must not be accessible in unencrypted form outside of approved processes.
* Encryption processes must not compromise the operational integrity or performance of production environments.
* Developers must be able to perform testing with encrypted data that complies with the firm’s mandated data access controls, using techniques like masking or tokenization where necessary.

**Reference Implementation Proposal**

To address the identified risks and meet the requirements, the following reference implementation for data encryption at rest is proposed:

1. *Encryption Standards*
   * Adopt AES-256 encryption in Galois/Counter Mode (GCM) for all sensitive data at rest to ensure confidentiality and integrity, compliant with FIPS 140-3 and the firm’s cryptography standard.
   * Use cryptographic modules validated by Data Asset Owners to protect sensitive data residing on-premises and in GS data centers.
2. *Key Management*
   * Deploy a centralized Key Management System (KMS) to generate, store, and manage encryption keys securely, leveraging Hardware Security Modules (HSMs) for key generation.
   * Implement automated key rotation every 12 months, with versioning to prevent data loss, and enforce role-based access controls (RBAC) as defined by Entitlement Approvers and Administrators.
   * Maintain audit trails of key operations, aligning with control specifications (e.g., AU-3.102).
3. *Environment-Specific Encryption*
   * *Production*: Enable encryption at rest for all databases (e.g., SQL Server Transparent Data Encryption), file systems (e.g., dm-crypt), and object storage (e.g., AWS S3 with SSE-KMS), validated by Technical Owners.
   * *Non-Production*: Encrypt sensitive data with separate keys to limit risk exposure, ensuring compliance with production-level controls and approved entitlement workflows (e.g., AC-3.105).
   * *Backups*: Encrypt all backup data using KMS-managed keys, with access restricted to authorized recovery processes.
4. *Performance Optimization*
   * Leverage hardware-accelerated encryption (e.g., Intel AES-NI) to minimize CPU overhead and maintain performance, monitored by Solution Owners.
   * Configure encryption at the storage layer to reduce application-level changes, ensuring operational integrity.
5. *Access and Audit Controls*
   * Implement RBAC to restrict decryption capabilities to authorized users and services, with Privilege Managers reviewing entitlements quarterly (e.g., AC-6.7.100).
   * Log all encryption and decryption operations in a tamper-proof audit trail, integrated with the firm’s central inventory (e.g., AC-24.1).
   * Conduct quarterly validations of encryption feeds to the central inventory, as required by Technical Owners (e.g., AC-24.1.100).
6. *Development and Testing*
   * Provide developers with encrypted test datasets using masked or synthetic data, compliant with data access controls and entitlement management systems (e.g., PERMIT Central).
   * Use sandboxed environments with KMS-managed keys for testing encryption workflows, ensuring no self-approval (e.g., AC-3.106).

**Implementation Roadmap**

| **Phase** | **Activity** | **Timeline** | **Owner** |
| --- | --- | --- | --- |
| 1 | Assess current encryption posture and gaps | Month 1 | Security Team |
| 2 | Deploy KMS and integrate HSMs | Months 2-3 | Infrastructure Team |
| 3 | Enable encryption for production databases and storage | Months 4-5 | Database Team |
| 4 | Extend encryption to non-production environments | Month 6 | DevOps Team |
| 5 | Implement backup encryption and key rotation policies | Month 7 | Backup Team |
| 6 | Conduct compliance audit and address findings | Month 8 | Compliance Team |

**Metrics for Success**

* Percentage of sensitive data encrypted at rest: Target 100%.
* Number of key compromise incidents: Target 0.
* Performance impact from encryption: Target <5% latency increase.
* Compliance audit findings related to encryption: Target 0.

**Risk Mitigation**

* *Key Loss*: Implement redundant KMS instances and maintain secure key backups.
* *Performance Degradation*: Optimize encryption with hardware acceleration and monitor resource usage.
* *Unauthorized Access*: Enforce least privilege access and monitor key usage for anomalies, with Privilege Manager oversight.
* *Non-Compliance*: Conduct regular audits to verify adherence to DP30\*\* and internal standards.

**Environment Encryption Matrix**

| **Environment** | **Data Read Sensitive** | **Data Write Sensitive** | **Operational Sensitivity** | **Business Sensitivity** | **Encryption Required** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| Production Database | Y | Y | Y | Y | Y | AES-256 with KMS-managed keys |
| Non-Production Database | Y | N | N | N | Y | Separate keys from production |
| File Storage | Y | Y | N | Y | Y | Disk-level encryption (e.g., dm-crypt) |
| Object Storage | Y | Y | N | Y | Y | SSE-KMS for cloud storage |
| Backups | Y | N | N | Y | Y | KMS-encrypted archives |
| Development Sandbox | N | N | N | N | N | Use tokenized or synthetic data |

**Encryption Configuration Matrix**

| **Data Read** | **Data Write** | **Operational** | **Business** | **End User** | **Environment** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| Y | Y | Y | Y | Application | Production Database | TDE with AES-256, KMS keys |
| Y | N | N | N | Developer | Non-Production Database | TDE with separate KMS keys |
| Y | Y | N | Y | Application | File Storage | dm-crypt with AES-256 |
| Y | Y | N | Y | Application | Object Storage | SSE-KMS encryption |
| Y | N | N | Y | Admin | Backups | KMS-encrypted archives |
| N | N | N | N | Developer | Sandbox | No encryption; synthetic data |

**Conclusion**

Implementing data encryption at rest is essential to protect sensitive data, ensure compliance with standards like DP30\*\*, and mitigate risks of unauthorized access or data loss. This reference implementation provides a comprehensive framework to encrypt data across environments, manage keys securely, and support development needs without compromising performance or security. By adopting these controls, the organization can strengthen its data protection posture and align with industry and regulatory requirements.