**Background**

Goldman Sachs manages significant volumes of sensitive data, including PII and financial records, across off-premises environments like public clouds and third-party providers. Vulnerabilities from inconsistent encryption and weak key management have prompted this updated reference implementation to secure data at rest, comply with firmwide standards like DP30\*\*, and support operational efficiency while mitigating risks in off-premises settings.

**Definitions**

* **Sensitive Data** - Data classified DP20 or above per the Firmwide Data Classification Policy, including PII, financial data, and proprietary information.
* **Encryption at Rest** - Cryptographic protection of data stored in off-premises digital media (e.g., cloud databases, third-party storage).
* **Off-Premises Environment** - Data residing outside GS datacenters, such as public cloud (AWS, Azure) or third-party systems.
* **Production** - Off-premises environments supporting business processes.
* **Non-Production** - Off-premises environments for development and testing.
* **Data Asset Owner** - Responsible for encryption compliance in off-premises deployments.

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

* Off-premises sensitive data must use firm-approved encryption to counter breach risks.
* Optimize performance and cost in cloud or third-party systems.
* Ensure entitlement controls align with GS policies for off-premises access.

**Risks and Threat Vectors**

| **Class** | **Risk** | **Threat Vector** | **Examples** |
| --- | --- | --- | --- |
| Data | Unauthorized Access | Misconfigured cloud storage | - Unencrypted S3 bucket exposes client data due to misconfiguration. |
|  |  | Third-party insider threat | - A provider employee accesses unencrypted off-premises backups. |
|  | Key Compromise | Exposed cloud KMS keys | - Leaked IAM roles compromise encryption keys on AWS. |
|  |  | Stale keys | - Outdated keys decrypt archived off-premises data. |
|  | Data Leakage | Unsecured data sync | - Unencrypted API transfers data to a third-party service. |
|  | Data Loss | Lost cloud keys | - Deleted keys render Azure data unrecoverable. |
| Operational | Resource Starvation | Encryption overhead | - High CPU usage slows off-premises application performance. |
|  | System Downtime | KMS failure | - Third-party KMS outage halts off-premises decryption. |

**Areas Requiring Further Review**

* Compatibility of off-premises platforms with GS cryptography standards.
* Integration of third-party KMS with GS-approved systems.
* Performance impact on off-premises high-traffic systems.
* Entitlement validation for off-premises access (e.g., AC-3.106).

**Pending Approvals**

* Business Sponsor approval for off-premises encryption workflows (e.g., AC-3.105).
* Data Protection Oversight Group sign-off on DP30\*\* compliance.
* Privilege Manager authorization for off-premises RBAC.

**Requirements**

* Off-premises data at rest (e.g., public cloud, third-party providers) must be encrypted per SC-28.1.107 using firm-approved cryptography.
* Keys must be managed with secure access, rotation, and audit logging via a GS-approved KMS.
* Data in off-premises business processes must remain encrypted outside approved workflows.
* Encryption must maintain performance in off-premises systems.
* Developers must use masked off-premises test data per SA-15.9.

**Proposal**

| **Feature** | **Immediate** | **Short Term** | **Target** |
| --- | --- | --- | --- |
| Unit Testing Data | - Manual creation not sourced from off-premises production<br>- Validation: Code reviews identifying sensitive data checked into off-premises code repository | - Add questions around operational, purpose, and data sensitivity to ARP for off-premises environments | - Tooling to provide automatic generation based on off-premises production data, ensuring compliance with SA-15.9 masking requirements |
| Block write to production from non-production process | - Review of Configuration<br>- Separation of system users by environment/role in off-premises setups<br>- Validation: App Team to document multi-environment architecture as deployment diagram to be reviewed by Technical Owners prior to onboarding | - Gateway that understands environment to handle whether traffic should pass through off-premises providers | - Environment-aware entitlements mechanism that does not allow permissioning other than as per the Environments Access Matrix in off-premises systems |
| Block read from non-production data from production process | - Separation of system users by environment/role in off-premises setups<br>- Validation: App Team to document multi-environment architecture as deployment diagram to be reviewed by Technical Owners prior to onboarding | - Gateway that understands environment to handle whether traffic should pass through off-premises providers | - Environment-aware entitlements mechanism that does not allow permissioning other than as per the Environments Access Matrix in off-premises systems |
| Noisy neighbors for Prod and Non-Prod processes | - Isolation of compute resources by environment/role in off-premises cloud instances | - N/A | - Enforced environment isolation<br>- Quota-based storage usage to prevent resource contention in off-premises environments |
| Break/glass access for a Production Engineer (write) | - Remote processes run with system account credentials<br>- User must be authenticated using a firm-approved mechanism<br>- Validation: Ensure off-premises access logs feed into GS audit systems (e.g., AU-3.102) | - Perform independent verification that the user requesting an action has a superior set of permissions to the system user<br>- System accounts (P2 users) are setup for specific activities<br>- System accounts (P2 users) are setup with minimum privileges for activity they need to perform | - Remote processes run with credentials provided to them in the form of limited time tokens<br>- Automatic generation of "DUMMY" data so that effective unit testing can be performed in off-premises sandboxes |
| Break/glass access for a Production Engineer (read) | - Remote processes run with system account credentials<br>- User must be authenticated using a firm-approved mechanism<br>- Validation: Ensure off-premises access logs feed into GS audit systems (e.g., AU-3.102) | - Perform independent verification that the user requesting an action has a superior set of permissions to the system user<br>- System accounts (P2 users) are setup for specific activities<br>- System accounts (P2 users) are setup with minimum privileges for activity they need to perform | - Remote processes run with credentials provided to them in the form of limited time tokens<br>- Automatic generation of "DUMMY" data so that effective unit testing can be performed in off-premises sandboxes |
| Segregation of roles between developer engineer and production engineer | - Separation of system users by environment/role in off-premises setups<br>- Validation: App Team to document multi-environment architecture as deployment diagram to be reviewed by Technical Owners prior to onboarding | - N/A | - Preventive measures to ensure that when credentials are provisioned there is no overloading |
| Ensure that credentials are not shared between environments | - Prod credentials should never be shared with a non-production off-premises environment<br>- Validation: Implement credential rotation checks for off-premises systems<br>- Q: Given that there is not separate classification for Prod Parallel, how would we achieve this granularity?<br>- Q: Does the separation of credentials apply in SecDB ecosystem (Does it have environments)? | - Should be able to find when credentials have been shared so that we can detect after the fact credentials have been shared<br>- (Could this be Tim D’s credential rotation (Massive Tool)?) | - Preventive measures to ensure that when credentials are provisioned there is no overloading |
| Sharing of credentials across environments i.e. Production and Non Production or Prod 1 v Prod 2 | - Why? People do this for simplicity of management at the expense of isolation<br>- Impact?<br>- Because you're not using a credential for a specific purpose then there is a chance that it may end up with accidental | - Given that there is not separate classification for Prod Parallel, how would we achieve this granularity?<br>- Could this be Tim D’s credential rotation (Massive Tool)?<br>- Q: Does the separation of credentials apply in SecDB ecosystem (Does it have environments)? | - N/A |
| Identification of a sensitive data testing environment | - Storage will be classified as prod to ensure that all audit and security controls for sensitive data are applied to off-premises testing environments<br>- Validation: App Team to document multi-environment architecture as deployment diagram to be reviewed by Technical Owners prior to onboarding | - Add questions around operational, purpose and data sensitivity to ARP for off-premises environments | - Add operational, purpose and data sensitivity to AppDir for off-premises assets |
| Malicious code to hijack a user | - Remote processes run with system account credentials<br>- Only run remote processes from controlled code bases (protected branches)<br>- Not all SDLCs have the concept of protected branch (e.g., Slang.js) | - Remote processes run with system account credentials | - Static analysis in SDLC pipeline to detect malicious code that attempts to harvest sensitive data in off-premises environments |
| Ensure that users can’t escalate their privileges by running with remote user | - Remote processes run with system account credentials<br>- User must be authenticated using a firm-approved mechanism<br>- Validation: Ensure off-premises access logs feed into GS audit systems (e.g., AU-3.102) | - Perform independent verification that the user requesting an action has a superior set of permissions to the system user<br>- System accounts (P2 users) are setup for specific activities<br>- System accounts (P2 users) are setup with minimum privileges for activity they need to perform | - Remote processes run with credentials provided to them in the form of limited time tokens<br>- Automatic generation of "DUMMY" data so that effective unit testing can be performed in off-premises sandboxes |
| Moving sensitive data to a non-controlled environment | - Not available<br>- Validation:<br>- Snowflake Prod - Non-Prod visa a share is disallowed by Middleware Eng.<br>- Design Review / Self Attestation | - Not available | - Data masking of data as it moves out of the controlled off-premises environment<br>- Enforce out of bounds copying of data by enforcing the uncontrolled environment |

**Features and Intermediate Short-Term Targets**

* **Features**:
  + Centralized off-premises KMS with HSM integration for key management.
  + Hardware-accelerated encryption (e.g., AWS Nitro) to reduce cloud performance impact.
  + Real-time audit logging of off-premises encryption events, compliant with AU-3.102.
  + Masked encrypted test datasets for off-premises non-production use, per SA-15.9.
* **Intermediate Short-Term Targets**:
  + Month 2: Deploy KMS pilot for off-premises with initial key rotation (Infrastructure Team).
  + Month 4: Achieve 50% encryption coverage in off-premises production databases (Database Team).
  + Month 6: Extend encryption to 75% of off-premises non-production environments (DevOps Team).
  + Month 7: Validate backup encryption in off-premises systems (Backup Team).

**Reference Implementation Proposal**

1. *Encryption Standards*
   * Use AES-256 in Galois/Counter Mode (GCM) for off-premises data, compliant with FIPS 140-3 and GS standards.
   * Validate cryptographic modules by Data Asset Owners for off-premises platforms.
2. *Key Management*
   * Deploy a GS-approved KMS integrated with off-premises providers (e.g., AWS KMS, Azure Key Vault), using HSMs.
   * Automate key rotation every 12 months with versioning, enforcing RBAC per AC-3.111.
   * Maintain audit trails, aligning with AU-3.102.
3. *Environment-Specific Encryption*
   * *Production*: Encrypt off-premises databases (e.g., AWS RDS with TDE), object storage (e.g., S3), and third-party systems.
   * *Non-Production*: Use separate keys for off-premises non-production, per AC-3.105.
   * *Backups*: Encrypt off-premises backups with KMS-managed keys.
4. *Performance Optimization*
   * Leverage cloud-native acceleration (e.g., AWS Nitro) to minimize overhead.
   * Configure encryption at the storage layer to reduce application changes.
5. *Access and Audit Controls*
   * Implement RBAC for off-premises decryption, with quarterly reviews by Privilege Managers (e.g., AC-6.7.100).
   * Log encryption/decryption in a tamper-proof audit trail, integrated with the central inventory (e.g., AC-24.1).
   * Conduct quarterly validations of off-premises encryption feeds (e.g., AC-24.1.100).
6. *Development and Testing*
   * Provide masked off-premises test datasets, compliant with SA-15.9.
   * Use off-premises sandboxes with KMS-managed keys, preventing self-approval (e.g., AC-3.106).

**Implementation Roadmap**

| **Phase** | **Activity** | **Timeline** | **Owner** |
| --- | --- | --- | --- |
| 1 | Assess off-premises encryption posture | Month 1 | Security Team |
| 2 | Deploy KMS and integrate with off-premises providers | Months 2-3 | Infrastructure Team |
| 3 | Enable encryption for off-premises production | Months 4-5 | Database Team |
| 4 | Extend encryption to off-premises non-production | Month 6 | DevOps Team |
| 5 | Implement backup encryption and key rotation | Month 7 | Backup Team |
| 6 | Conduct compliance audit | Month 8 | Compliance Team |

**Metrics for Success**

* Percentage of off-premises sensitive data encrypted: Target 100%.
* Key compromise incidents: Target 0.
* Performance impact: Target <5% latency increase.
* Compliance findings: Target 0.

**Risk Mitigation**

* *Key Loss*: Maintain redundant off-premises KMS instances and backups.
* *Performance Degradation*: Optimize with cloud acceleration and monitoring.
* *Unauthorized Access*: Enforce least privilege and monitor key usage.
* *Non-Compliance*: Conduct regular DP30\*\* audits.

**Environment Encryption Matrix**

| **Environment** | **Data Read Sensitive** | **Data Write Sensitive** | **Operational Sensitivity** | **Business Sensitivity** | **Encryption Required** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| Off-Premises Production Database | Y | Y | Y | Y | Y | AES-256 with KMS keys |
| Off-Premises Non-Production Database | Y | N | N | N | Y | Separate keys |
| Off-Premises Object Storage | Y | Y | N | Y | Y | Cloud encryption (e.g., S3) |
| Off-Premises Backups | Y | N | N | Y | Y | KMS-encrypted |
| Off-Premises Sandbox | N | N | N | N | N | Masked/synthetic data |

**Encryption Configuration Matrix**

| **Data Read** | **Data Write** | **Operational** | **Business** | **End User** | **Environment** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- |
| Y | Y | Y | Y | Application | Off-Premises Production Database | TDE with AES-256 |
| Y | N | N | N | Developer | Off-Premises Non-Production Database | TDE with separate keys |
| Y | Y | N | Y | Application | Off-Premises Object Storage | Cloud encryption |
| Y | N | N | Y | Admin | Off-Premises Backups | KMS-encrypted |
| N | N | N | N | Developer | Off-Premises Sandbox | Masked data |

**Conclusion**

This updated implementation secures off-premises data at rest, ensuring compliance with DP30\*\* and mitigating risks like unauthorized access. It supports operational needs with optimized encryption and robust key management, strengthening GS’s off-premises security posture.