Sprint 4: Securing Sensitive Data in a Financial Application with AWS KMS and Cloud HSM

Team Name: DataDefender

Team Members: Takala, Awa, Jaelin, and Mariana

Takala
Awa
Jaelin
Mariana

Sprint 1: Foundation and Planning (Securing Sensitive Data with AWS KMS and CloudHSM) - Jaelin, Awa, Mariana
Sprint 2: Secure Key Management with AWS KMS - Jaelin,Awa
Sprint 3: Enhanced Security with AWS CloudHSM - Jaelin
Sprint 4: Securing Data in Transit with KMS - Awa,Jaelin

Submit the following requirements for Sprint 4: Securing Data in Transit with KMS

Technical Documentation:

KMS Integration with Key Management Middleware

Key Management Middleware Selection

Describe the key management middleware solution you have chosen (e.g., HashiCorp Vault) and justify your selection.

We selected HashiCorp Vault for its centralized secret management, advanced access controls, and seamless AWS KMS integration. Vault simplifies key management and enhances security by providing fine-grained policies and robust auditing capabilities.

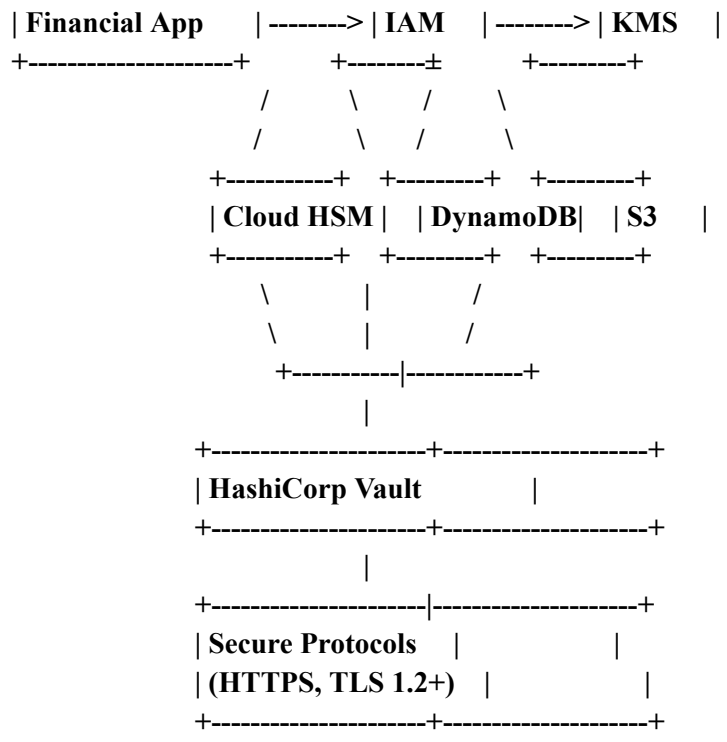
Explain the benefits of using a key management middleware over directly integrating with AWS KMS.

Benefits of Middleware over Direct Integration with AWS KMS:

- Centralized management of secrets and encryption keys
- Enhanced access controls and policy management
- Comprehensive auditing and logging
- Flexibility to integrate with multiple backends

Integration Architecture

±----------------± ±-------+ ±-------±



Provide an architectural diagram illustrating the integration between AWS KMS, the key management middleware, and your applications/services.

Overview: Our architecture ensures secure key management and data encryption for our financial application.

- ➢ **Financial Application:** Interacts with IAM for access control.
- ➢ **IAM:** manages access and controls KMS for key operations.
- ➢ **KMS:** Encrypts financial data using managed keys.
- ➢ **Cloud HSM:** Provides hardware-based key security.
- ➢ **DynamoDB:** Securely stores encrypted financial data.
- ➢ **HashiCorp Vault:** Optionally used for enhanced key management.
- ➢ **Secure Protocols:** HTTPS and TLS 1.2+ for secure data transmission.

Describe the data flow and key management processes involved in this integration.

Data Flow and Key Management Processes:

Integration Architecture:

- **Financial App:** Initiates data transactions and requests access to encryption keys.
- **IAM (Identity and Access Management):** Controls access to AWS services, granting permissions to users based on their roles.
- **KMS (Key Management Service):** Responsible for encryption key management, encrypting and decrypting data.
- **Cloud HSM (Hardware Security Module):** Enhances security by securely storing and managing encryption keys used by KMS.
- **DynamoDB:** A NoSQL database service for storing encrypted data.
- **S3 (Simple Storage Service):** Securely stores objects, providing scalable storage capacity.
- **HashiCorp Vault:** Centrally manages encryption keys and integrates with KMS for enhanced security.
- **Secure Protocols (HTTPS, TLS 1.2+):** Ensures secure transmission of data between components.

How Roles Play Out:

1. **Administrator:**
    - Role: Oversees overall AWS environment management and security for the financial app.
    - Permissions: Full access to KMS, CloudHSM, and IAM for key creation, management, and user permissions.
2. **Data Engineer:**
    - Role: Focuses on implementing and managing data encryption mechanisms within the financial app.
    - Permissions: Full access to KMS, CloudHSM, Amazon S3, and Amazon RDS for encryption, storage, and management.
3. **Security Analyst:**
    - Role: Monitors and analyzes security events and compliance status within the AWS environment.
    - Permissions: Read-only access to KMS for monitoring key usage, CloudHSM for security analysis, and CloudTrail for tracking API activity.
4. **Developer:**
    - Role: Develops and integrates security features, including encryption, within the financial app.

    - Permissions: Read-only access to KMS for testing encryption mechanisms, and read-only access to Amazon S3 and Amazon RDS for testing data storage and retrieval.
5. **Compliance Officer:**
    - Role: Ensures regulatory compliance of the financial app.
    - Permissions: Read-only access to KMS for reviewing compliance-related key usage reports, and read-only access to CloudTrail for compliance auditing.

Key Management Workflows

Outline the key management workflows implemented in your solution, such as key generation, key rotation, and key revocation.Explain how the key management middleware interacts with AWS KMS to perform these workflows.

- **Key Generation:**
    - New encryption keys are created securely when needed.
    - Example: When a new customer signs up, a unique encryption key is made.
- **Key Rotation:**
    - Keys are regularly changed for added security.
    - Example: Every six months, keys are automatically updated to newer ones.
- **Key Revocation:**
    - Keys can be revoked if needed, like when a customer cancels their account.
    - Example: If a customer leaves, their key is made unusable to protect their data.

Interaction with AWS KMS:

- Our system talks to AWS KMS to handle these key tasks.
- For key generation, our system asks AWS KMS to create a new key.
- For rotation, it tells AWS KMS to swap out old keys for new ones.
- When revoking keys, our system tells AWS KMS to make them invalid.

Discuss any challenges faced during the integration process and how you addressed them.

Challenges and Solutions:

Integration Complexity:

- **Challenge:** Connecting different parts of our cloud setup is still tricky as each has its own way of working.
- **Solution:** We keep talking and planning with all teams involved, making sure everyone knows what to do and how to do it.

Access Control Configuration:

- **Challenge:** Figuring out who should have access to what, especially with encryption keys, is still a puzzle.
- **Solution:** We're carefully setting up rules so people only have access to what they need, and we're checking regularly to make sure it's right.

Logging and Auditing Setup:

- **Challenge:** Keeping track of who's doing what with our keys and data is complex.
- **Solution:** We're using CloudTrail and Vault logs to watch everything. It's taking some time to get it all set up, but we're making progress.

Ensuring Compliance:

- **Challenge:** Meeting all the rules and regulations, like GDPR and HIPAA, is tough because there are lots of details.
- **Solution:** We're studying the rules and making sure our security measures match up. Regular checks help us stay on track.

Access Control and Auditing

Describe the access control mechanisms implemented to secure access to encryption keys managed by the middleware. Explain how you have integrated auditing and logging capabilities to track key usage and changes.

Access Control Mechanisms:

- Access to customer data is controlled based on user roles.
- For example, only users assigned the “Administrator” role have full access to manage AWS services such as IAM and KMS.

- Similarly, the “Developer” role is granted permissions to test encryption mechanisms within the financial application, while the “Compliance Officer” role has read-only access to review compliance-related reports.

Auditing and logging capabilities:

- We track access and changes to data using AWS Cloud Trail and Vault logs.
- Cloud Trail records activity related to AWS services.
- Vault logs track key usage and modifications within the Vault environment.
- These logs help us maintain compliance with regulations like GDPR, PCI-DSS, and HIPAA, ensuring data security.

HTTPS and TLS Implementation

Describe the approach you have taken to implement HTTPS and TLS for secure data transmission across your cloud environment. Explain the process of obtaining and managing SSL/TLS certificates for your applications and services. Discuss any specific configurations or best practices you have followed for HTTPS and TLS implementation.

Secure communication channels:

- ➢ We make sure data travels safely between different parts of our cloud setup at **SecureUtility Solutions**.
- ➢ We use HTTPS and TLS to keep data encrypted and secure during transmission.
- ➢ To enable this encryption, we will get SSL/TLS certificates from trusted sources.
- ➢ These certificates will help us verify the identity of our servers and ensure secure connections with client applications.
- ➢ We regularly update and monitor these certificates to ensure they're valid and secure.
- ➢ Specific configurations include using the latest TLS versions, strong encryption methods, and enforcing HTTPS usage.
- ➢ By doing this, we ensure that data moving through our cloud environment stays safe from prying eyes and potential attacks.

Data Encryption in Transit

Outline the data encryption mechanisms used to secure data in transit between cloud resources (e.g., applications, databases, storage services). Describe how you have integrated AWS KMS or other encryption key management solutions to manage the encryption keys used for data in transit.

Data Encryption in Transit:

- ➢ We keep data safe as it moves between different parts of our cloud setup.
- ➢ We use special protocols like HTTPS and TLS 1.2+ to encrypt the data while it travels.
- ➢ These protocols make sure that only authorized people can read the data.
- ➢ We work with AWS KMS to manage the keys that are used to encrypt and decrypt the data.
- ➢ AWS KMS helps us create and manage these keys securely.
- ➢ By doing this, we make sure that sensitive data stays safe and protected while it's on the move in our cloud system.

Secure Communication Channels

Explain the secure communication channels established between different components of your cloud environment (e.g., VPN, AWS Direct Connect, Transit Gateway). Discuss any specific security measures or configurations implemented to secure these communication channels.

Secure Communication Channels:

- ➢ We've set up safe ways for different parts of our cloud system to talk to each other.
- ➢ Our financial app talks securely with IAM, KMS, Cloud HSM, DynamoDB, and S3 using HTTPS and TLS 1.2+.
- ➢ HTTPS keeps data encrypted, so it can't be seen by anyone who shouldn't.
- ➢ TLS 1.2+ makes sure that data is protected as it moves between components, stopping anyone from listening in or tampering with it.
- ➢ We've made sure that all communication between our components stays inside a private network, so it's safe from outsiders.
- ➢ Only authorized users and components can access sensitive resources like KMS, Cloud HSM, DynamoDB, and S3, using strict access controls.
- ➢ With these measures in place, we make sure that our data stays safe as it travels within our cloud environment.

Monitoring and Compliance

Describe the monitoring and logging mechanisms implemented to ensure the integrity and security of data in transit. Explain how you have addressed compliance requirements or industry standards related to secure data transmission (e.g., PCI-DSS, HIPAA, GDPR).

Monitoring and logging mechanisms:

- ➔ We keep a close watch on how sensitive financial data moves through our system.
- ➔ AWS CloudWatch and Vault help us track what's happening in real-time.
- ➔ CloudWatch looks at network traffic and system performance, while Vault checks data integrity and security.

- ➔ With these tools, we can quickly spot any problems and keep our financial data safe.

Compliance with Standards:

- ➔ Our system follows strict rules like PCI-DSS to protect financial data.
- ➔ We use strong encryption and access controls at different points in our setup.
- ➔ AWS KMS manages encryption keys securely, and Cloud HSM adds an extra layer of security.
- ➔ We regularly check to make sure we're following the rules.
- ➔ By doing this, we show that we're serious about keeping financial data safe and meeting industry standards.