IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15344

Secure docker container with AWS secrets manager

M.J.Premkarthik[1], Dr. S. Shylaja[2]

III BCA, Department of Computer Applications,

Sri Ramakrishna College of Arts & Science (Autonomous), Coimbatore, Tamil Nadu, India[1]

Assistant Professor, Department of Computer Applications,

Sri Ramakrishna College of Arts & Science (Autonomous), Coimbatore, Tamil Nadu, India[2]

Abstract: Modern software systems require flexible and scalable deployment environments to support growing workloads and complex applications. Containerization has emerged as an efficient solution that allows applications to be packaged with all required dependencies into lightweight containers. Docker is one of the most widely used containerization technologies that enables developers to build portable and consistent application environments. However, managing sensitive data such as API keys, database passwords, and configuration tokens within these containers requires a highly secure and reliable approach.
This research presents a cloud-based security architecture titled “Secure Docker Container with AWS Secrets Manager.” The system integrates Docker containerization with Amazon Web Services (AWS) security components, primarily focusing on AWS Secrets Manager, along with Amazon Elastic Compute Cloud (EC2) and AWS Identity and Access Management (IAM).

The proposed system demonstrates a highly secure container deployment model that improves application safety, credential management, and compliance. The results show that combining Docker with AWS Secrets Manager can significantly reduce security vulnerabilities and support modern DevSecOps practices for building resilient, secure applications.

Keywords: Docker, AWS Secrets Manager, Container Security, Cloud Deployment, Credential Management, DevSecOps.

I. INTRODUCTION

Cloud computing has become a fundamental technology for modern software development and deployment. Organizations increasingly rely on cloud platforms to host applications because they offer scalability, reliability, and flexible resource management. However, traditional application deployment methods often depend on hardcoded credentials and manual security configurations. These approaches can lead to critical data breaches, unauthorized access, and complex compliance auditing processes.

Containerization has emerged as an effective approach to package applications. Containers allow applications to run in isolated environments while sharing the underlying operating system. This ensures that applications behave consistently across development, testing, and production environments. Docker is one of the most widely adopted container platforms used to build, package, and distribute containerized applications.

Although Docker simplifies application packaging, managing the sensitive credentials required by these applications presents a major security challenge. Passing secrets via plain-text environment variables or embedding them directly into Docker images exposes them to significant risks. Amazon Web Services provides AWS Secrets Manager, a fully managed credentials management service designed to securely store, distribute, and rotate secrets in the cloud.

This research focuses on implementing a secure container deployment system using Docker and AWS Secrets Manager.The architecture integrates several AWS services including IAM for access control, Secrets Manager for credential storage, and EC2 for infrastructure management. The goal of the project is to demonstrate a secure container lifecycle in a cloud environment while ensuring robust credential protection, operational efficiency, and strict access controls.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15344

II. RELATED WORK

Several studies have explored the use of containerization and cloud security technologies to improve application deployment processes. Traditional deployment methods rely on storing configuration files and credentials directly on physical servers or virtual machines. These systems often require manual rotation of passwords and leave sensitive data exposed in plain text, which can lead to severe security vulnerabilities.

Container technologies such as Docker have been introduced to provide lightweight and portable environments for application execution. While Docker containers package application code and dependencies effectively, early adoption often relied on insecure practices like committing .env files to source control or passing secrets via the Docker command line.

In recent years, centralized secret management platforms such as HashiCorp Vault, AWS Systems Manager Parameter Store, and AWS Secrets Manager have been developed to manage credentials in distributed deployments. These platforms automate tasks such as secure secret storage, dynamic retrieval, and automated password rotation.

Among these tools, AWS Secrets Manager offers strong integration with other AWS services, making it highly suitable for organizations already using AWS cloud infrastructure. Secrets Manager simplifies credential handling by providing encrypted storage and automated lifecycle management. By combining containerization with managed secrets, organizations can build highly secure and compliant cloud-native applications.

III. OBJECTIVES AND CHALLENGES

Objectives

The primary objectives of this project are:

1. To implement application containerization using Docker.
2. To design a secure credential management architecture in a cloud environment.
3. To store and manage sensitive data securely using AWS Secrets Manager.
4. To deploy containerized applications that retrieve secrets dynamically at runtime without exposing them.
5. To implement secure access control using AWS Identity and Access Management (IAM).
6. To demonstrate automated and secure container deployment using AWS infrastructure.

Development Challenges

During the implementation of the system, several challenges were encountered while integrating Docker with AWS security services.

One of the major challenges involved configuring IAM policies to ensure secure access control, allowing only authorized containers to fetch specific secrets while blocking all other access. Another challenge involved authenticating the Docker containerized application with AWS SDKs and retrieving the secrets securely at runtime.

Correct configuration of AWS credentials and ensuring the application could parse the retrieved JSON secret payloads was necessary for successful database or API connections.

Managing the securely passed variables and preventing them from leaking into application logs also required careful planning. Additionally, troubleshooting IAM permission errors and resolving runtime secret-fetching issues were important tasks during system implementation.

IV. SYSTEM ARCHITECTURE

The proposed system architecture follows a cloud-based secure deployment model that integrates Docker containerization with AWS security services.The architecture consists of multiple layers that work together to build, secure, and deploy

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15344

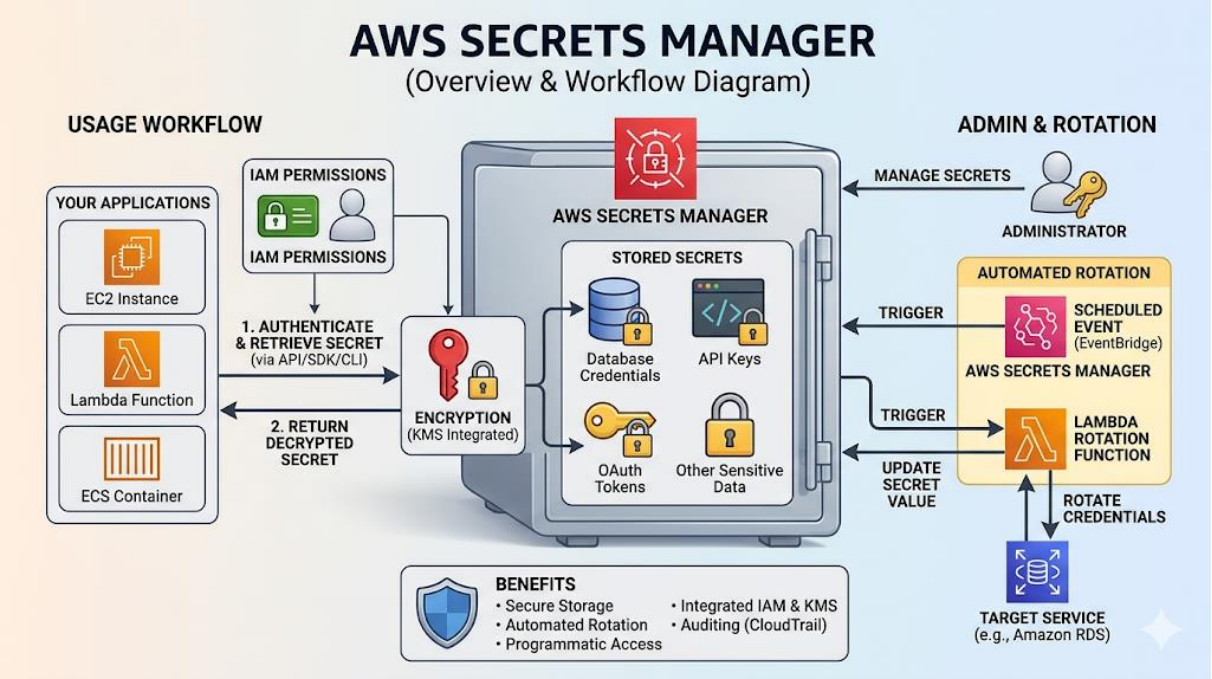
containerized applications.

The Application Layer represents the web application that is prepared for containerization. The application code is designed to utilize AWS SDKs or API calls to fetch required credentials upon startup rather than relying on local configuration files.

The Containerization Layer uses Docker to build container images. A Dockerfile is used to define the base environment and application files, ensuring no sensitive data is baked into the image layers. The Secrets Management Layer is handled by AWS Secrets Manager.

This service stores database passwords and API keys securely using KMS (Key Management Service) encryption and serves them to the application dynamically when requested. The Infrastructure Layer is provided by Amazon EC2 instances.

These instances host the Docker containers and provide the computing resources needed for execution. Security is implemented through AWS Identity and Access Management (IAM), which controls access permissions via instance profiles or task roles, ensuring that only the specific Docker container.



V. IMPLEMENTATION

The system implementation involved several steps starting from container creation to secure cloud deployment. Initially, the application was containerized using Docker.

A Dockerfile was created to define the container environment and copy application files into the container image. The application code was modified to include the AWS SDK to fetch credentials from Secrets Manager during initialization.

The Docker image was then built using Docker commands. After building the container image, the necessary credentials (such as database passwords) were manually created and stored as secure strings inside AWS Secrets Manager.

Next, an IAM Role was created with strict, least-privilege policies that only granted the secretsmanager:GetSecretValue permission for the specific secret ARN. This role was attached to the EC2 instance hosting the container.

Finally, the Docker container was launched on the EC2 instance. Upon startup, the application successfully authenticated using the assigned IAM role, retrieved the secret payload from AWS Secrets Manager, and established its required database connections. The implementation demonstrated how containerized applications can securely manage

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15344

sensitive data efficiently using AWS cloud infrastructure.

VI. EVALUATION RESULTS AND DISCUSSIONS

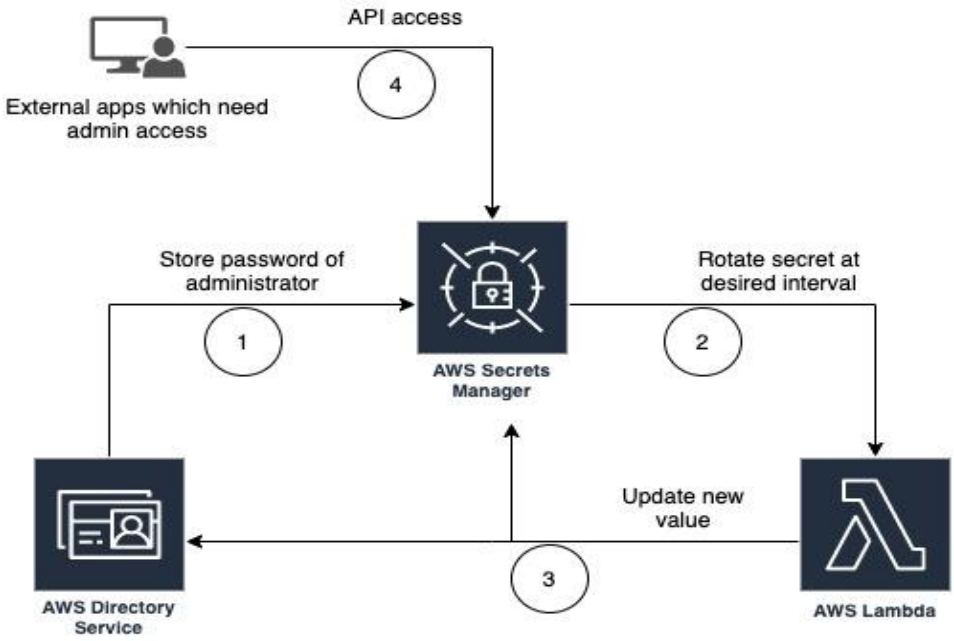
The system was evaluated based on security posture, credential retrieval performance, and application accessibility.

Several tests were conducted to measure the efficiency of the secure deployment process. Security evaluation included verifying that no secrets were exposed in the Docker history, image layers, or environment variables (docker inspect).

The results showed that integrating AWS Secrets Manager completely eliminated the risk of exposed hardcoded credentials, significantly improving system reliability and compliance. Containers started successfully, and the latency added by the API call to fetch secrets from AWS Secrets Manager during startup was measured and found to be negligible.

The application was accessible and fully functional, indicating a successful and secure deployment. The system also demonstrated highly efficient access control, as testing with unauthorized IAM roles successfully blocked the container from retrieving the secrets, proving the effectiveness of the security boundary.

Overall, the evaluation results confirm that securing Docker containers using AWS Secrets Manager provides a highly reliable, safe, and robust solution for protecting modern cloud applications.



VII. CONCLUSION

This research successfully implemented a secure Dockerized architecture using Amazon Web Services. The integration of Docker with AWS services such as Secrets Manager, EC2, and IAM provides a highly secure platform for deploying containerized applications without exposing sensitive data.

Docker simplifies application packaging and ensures consistency across environments, while AWS Secrets Manager automates secure credential storage and dynamic retrieval. The use of IAM enhances system security through controlled access permissions and least-privilege role-based authentication.

The proposed architecture replaces traditional, insecure credential management models with a cloud-native, DevSecOps-aligned system.

This approach eliminates hardcoded secrets, reduces security vulnerabilities, and simplifies compliance auditing. Overall, the project demonstrates how containerization and cloud security technologies can be seamlessly combined to create highly secure and reliable application deployment frameworks.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15344

VIII. FUTURE ENHANCEMENTS

Several improvements can be implemented to enhance the system in the future. One possible enhancement is the integration of automatic secret rotation within AWS Secrets Manager, allowing credentials (like database passwords) to be changed automatically on a schedule without requiring application downtime or manual intervention.

Another improvement is extending this security architecture to a fully managed orchestration platform like Amazon Elastic Kubernetes Service (EKS) or ECS, utilizing native secrets integration features to inject secrets directly into containers.

The system can also be enhanced by implementing continuous auditing using AWS CloudTrail to monitor and alert on exactly when and how often secrets are accessed by the containers. Additional enhancements may include implementing CI/CD pipelines using AWS CodePipeline to automate vulnerability scanning of the Docker images before they are deployed to the EC2 instances.

REFERENCES

[1]. Amazon Web Services Documentation – https://docs.aws.amazon.com
[2]. Docker Documentation – https://docs.docker.com
[3]. AWS Secrets Manager User Guide – https://docs.aws.amazon.com/secretsmanager/
[4]. Wittig, A., & Wittig, M. Amazon Web Services in Action.
[5]. Turnbull, J. The Docker Book: Containerization is the New Virtualization.
[6]. IEEE Cloud Computing Journal – Cloud Security and Containerization Research Papers.
[7]. Cloud Security Alliance – Best Practices for Secure Cloud Deployment.