**AWS Cloud Security Automation Project**

***Phase 1: Planning & Documentation***

**Prepared by:** Vinay Kumar  
**Date:** October 2025  
**Version:** 1.0

**Table of Contents**

1. Project Objective
2. Problem Statement
3. Scope of Work
4. Tools and Services
5. Architecture Plan
6. Data Flow Design
7. Project Roadmap & Deliverables

**1. Project Objective**

The objective of this project is to design and implement an **automated AWS cloud security monitoring system** that continuously assesses and improves the security posture of cloud resources.

The system integrates **Cloud Security Posture Management (CSPM)**, **Cloud Workload Protection Platform (CWPP)**, and **Cloud Access Security Broker (CASB)** simulation to:

* Detect and report misconfigurations, vulnerabilities, and non-compliant resources.
* Automate alerting and basic remediation workflows using AWS-native tools.
* Provide a centralized dashboard for visibility across the AWS environment.
* Reduce manual effort and improve compliance with organizational and regulatory standards.

**2. Problem Statement**

Managing security in a growing AWS environment is complex and time-consuming.  
Organizations often face challenges such as:

* Misconfigured resources leading to data exposure.
* Lack of continuous monitoring for workloads and accounts.
* Limited visibility across multiple AWS services and user access.
* Manual review of alerts and security findings causing delays in response.

This project aims to solve these issues by **automating cloud security posture management and visibility** through a centralized, AWS-native solution integrated with Python automation.

**3. Scope of Work**

**In Scope:**

* Implementation of AWS-native tools such as AWS Config, Security Hub, Guard Duty, and Inspector.
* Python automation for data collection, alerting, and reporting.
* Security posture assessment (CSPM), workload protection (CWPP), and SaaS visibility simulation (CASB).
* Dashboard/report generation for findings visualization.

**Out of Scope:**

* Third-party SaaS security integrations (non-AWS).
* Real-world production data or external threat feeds.
* Large-scale multi-cloud environments (Azure/GCP).

**4. Tools and Services**

| **Category** | **AWS Service / Tool** | **Purpose** |
| --- | --- | --- |
| Configuration Management | AWS Config | Detect configuration changes and compliance status |
| Threat Detection | Amazon GuardDuty | Identify potential threats and malicious activity |
| Posture Management | AWS Security Hub | Aggregate and prioritize findings |
| Vulnerability Scanning | Amazon Inspector | Detect vulnerabilities in workloads |
| Automation | AWS Lambda | Automate data collection and remediation |
| Storage | Amazon S3 | Store findings and reports |
| Access Control | IAM | Manage secure roles and permissions |
| Visualization | Python + Streamlit (optional) | Dashboard for findings and summary reports |

**5. Architecture Plan**

The architecture combines AWS-native services for continuous monitoring and automation.

**Workflow Overview:**

1. **AWS Config** monitors configurations and compliance rules.
2. **Security Hub** aggregates findings from Config, GuardDuty, and Inspector.
3. **Python scripts (via Lambda)** fetch findings using AWS SDK (boto3).
4. **S3** stores raw findings and processed reports.
5. **Optional dashboard (Streamlit or HTML)** displays findings and metrics.

AWS Config (Compliance)

Security Hub (Aggregates findings)

Python Scripts (boto3 / Lambda automation)

Streamlit / Dashboard

Amazon S3 (Stores data & reports)

**6. Data Flow Design**

**Flow Description:**

1. Cloud resources generate configuration and event data.
2. AWS Config and Security Hub collect and evaluate the data.
3. Findings are fetched through Python scripts using the AWS SDK.
4. Results are stored in S3 and optionally visualized in a dashboard.

Python Scripts (AWS SDK)

Cloud Resources

Dashboard (optional)

Python Scripts (AWS SDK)

AWS Config & Security Hub

S3 Bucket

**7. Project Roadmap and Deliverables**

The following roadmap outlines the phases, descriptions, and expected deliverables of the project:

| **Phase** | **Description** | **Deliverables** |
| --- | --- | --- |
| **Phase 1: Planning & Documentation** | Define project, problem statement, scope, architecture, and roadmap. | - Complete planning document including scope, objectives, and timelines - Architecture & Data Flow Diagrams |
| **Phase 2: Technical Implementation** | Set up AWS services, write Python scripts for data collection, and build dashboard. | - Python scripts for automation and data collection - AWS architecture setup - Sample outputs |
| **Phase 3: Reporting & Showcase** | Prepare final reports, screenshots, and upload to GitHub. | - Final report - Presentation (PPT) - GitHub repository |