

## Enterprise Automated Security Remediation and Server Hardening Framework on AWS

### 1. Problem Statement.

An online commerce platform experienced frequent configuration drift and delayed remediation of security findings. Manual intervention resulted in slow response times and increased exposure to misconfigurations. The organization required automated remediation workflows to reduce risk and improve security posture.

I implemented this framework to reduce manual security operations, prevent misconfigurations, and improve incident response time through automation and centralized monitoring.

The solution integrates detection, remediation, patching, and compliance enforcement into a unified security operations model.

### 2. Objectives

The primary objectives were to:

- Reduce manual security remediation efforts
- Automate response to common security incidents
- Establish hardened server baselines
- Improve patch management consistency
- Enforce baseline compliance
- Enhance monitoring and alerting
- Improve overall security maturity

### 3. Business and Security Context

The environment supported a rapidly scaling e-commerce platform relying heavily on EC2 and managed AWS services.

Rapid growth led to:

- Overly permissive security groups

- Outdated operating systems
- Delayed remediation
- Inconsistent server hardening

These risks increased exposure to compromise and operational disruption.

Automation was introduced to address these challenges.

#### 4. Solution Architecture

The security automation platform was built using:

- Amazon GuardDuty
- Amazon EventBridge
- AWS Lambda
- Amazon SNS
- AWS Systems Manager
- AWS Config
- Hardened Amazon Machine Images (AMIs)

Security findings from GuardDuty and CloudTrail were routed through EventBridge to trigger automated remediation workflows.

#### 5. Automated Remediation Implementation

##### 5.1 Security Group Remediation

A Lambda function monitored CloudTrail events related to security group changes.

The screenshot shows the AWS CloudWatch Log Management interface. The left sidebar includes sections for CloudWatch, Favorites and recent, Alarms, AI Operations, GenAI Observability, Application Signals (APM), Infrastructure Monitoring, Log Management, Metrics, and Metrics Explorer. The main pane displays 'Log events' for the path /aws/lambda/AutoFix-SecurityGroup. A search bar at the top allows filtering by timestamp or message. Below the search bar are buttons for Actions, Start tailing, Create metric filter, Clear, and time range selection (1m, 30m, 1h, 12h, Custom, UTC timezone). A 'Display' button is also present. The log entries show the following sequence:

- 2026-02-02T19:39:22.594Z INIT\_START Runtime Version: python:3.12-v102 Runtime Version ARN: arn:aws:lambda:us-east-1:runtime:e7b9763d75865080d2f1c073b67e435566c240009faa1c13d7389d7ba#f25
- 2026-02-02T19:39:23.113Z START RequestId: 2571315f-2958-4d21-a30e-2f75b2b4c44 Version: SLATEST
- 2026-02-02T19:39:23.116Z EVENT RECEIVED: {"version": "0", "id": "f7946290-bb02-d840-3121-c3ee866cd8d", "detail-type": "AWS API Call via CloudTrail", "source": "aws.ec2", "account": "879381257906", "time": "2026-02-02T19:39:23.116Z", "region": "us-east-1", "resources": [{"ARN": "arn:aws:sns:us-east-1:879381257906:Security-Automation-Alerts"}]}
- 2026-02-02T19:39:23.116Z SNS TOPIC ARN: arn:aws:sns:us-east-1:879381257906:Security-Automation-Alerts
- 2026-02-02T19:39:23.116Z SNS TOPIC ARN: arn:aws:sns:us-east-1:879381257906:Security-Automation-Alerts
- 2026-02-02T19:39:23.116Z Found open rule, removing...
- 2026-02-02T19:39:23.721Z Remediation result: True
- 2026-02-02T19:39:23.721Z Remediation result: True
- 2026-02-02T19:39:23.721Z Attempting to publish to SNS...
- 2026-02-02T19:39:23.721Z Attempting to publish to SNS...
- 2026-02-02T19:39:23.721Z SNS publish response: {"MessageId": "f88ade94-0e22-58f9-9819-bf0ef189b2ec", "ResponseMetadata": {"RequestId": "02cbff91-3dc8-5e27-e908-9eb0c2033432", "HTTPStatusCode": 200, "HTTPHeaders": {"x-amzn-requestid": "02cbff91-3dc8-5e27-e908-9eb0c2033432", "date": "Mon, 02 Feb 2026 19:39:23 GMT", "content-type": "text/xml", "content-length": "294", "connection": "keep-alive", "RetryAttempts": "0"}}}
- 2026-02-02T19:39:23.721Z SNS publish response: {"MessageId": "f88ade94-0e22-58f9-9819-bf0ef189b2ec", "ResponseMetadata": {"RequestId": "02cbff91-3dc8-5e27-e908-9eb0c2033432", "HTTPStatusCode": 200, "HTTPHeaders": {"x-amzn-requestid": "02cbff91-3dc8-5e27-e908-9eb0c2033432", "date": "Mon, 02 Feb 2026 19:39:23 GMT", "content-type": "text/xml", "content-length": "294", "connection": "keep-alive", "RetryAttempts": "0"}}}
- 2026-02-02T19:39:23.927Z END RequestId: 2571315f-2958-4d21-a30e-2f75b2b4c44
- 2026-02-02T19:39:23.927Z REPORT RequestId: 2571315f-2958-4d21-a30e-2f75b2b4c44 Duration: 811.12 ms Billed Duration: 1330 ms Memory Size: 128 MB Max Memory Used: 98 MB Init Duration: 517.92 ms
- No newer events at this moment. Auto retry paused. Resume

When unrestricted access (0.0.0.0/0) was detected, the rule was automatically removed and administrators were notified.

The screenshot shows a Gmail inbox with 383 unread messages. The email in focus is from 'AWS Notifications' to 'me' with the subject 'Auto-remediation: Open Security Group Fixed'. The email body contains the following text:

Removed 0.0.0.0 rule from Security Group: sg-0d7924a3c79f0fb9

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:  
<https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:879381257906:Security-Automation-Alerts:96fa7c8b-6334-4c3f-a58b-a4bf794564df&EndPoint=agentladav0@gmail.com>

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at <https://aws.amazon.com/support>

## 5.2 EC2 Quarantine Automation

A dedicated Lambda function processed GuardDuty findings related to compromised instances.

The screenshot shows the AWS Lambda function details page for 'AutoQuarantine-EC2'. A green success message at the top states: 'Successfully updated the function AutoQuarantine-EC2.' Below it, the 'Executing function: succeeded' section shows the log output:

```
{
  "status": "quarantined",
  "instance_id": "i-0x10253d0683a84",
  "eni_id": "eni-01119ec5c8df96d1"
}
```

The 'Summary' section includes the following details:

- Code SHA-256:** BICXip0tDWaCaLMAYMqCHkzUfnJMyudtl2iVbts=
- Function version:** \$LATEST
- Duration:** 967.63 ms
- Resources configured:** 128 MB
- Init duration:** 681.19 ms

**Log output:**

The area below shows the last 4 KB of the execution log. Click here to view the corresponding CloudWatch log group.

```
START RequestId: 994a5c21-140e-425b-841d-7ab2efbf863 Version: $LATEST
ENV: eni-01119ec5c8df96d1
Instance From Finding: i-0x10253d0683a84
ENI: eni-01119ec5c8df96d1
Quarantine applied.
END RequestId: 994a5c21-140e-425b-841d-7ab2efbf863
REPORT RequestId: 994a5c21-140e-425b-841d-7ab2efbf863 Duration: 967.63 ms Billed Duration: 1649 ms Memory Size: 128 MB Max Memory Used: 98 MB Init Duration: 681.19 ms
```

High-risk instances were isolated by attaching a restrictive quarantine security group.

The screenshot shows the AWS CloudWatch Log management interface for the log group '/aws/lambda/AutoQuarantine-EC2'. The 'Log events' section displays the following log entries:

- 2026-02-02T20:43:10.841Z: INIT\_START Runtime Version: python3.12.v102 Runtime Version ARN: arn:aws:lambda:us-east-1:runtime:e70763d79865080d2f1c073367ca436566c24809fa1c13d7389d70a9f239
- 2026-02-02T20:43:10.726Z: START RequestId: 994a5c21-140e-425b-841d-7ab2efbf863 Version: \$LATEST
- 2026-02-02T20:43:10.727Z: EVENT: {"details": {"resource": {"instanceDetails": {"InstanceId": "i-0x10253d0683a84"}}}}
- 2026-02-02T20:43:10.727Z: EVENT: {"details": {"resource": {"instanceDetails": {"InstanceId": "i-0x10253d0683a84"}}}}
- 2026-02-02T20:43:10.727Z: Instance From Finding: i-0x10253d0683a84
- 2026-02-02T20:43:10.727Z: Instance From Finding: i-0x10253d0683a84
- 2026-02-02T20:43:11.092Z: ENI: eni-01119ec5c8df96d1
- 2026-02-02T20:43:11.092Z: ENI: eni-01119ec5c8df96d1
- 2026-02-02T20:43:11.676Z: Quarantine applied.
- 2026-02-02T20:43:11.696Z: END RequestId: 994a5c21-140e-425b-841d-7ab2efbf863
- 2026-02-02T20:43:11.696Z: REPORT RequestId: 994a5c21-140e-425b-841d-7ab2efbf863 Duration: 967.63 ms Billed Duration: 1649 ms Memory Size: 128 MB Max Memory Used: 98 MB Init Duration: 681.19 ms

This limited lateral movement and reduced impact.

### 5.3 Alerting and Logging

All remediation actions were logged in CloudWatch Logs.

SNS notifications were sent to security administrators for visibility and escalation.

## 6. Event Monitoring and Detection

EventBridge rules were configured to monitor:

- Unauthorized API calls
- Privilege escalation attempts - Security group modifications
- Malware indicators
- Network anomalies

Each rule triggered automated actions or alerts.

CloudWatch metrics were monitored to validate automation reliability.

## 7. EC2 Hardening and Golden AMI Management

### 7.1 Baseline Hardening

A dedicated EC2 instance was used to build hardened baselines.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP
test-instance	i-0a310253d6b683a84	Running	t2.micro	2/2 checks passed	View alarms	us-east-1d	-	-	-

Hardening actions included:

- Installing latest security patches
- Configuring Systems Manager Agent
- Disabling password-based SSH
- Enabling time synchronization
- Attaching least-privilege IAM roles

## 7.2 Golden AMI Creation

After validation, hardened instances were converted into Golden AMIs.

The screenshot shows the AWS Management Console with the EC2 service selected. The left sidebar has sections for Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area is titled "Amazon Machine Images (AMIs) (1) Info". It displays a table with one row, representing the "Golden-AMI-v1" AMI. The columns are: Name (Golden-AMI-v1), AMI ID (ami-0a30a4be947f50c4c), Source (879381257906/Golden-AMI-v1), Owner (879381257906), Visibility (Private), Status (Available), and Creation date (2026-02-02 22:04 GMT+0). There are buttons for Recycle Bin, EC2 Image Builder, Actions, and Launch instance from AMI.

Name	AMI ID	Source	Owner	Visibility	Status	Creation date
Golden-AMI-v1	ami-0a30a4be947f50c4c	879381257906/Golden-AMI-v1	879381257906	Private	Available	2026-02-02 22:04 GMT+0

These images became the standard deployment baseline.

### 7.3 Validation

New instances launched from Golden AMIs were validated to ensure security controls were preserved.

## 8. Automated Patch Management

AWS Systems Manager Patch Manager was configured to maintain operating system updates.

Patch baselines were defined to:

- Scan instances daily
- Apply critical patches weekly
- Reboot automatically when required

Compliance reports were reviewed regularly.

#### 9. Baseline Compliance Enforcement

AWS Config was enabled for continuous configuration evaluation.

Key rules included:

- No public IPs on EC2 instances
- Mandatory EBS encryption
- Restricted SSH access
- Mandatory MFA for IAM users

Non-compliant resources were flagged and remediated where possible.

#### 10. Testing and Validation Framework

The framework was validated through controlled simulations, including:

- GuardDuty sample findings
- Insecure security group creation
- EventBridge trigger testing
- Lambda execution review
- Quarantine verification
- Patch compliance testing
- AMI validation

These tests confirmed end-to-end automation effectiveness.

#### 11. Challenges and Lessons Learned

Challenges encountered included:

- Private instance connectivity issues
- Missing IAM roles for SSM
- Sample findings using dummy IDs
- SNS permission errors

These issues were resolved through policy tuning, network adjustments, and service integration improvements.

They strengthened operational understanding.

## 12. Outcomes and Impact

This implementation delivered:

- Reduced manual security workload
- Faster incident containment
- Standardized hardened baselines
- Improved compliance visibility
- Enhanced monitoring coverage
- Reduced attack surface

Overall security maturity improved significantly.

## 13. Professional Impact

This engagement strengthened practical skills in:

- Security automation
- Incident response
- Infrastructure hardening - Compliance enforcement
- Cloud-native security operations

It demonstrates the ability to design, implement, and manage enterprise-scale security automation frameworks on AWS.