PRJ-SEC-008: Automated Threat Detection & Response — AWS Console Deployment Guide (v2 - Corrected)

Author: Mo Suleiman | **Date:** 2026-03-13

1. Introduction

This guide provides a complete, step-by-step walkthrough for deploying **PRJ-SEC-008**, a serverless **Security Orchestration, Automation, and Response (SOAR)** platform on AWS, using only the AWS Management Console. This updated version includes critical fixes identified during live deployment to ensure a successful and testable workflow.

The platform uses **Amazon GuardDuty** for threat detection, **AWS Security Hub** for finding aggregation, **Amazon EventBridge** for event filtering, and an **AWS Lambda** function to execute a remediation playbook.

GRC & Business Context

This architecture directly supports key governance, risk, and compliance (GRC) objectives by automating security controls.

Framework	Control ID	Description
NIST CSF	DE.AE-4	**Event Analysis:** Events are analyzed to understand attack targets and methods. The Lambda playbook is triggered by specific, high-severity event patterns.
ISO 27001	A.16.1.7	**Information Security Incident Management:** Procedures are established to respond to security incidents. This project automates the initial response (containment).
PCI-DSS	11.4	**Intrusion Detection:** Use intrusion detection systems to monitor all traffic. GuardDuty serves as the IDS, and this platform automates the response.
Business Value	-	Reduces Mean Time to Respond (MTTR) from hours to seconds, minimizes the blast radius of a breach by immediately containing threats, ensures consistent and auditable response actions, and frees up security analysts to focus on high-value investigation.

---

2. Deployment Steps

This deployment is broken down into 5 main parts, which should be followed in order.

Part 1: Enable Detection Services

1. **Enable Amazon GuardDuty:**
    - Navigate to the **GuardDuty** service in the AWS Console.
    - If it's your first time, click **Get Started**, then **Enable GuardDuty**.
2. **Enable AWS Security Hub:**
    - Navigate to the **Security Hub** service.
    - If it's your first time, click **Go to Security Hub**, then **Enable Security Hub**. This will automatically begin aggregating findings from GuardDuty.

Part 2: Create Remediation Resources

1. **Create the Isolation Security Group:**
    - Navigate to the **VPC** service, then select **Security Groups** from the left menu.
    - Click **Create security group**.
    - **Name:** `prj-sec-008-isolation-sg`
    - **Description:** `Deny all traffic for isolated instances.`
    - **VPC:** Select your default VPC.
    - **Inbound rules:** Delete the default rule. There should be no inbound rules.
    - **Outbound rules:** Delete the default rule. There should be no outbound rules.
    - Click **Create security group**. **Copy the Security Group ID** (e.g., `sg-0123...`) to a text editor. You will need it later.
2. **Create the SNS Topic for Alerts:**
    - Navigate to the **Simple Notification Service (SNS)** service.
    - Select **Topics** from the left menu, then click **Create topic**.
    - Select **Standard** type.
    - **Name:** `prj-sec-008-remediation-alerts`
    - Click **Create topic**.
    - In the topic's page, click **Create subscription**.
    - **Protocol:** Select **Email**.
    - **Endpoint:** Enter your email address.
    - Click **Create subscription**. **Go to your email inbox and click the link from AWS to confirm the subscription.**
    - **Copy the Topic ARN** (e.g., `arn:aws:sns:...`) to your text editor.

Part 3: Create the Lambda IAM Role

1. Navigate to the **IAM** service.

2. Select **Roles** from the left menu, then click **Create role**.
3. **Trusted entity type:** Select **AWS service**.
4. **Use case:** Select **Lambda**.
5. Click **Next**.
6. On the permissions page, search for and add the `AWSLambdaBasicExecutionRole` policy.
7. Click **Next**.
8. **Role name:** `prj-sec-008-lambda-remediation-role`
9. Click **Create role**.
10. **Add Inline Policy:**
    - Find and click on the role you just created.
    - On the **Permissions** tab, click **Add permissions -> Create inline policy**.
    - Select the **JSON** tab and paste the following policy. This grants the Lambda the *least-privilege* permissions it needs to perform its tasks.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2IsolationActions",
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyInstanceAttribute",
                "ec2:DescribeInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:CreateSnapshot",
                "ec2:DescribeVolumes"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SNSPublishAlert",
            "Effect": "Allow",
            "Action": "sns:Publish",
            "Resource": "PASTE_YOUR_SNS_TOPIC_ARN_HERE"
        },
        {
            "Sid": "SecurityHubUpdateFinding",
            "Effect": "Allow",
            "Action": "securityhub:BatchUpdateFindings",
            "Resource": "*"
        }
    ]
}
```

- **IMPORTANT:** Replace `PASTE_YOUR_SNS_TOPIC_ARN_HERE` with the SNS Topic ARN you copied earlier.
- Click **Next**.
- **Policy name:** `prj-sec-008-lambda-remediation-policy`
- Click **Create policy**.

Part 4: Create the Lambda Function

1. Navigate to the **Lambda** service.
2. Click **Create function**.

3. **Author from scratch**.
4. **Function name:** `prj-sec-008-remediation-playbook`
5. **Runtime:** Select **Python 3.12**.
6. **Architecture:** Keep `x86_64`.
7. **Permissions:** Expand **Change default execution role**, select **Use an existing role**, and choose the `prj-sec-008-lambda-remediation-role` you created.
8. Click **Create function**.
9. **Add Code:**
    - In the **Code source** editor, delete the default code and paste in the full Python script from `lambda/remediation_playbook.py` (provided in the project zip).
    - Click the **Deploy** button to save your code.
10. **Add Environment Variables:**
    - Go to the **Configuration** tab, then **Environment variables**.
    - Click **Edit**.
    - Click **Add environment variable** twice:
        - **Key:** `ISOLATION_SG_ID`, **Value:** (Paste the Security Group ID you copied earlier)
        - **Key:** `SNS_TOPIC_ARN`, **Value:** (Paste the SNS Topic ARN you copied earlier)
    - Click **Save**.
11. **Increase Timeout:**
    - Go to the **Configuration** tab, then **General configuration**.
    - Click **Edit**.
    - Set the **Timeout** to **1 minute**.
    - Click **Save**.

Part 5: Create the EventBridge Rule

1. Navigate to the **Amazon EventBridge** service.
2. Select **Rules** from the left menu, then click **Create rule**.

3. **Name:** `prj-sec-008-ec2-remediation-trigger`
4. **Event bus:** `default`
5. **Rule type: Rule with an event pattern**.
6. Click **Next**.
7. **Event source: AWS events or EventBridge partner events**.
8. Scroll down to **Event pattern**. Select **Custom patterns (JSON editor)** and paste the following **corrected** pattern. This version is less strict and correctly matches findings imported via the API/CLI for testing.

```
{
  "source": ["aws.securityhub"],
  "detail-type": ["Security Hub Findings - Imported"],
  "detail": {
    "findings": {
      "Severity": {
        "Label": ["HIGH", "CRITICAL"]
      },
      "RecordState": ["ACTIVE"],
      "Workflow": {
        "Status": ["NEW"]
      },
      "Resources": {
        "Type": ["AwsEc2Instance"]
      }
    }
  }
}
```

9. Click **Next**.
10. **Select a target:**
    - **Target type: AWS service**.
    - **Select a target: Lambda function**.
    - **Function:** Select the `prj-sec-008-remediation-playbook` function.
11. Click **Next**, then **Next** again, then **Create rule**.

---

3. Testing & Validation (Corrected Method)

The original testing method using GuardDuty sample findings is unreliable because they contain fake resource IDs. The correct way to test the full workflow is to inject a custom finding into Security Hub via the AWS CLI. This simulates a real event and triggers the entire automated chain.

Prerequisites for Testing

- **AWS CLI Installed and Configured:** You need the AWS CLI installed on your local machine with credentials configured for your AWS account.
- **Test EC2 Instance:** Launch a standard Amazon Linux 2 EC2 instance in your default VPC. Give it a name tag of `test-compromised-host`. Note its **Instance ID** (e.g., `i-0123...`).

Step 1: Get Your Account and Instance IDs

Run these commands in your terminal to get the required IDs.

```
# Get your AWS Account ID
aws sts get-caller-identity --query Account --output text

# Get your test instance ID
aws ec2 describe-instances \
  --filters "Name=tag:Name,Values=test-compromised-host" \
  --query "Reservations[0].Instances[0].InstanceId" \
  --output text
```

Step 2: Inject a Custom Finding into Security Hub

Copy the JSON below into a file named `finding.json`. **Replace `YOUR_ACCOUNT_ID` and `YOUR_INSTANCE_ID` with the values from Step 1.**

```
[
  {
    "SchemaVersion": "2018-10-08",
    "Id": "prj-sec-008-test-finding-001",
    "ProductArn": "arn:aws:securityhub:us-east-
1:094869897684:product/094869897684/default",
    "GeneratorId": "prj-sec-008-manual-test",
    "AwsAccountId": "094869897684",
    "Types": ["TTPs/Initial Access/UnauthorizedAccess:EC2-SSHBruteForce"],
    "CreatedAt": "2026-03-13T00:00:00Z",
    "UpdatedAt": "2026-03-13T00:00:00Z",
    "Severity": {"Label": "HIGH"},
    "Title": "Simulated: Brute force attack on compromised EC2 host",
    "Description": "PRJ-SEC-008 test finding to validate automated
remediation workflow.",
    "RecordState": "ACTIVE",
    "Workflow": {"Status": "NEW"},
    "Resources": [
      {
        "Type": "AwsEc2Instance",
        "Id": "arn:aws:ec2:us-east-1:094869897684:instance/i-
0c3e653dce225398d",
        "Region": "us-east-1"
      }
    ]
  }
]
```

Now, run the `batch-import-findings` command:

```
aws securityhub batch-import-findings --findings file://finding.json
```

> ***Note:*** *If you need to re-run the test, you must change the* `Id` *field in* `finding.json` *(e.g., to* `prj-sec-008-test-finding-002` *) to avoid deduplication by Security Hub.*

Step 3: Monitor the Workflow

Immediately after running the command above, use these commands in separate terminal windows to watch the automation happen in real-time.

Terminal 1: Watch Lambda Logs

```
aws logs tail /aws/lambda/prj-sec-008-remediation-playbook --follow
```

Terminal 2: Watch the Instance's Security Group (Replace `YOUR_INSTANCE_ID`)

```
watch -n 5 "aws ec2 describe-instances --instance-ids i-0835ee17640946f34 --
query 'Reservations[0].Instances[0].SecurityGroups' --output table"
```

Expected Outcome (within ~30 seconds)

1. The `batch-import-findings` command will return a success message.
2. The Lambda logs will show the event being received and processed.
3. The `watch` command will show the instance's security group change from its original SG to the `prj-sec-008-isolation-sg`.
4. You will receive an email alert from SNS with the full details of the remediation.

This confirms the entire end-to-end workflow is functioning correctly.