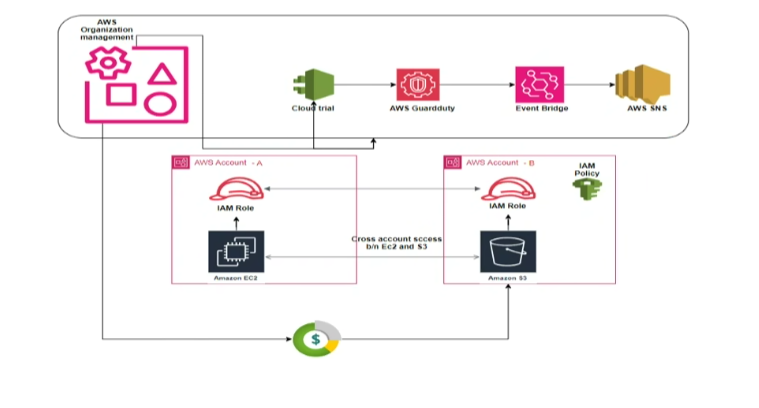
**Problem Statement:** A company is reviewing its AWS account security policies. The company has staff members in different countries and wants to monitor its AWS accounts for **unusual behavior** that is associated with an IAM identity. The company wants to send a **notification** to any staff member for whom unusual activity is detected. The company needs to manage its multiple AWS accounts to capture **consolidated billing** to help the finance team; store all billing reports in Account-B's S3 bucket. Along with this Account-A's EC2 instances need permissions to access Account-B's S3 bucket to develop the final report.



**Step 1: Monitor AWS Accounts for Unusual IAM Activity**

We'll begin by enabling **AWS CloudTrail** and **Amazon GuardDuty** across your AWS accounts to detect unusual activity.

**1.1 Enable AWS CloudTrail**

CloudTrail helps you log and monitor AWS API calls, including those made by IAM users.

1. **Go to the AWS Management Console** and navigate to **CloudTrail**.
2. **Create a new Trail** :
   * Name the trail (e.g., "SecurityTrail").
   * Apply it to **all regions** to ensure you're logging IAM activities across all regions.
   * Choose an **S3 bucket** to store CloudTrail logs.
   * Enable **Log file validation** for security.
3. **Enable CloudWatch Logs** integration for real-time monitoring.

**1.2 Enable Amazon GuardDuty**

GuardDuty detects suspicious activity in your AWS environment (e.g., compromised IAM credentials).

1. In the **GuardDuty** console, enable it for the **management account** of your AWS Organization.
2. After enabling GuardDuty, invite all member accounts (if in a multi-account setup) to join the GuardDuty configuration.
3. **GuardDuty will start analyzing CloudTrail logs**, VPC Flow Logs, and DNS logs for unusual behavior.

**Step 2: Send Notifications for Unusual Activity**

**2.1 Set Up SNS Topic for Notifications**

Amazon SNS (Simple Notification Service) will allow us to send alerts when unusual IAM activity is detected.

1. **Go to the AWS Management Console**, and navigate to **Amazon SNS**.
2. **Create a Topic**:
   * Choose **Standard** type.
   * Name the topic (e.g., UnusualIAMActivityAlerts).
   * Create the topic.
3. After creating the topic, **create a subscription** to send notifications via email, SMS, or other endpoints:
   * In the SNS console, choose the topic you created.
   * Click **Create subscription**.
   * Select the **protocol** (Email, SMS, etc.).
   * Enter the **endpoint** (e.g., your email address).
4. **Confirm the subscription**: You'll receive a confirmation request via the provided endpoint (e.g., email). Click the confirmation link.

**2.2 Create CloudWatch Event Rule for GuardDuty Findings**

Now, we’ll create a CloudWatch Event rule that triggers when GuardDuty detects unusual behavior, which will send a notification via SNS.

1. **Go to the AWS Management Console**, and navigate to **CloudWatch**.
2. In the left navigation pane, choose **Rules**.
3. **Create a new rule**:
   * **Event Source**: Select **Event Source** as **GuardDuty**.
   * **Event Pattern**: Use a custom event pattern for specific GuardDuty findings, like IAM-related findings:

Policy:

{

"source": ["aws.guardduty"],

"detail-type": ["GuardDuty Finding"],

"detail": {

"type": ["UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration"]

}

}

* + **Targets**: Choose **SNS Topic** as the target.
  + **Select the SNS topic** you created (UnusualIAMActivityAlerts).

1. **Save the rule**. Now, any unusual IAM activity detected by GuardDuty will trigger an SNS notification.

**Step 3: Manage Multiple AWS Accounts with Consolidated Billing**

**3.1 Set Up AWS Organizations**

AWS Organizations allows you to consolidate billing for multiple accounts.

1. **Log in to the AWS Management Console** as the root user of the **management account** (Account-B, which will manage billing).
2. Navigate to **AWS Organizations**.
3. **Create an Organization**:
   * Click on **Create organization** and select the **Consolidated billing** feature.
   * This creates your Organization and establishes Account-B as the **management account**.
4. **Invite other AWS accounts** (e.g., Account-A) to join the organization:
   * Go to the **Accounts** tab.
   * Click **Add account** → **Invite account** and enter the email address or AWS account ID of Account-A.
   * Account-A will receive an invitation to join your Organization.

**3.2 Enable Consolidated Billing**

Once Account-A has accepted the invitation, you can view billing information for all accounts under the management account.

1. In the AWS Organizations console, navigate to the **Billing** section.
2. Under **Consolidated Billing**, you’ll now see a summary of costs for all accounts (including Account-A).
3. To allow **Account-A** to access billing reports:
   * **Grant billing access** to the member accounts by enabling the EnableIAMAccess setting in the **Billing Preferences**.

**3.3 Enable AWS Cost and Usage Reports**

We’ll now configure **AWS Cost and Usage Reports (CUR)** to store billing data in an S3 bucket in Account-B.

1. Go to the **Billing and Cost Management** dashboard in Account-B.
2. In the left navigation pane, click on **Cost & Usage Reports**.
3. **Create report**:
   * Name the report (e.g., ConsolidatedBillingReports).
   * Choose the options you want, such as **time granularity** (daily or hourly), and enable **Resource IDs** for detailed reports.
4. **Configure S3 bucket** for the reports:
   * Create or select an S3 bucket in Account-B to store the billing reports.
   * Make sure the bucket is encrypted and has the right permissions (we'll update cross-account access later in Step 5).
5. Finalize the report setup and wait for AWS to start generating billing reports into the S3 bucket.

**Step 4: Store Billing Reports in Account-B's S3 Bucket**

**4.1 Create or Select the S3 Bucket in Account-B**

1. **Go to the AWS S3 console** in Account-B.
2. **Create a new S3 bucket** (if you haven't done so already) for storing billing reports:
   * Name the bucket (e.g., account-b-billing-reports).
   * Select the appropriate **region**.
   * Ensure that **Bucket Versioning** and **Server-Side Encryption (SSE)** are enabled to meet security and compliance requirements.
3. Once the bucket is created, confirm that **AWS Cost and Usage Reports** are configured to be delivered to this bucket

**Step 5: Cross-Account S3 Bucket Access**

**5.1 Configure Cross-Account Access to the S3 Bucket**

To allow **Account-A's EC2 instances** to access the S3 bucket in **Account-B**, we need to:

1. **Modify the S3 bucket policy** in Account-B.
2. **Create an IAM role** in Account-A that grants EC2 instances the necessary permissions.

**5.2 Modify the S3 Bucket Policy in Account-B**

1. **Go to the S3 console** in Account-B.
2. Select the bucket where billing reports are stored (account-b-billing-reports).
3. Under **Permissions**, click on **Bucket Policy**.
4. Add the following bucket policy to allow access from Account-A's IAM role:

json

Copy code

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::Account-A-ID:role/Account-A-EC2-Role"

},

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::account-b-billing-reports/\*"

}

]

}

* Replace Account-A-ID with the actual AWS account ID for Account-A.
* Replace Account-A-EC2-Role with the IAM role name (we’ll create this role next).

This policy allows the IAM role in Account-A to retrieve objects from the S3 bucket in Account-B.

**5.3 Create an IAM Role in Account-A for EC2 Instances**

1. **Go to the IAM console** in Account-A.
2. **Create a new IAM role**:
   * Choose **EC2** as the trusted entity.
   * Attach the following permissions policy, which grants read access to Account-B's S3 bucket:

Policy: { "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::account-b-billing-reports/\*" } ] }

**Attach the role** to the EC2 instances in Account-A that need to access the billing reports.

**5.4 Accessing the S3 Bucket from EC2 Instances in Account-A**

1. SSH into your EC2 instance in **Account-A**.
2. Verify the role is attached by running:

**curl** [**http://169.254.169.254/latest/meta-data/iam/security-credentials/**](http://169.254.169.254/latest/meta-data/iam/security-credentials/)

1. Once the role is verified, you can use the AWS CLI on the EC2 instance to retrieve object from Account-B’s S3 bucket**:**

**aws s3 cp s3://account-b-billing-reports/report-name /local/path**