IAM Implementation and Enforcement Starter Template



# STEP 0: Overview Instructions

|  |  |
| --- | --- |
| **Project Overview** | In this project, you will apply skills learned through this course to implement and enforce identity access management within AWS. You will implement a role structure with policies that will be evaluated and enforced. You will be provided with an access control matrix that outlines the appropriate roles, resources, and actions to be implemented and specific organizational requirements for policies. Next, you will evaluate the access control matrix and restrictions to ensure that each role has the appropriate policies and permissions implemented with the principle of least privilege. Upon implementation of the policies and permissions, you will create an AWS Config rule that will detach a policy that does not meet the organizational requirements. You’ll finish this project with a visualization of organizational roles and policies. |
| **Project Steps** | 1. Based on the access control matrix, evaluate the provided IAM policies to ensure that they are implemented with the principle of least privilege. 2. Update the policies with the permissions that meet the requirements of the access control matrix and organizational requirements 3. Attach the IAM policies to the proper IAM roles based on the access control matrix 4. Assume the IAM roles and validate the access that has been granted 5. Update Lambda code for the AWS config rule with the organizational resource restrictions 6. Execute AWS Config re-evaluation to validate that the bad policy that was created is marked as non-compliant 7. Create an architecture diagram to visualize the organizational role structure along with the permissions granted and denied. |

# STEPS 1 & 2: Permissions and Policies

|  |  |
| --- | --- |
| **Submission 1 Instructions:** | Provide a screenshot of each updated policy with the proper statements removed to meet the organizational requirements, and if needed, copy the json policy into a text file and take a screenshot to ensure that the whole policy can be reviewed. |
| **Submission 1a:** *Submit a screenshot of the JSON for your:* enterprise-analyst-policy |  |
| **Submission 1b:** *Submit a screenshot of the JSON for your:* enterprise-developer-policy |  |
| **Submission 1c:** *Submit a screenshot of the JSON for your:* enterprise-finance-policy |  |
| **Submission 1d:** *Submit a screenshot of the JSON for your:* enterprise-restrictions-policy |  |

# 

# STEPS 3 & 4: IAM Roles

|  |  |
| --- | --- |
| **Submission 2 Instructions:** | Attach the appropriate IAM policies to the appropriate IAM roles.   * The following roles should have the below policies attached.   + enterprise-analyst-role     - enterprise-analyst-policy     - enterprise-restrictions-policy   + enterprise-developer-role     - enterprise-developer-policy     - enterprise-restrictions-policy   + enterprise-finance-role     - enterprise-finance-policy     - enterprise-restrictions-policy |
| **Submission 2a:** *Submit a screenshot of your IAM Policies for:* enterprise-analyst-role*.* |  |
| **Submission 2b:** *Submit a screenshot of your IAM Policies for:* enterprise-developer-role*.* |  |
| **Submission 2c:** *Submit a screenshot of your IAM Policies for:* enterprise-finance-role*.* |  |
| **Submission 3 Instructions:** | While assuming each role in the AWS console, take a screenshot of the results of accessing each service or taking action. Ensure that the role name is present in the screenshot so that the role and permissions can be validated.  The following roles should have screenshots with the proofs that the allowed/denied permissions work as expected:   * + enterprise-analyst-role     - Screenshot with access denied to non\_obfuscated.txt object in s3     - Screenshot of downloaded obfuscated.txt object     - Screenshot of successfully uploaded analyst.txt object with supported tags to developer bucket   + enterprise-developer-role     - Screenshot of a CloudWatch Metric     - Screenshot of security group under EC2     - Screenshot of successfully uploaded developer.txt to developer bucket     - Screenshot of successfully downloaded developer.txt from developer bucket   + enterprise-finance-role     - Screenshot of a current usage in billing console |
| **Submission 3a:** *Submit a screenshot for your enterprise-analyst-role:*  Access denied to non\_obfuscated.txt object in s3. |  |
| **Submission 3b:** *Submit a screenshot for your enterprise-analyst-role:*  Downloaded obfuscated.txt object |  |
| **Submission 3c:** *Submit a screenshot for your enterprise-analyst-role:*  Successfully uploaded analyst.txt object with supported tags to developer bucket. |  |
| **Submission 3d:** *Submit a screenshot for your enterprise-developer-role:*  Accessing a CloudWatch Metric. |  |
| **Submission 3e:** *Submit a screenshot for your enterprise-developer-role:*  Accessing security group under EC2 |  |
| **Submission 3f:** *Submit a screenshot for your enterprise-developer-role:* Successfully uploaded developer.txt to developer bucket |  |
| **Submission 3g:** *Submit a screenshot for your enterprise-developer-role:* Successfully downloaded developer.txt from developer bucket |  |
| **Submission 3h:** *Submit a screenshot for your enterprise-finance-role:* Current usage in billing console |  |

# 

# STEPS 5 & 6: AWS Config

|  |  |
| --- | --- |
| **Submission 4 Instructions:** | Provide a screenshot of the Lambda console with the code updated to ensure that the RESTRICTED\_RESOURCE variable includes the “arn:aws:s3:::super-secret-bucket” string in the list. |
| **Submission 4:** *Submit a screenshot (or multiple screenshots, if necessary) of your*  Lambda code. |  |
| **Submission 5 Instructions:** | Manually remediate the AWS Config rule to indicate that policy that breaks the enterprise restrictions is marked as non-compliant within AWS Config. Provide a screenshot of the ConfigRulePolicyEnforcement rule with the bad-policy-that-breaks-enterprise-restrictions policy marked as non-compliant in the AWS Config service. |
| **Submission 5:** *Submit screenshot for AWS Config Rule marking bad-policy-that-breaks-enterprise-restrictions as non-compliant* |  |

# STEP 7: Organizational Role and Policy Visualization

|  |  |
| --- | --- |
| **Submission 6 Instructions:** | Using the draw.io organizational role diagram provided in the resources, provide a screenshot of the updated diagram to ensure that each resource defined in each policy has the appropriate permission associated with the resource.  Using the permissions defined in the “Permissions To Match to Resources in Policy” table, drag each permission to the right of the proper resource in the policy. These permissions should reflect those defined in the Access Control Matrix. |
| **Submission 6:** *Submit a screenshot of your*  organizational role and policy visualization. |  |

# Optional: Standout Suggestions

# 

|  |  |
| --- | --- |
| **Standout Suggestion #1 Instructions** | Using the optional AWS Config diagram, make the connections between the resources to provide an overview for how AWS config is used to monitor and alert on changes. |
| **Standout Submission #1** *Submit a screenshot (or multiple screenshots) of your*  updated condition statements. |  |
| **Standout Suggestion #2 Instructions** | Update lambda code to deliver a notification via SNS to alert on the non-compliant policy, subscribe email to provided SNS topic, and display email being sent from SNS. |
| **Standout Submission #2**   *Submit a screenshot (or multiple screenshots) of your*  updated Lambda code. |  |

# 