Assignment 11

New Attempt

- Due Dec 8, 2024 by 11:59pm
- Points 100
- Submitting a file upload
- File Types zip and pdf

Due: Sunday, December 8, 2024 (11:59 PM). Please submit your assignment on Canvas as a PDF.

IAM Auditing and Security Monitoring

In this assignment, you will learn about credential report generation in AWS identity and access management, and how a role can assume another role. You will use AWS SDK to audit some of the IAM Controls of **AWS CIS foundation benchmark 1.2 (https://docs.aws.amazon.com/securityhub/latest/userguide/securityhub-cis-controls.html)**. AWS Identity and Access Management (IAM) is a free AWS service that provides mechanisms for authenticating and securely controlling access. You can have access to AWS services through the AWS management console, AWS Command Line Tools, AWS SDKs, and IAM HTTPS API. In this assignment, you will use the AWS console and AWS SDKs. The following video introduces AWS IAM and how to generate the credential report:

https://www.youtube.com/watch?v=3y596T1eH_8 (https://www.youtube.com/watch?v=3y596T1eH_8)

(https://www.youtube.com/watch?v=3y596T1eH_8)

As you already experienced in previous assignments, the policy is an object that can be attached to users, groups of users, roles, or AWS resources, providing an access policy to those entities. We have three types of policies:

- AWS managed policy which is the recommended type to be used.
- Custom policy which users can define based on their specific requirements.
- Inline policy which is a type of policy directly attached to the user (not recommended).

An IAM user represents a user or service. As a best practice, it is recommended to create a group and provide access for users through groups. Note that we should not attach a policy to users directly. To authorize users to have specific access, first, you need to create a policy with sufficient privileges, and then attach the policy to a group. All the users inside the group can have all the access that is provided by all policies attached to the group. The IAM role provides temporary

credentials to access resources. We can create a role and attach a relative policy to the role to provide access. Roles can be assumed by another account and it is one of the most common solutions to provide cross-account access.

Part 1: IAM Using Console

In this part, first, you will create policies, users, and groups, and then, we will explore the IAM credential report.

Task 1 (5%): Open the IAM console-> choose Policies, and then choose to Create policy then you can use Visual editor or JSON file to create a custom policy. Create a policy with the least privilege strategy which can get the IAM credential report (name it IAM-Auditor-Policy).

Task 2 (5%): Create the following users:

- User1 without any policy or permission with console access
  - Enable MFA for User1 by your admin user and log in with user1 while MFA is enabled.
- User2 without any policy or permission with console access
  - Generate one access key for User2.
- User3 without any policy or permission with only console access

Task 3 (3%): Create a user group named IAM-Auditor-Group and attach the custom policy of task 1 to this user group.

Task 4 (2%): Add User3 to the IAM-Auditor group.

Task 5 (5%): Login with User3. Go to IAM service -> Setting and get the IAM credential report. Report your findings about each user from the generated credential report.

Task 6 (5%): When you are still logged in through User3, add User2 to the IAM-Auditor group. Explain your observation. In case of failure, describe the steps needed to be taken in order for User3 to add User2 to the IAM Auditor.

Hint: Pay attention to the least privilege policy.

Part 2: Security Automation

Automation is the key solution to reduce the workload of a security team. People make mistakes and it is something inevitable. Automation helps to avoid human error. In this part, we will create automation for auditing some CIS AWS Foundations benchmark controls. We will create a lambda function that assumes a role that has enough access to remediate some of CIS AWS foundation benchmark controls v1.2.0. For remediation purposes, we can use AWS SDK which is one of the most powerful tools to utilize different services.

The following videos introduce different ways that users can access AWS and how to assume a role by another role from a different account. Note that In our case, we assume the role from the same account.

https://www.youtube.com/watch?v=tDchQ0nv7Q4 (https://www.youtube.com/watch?v=tDchQ0nv7Q4)

(https://www.youtube.com/watch?v=tDchQ0nv7Q4)

https://www.youtube.com/watch?v=n1r9Fp7GKvk (https://www.youtube.com/watch?v=n1r9Fp7GKvk)

(https://www.youtube.com/watch?v=n1r9Fp7GKvk)

Task 7 (5%): Create a lambda function using Python 3.9 as a runtime, and x86_64 as an Architecture.

Task 8 (10%): Create a role “IAM-Auditor-role” that can be assumed by a role of Lambda function, and attach the IAM-Auditor policy to this role.

Task 9 (20%): In the Lambda function you created in Task 7:

- Define a function to assume the role that you created in task 8 (“IAM-Auditor-role”).
- Define a function to generate and get the IAM Credential Report.
- Provide a script that uses the first function to assume the IAM-Auditor role and the second function to generate and get the IAM credential report.

Task 10 (15%): In this task, you need to use the IAM credential report generated in the previous task to do a security audit. We are going to audit the AWS CIS controls 1.1 and 1.12.

- **CIS 1.1:** Based on the best practices the root user should not be used for daily activity. Create a function that reports the last time the root account has been used.
- **CIS 1.2:** Ensure multi-factor authentication (MFA) is enabled for all IAM users who have a console password. Now, create a function that reports all users with MFA disabled.
- **CIS 1.12:** We should ensure no access key is attached to the root account. Report if there is any key attached to the root account.

Task 11 (10%): In this task, you need to create a simple text report from task 10 and use the SNS service to send the report to your own email address. To do so, go to the Amazon Simple Notification Service (SNS) console -> create an SNS topic, and subscribe your email address to the topic. Then add SNS publish privilege to the Lambda function role. Finally, use Boto3 SNS publish function to send notifications.

Task 12 (5%): AWS Event bridge enables scheduling events to trigger AWS services such as the Lambda function. Explain the steps that need to be taken to receive a daily report from a script that

you created in Task 9.

Part 3: Security Intelligence

Monitoring is one of the important tasks of a security team. As a security team member, in case of incidents or during the auditing process, you need to analyze logs and investigate activities and API calls. For example, you need to identify:

- Which user creates a specific resource
- Which user changed the specific configuration of the EC2 instance or security group
- When the change happens

AWS CloudTrail is a service that logs all API calls from AWS Console, AWS SDK, and AWD CLI. AWS CloudTrail aggregates and stores all the logs in the S3 buckets. AWS CloudTrail history is where you can access and monitor the last 90 days' logs of all users' activities and API calls. If you need to query logs older than 90 days, you must use the Athena Service. Athena will create a table from the data stored inside the S3 bucket (s3 bucket which AWS CloudTrail uses to store logs) and it makes it possible to use SQL syntax to query information from the table.

Task 13 (10%): Use AWS CloudTrail-> event history to query the last time the root user logged in.