**IAM security best practices**

**AWS Management Console access and programmatic access using the AWS Command Line Interface or other custom tools.**

Securing AWS console access is crucial to protect your resources and data stored on the AWS platform. Here are some best practices to secure your AWS console access:

|  |  |
| --- | --- |
| **Secure AWS Console Access** | **Description** |
| Strong and Unique Passwords: | * Use a combination of upper and lowercase letters, numbers, and special characters. * Aim for a password length of at least 12 characters. * Avoid easily guessable passwords or common dictionary words. |
| Enable multi-factor authentication (MFA): | * Sign in to the AWS Management Console using your root account or an IAM user with sufficient permissions. * Go to the IAM service and select the user for which you want to enable MFA. * Choose the “Security credentials” tab and navigate to the “Multi-factor authentication (MFA)” section. * Follow the instructions to configure MFA using either a virtual MFA device (such as Google Authenticator) or a hardware token. |
| Periodically update passwords: | * Set a policy for password rotation, such as changing passwords every 90 days. * Remind users to update their passwords regularly. |
| Avoid password reuse: | * Educate users about the importance of not reusing passwords across multiple accounts. * Encourage the use of password managers to securely store and manage unique passwords. |
| Implement a password policy: | * Create a password policy in IAM that enforces strong password requirements, such as minimum length, complexity, and password expiration. * Associate the password policy with IAM users to ensure compliance. |
| Avoid using the AWS root account: | * Instead of using the root account, create individual IAM users with appropriate permissions for day-to-day operations. * Sign in to the AWS Management Console using your root account. * Go to the IAM service and create a new IAM user. * Assign necessary permissions and generate access keys for programmatic access, if required. |
| Grant minimum necessary permissions: | * Follow the principle of least privilege when assigning permissions to IAM users. * Analyze the specific actions and resources each user requires and grant only those permissions. * Regularly review and update IAM policies to remove unnecessary permissions. |
| Implement IAM groups | * Group IAM users based on common access requirements. * Assign policies to groups instead of individual users for easier management and scalability. * Add or remove users from groups as needed, and the permissions will be automatically applied. |
| Rotate IAM user access keys: | * Regularly rotate access keys for IAM users to reduce the risk of compromised credentials. * Sign in to the AWS Management Console using your root account or an IAM user with sufficient permissions. * Go to the IAM service, select the user, and navigate to the “Security credentials” tab. * Generate a new access key and update it in the appropriate applications or services. |
| Use IAM roles for applications and services: | * Create IAM roles with the necessary permissions and trust relationships for applications and services. * Assign the roles to EC2 instances, Lambda functions, or other AWS resources instead of using long-term access keys. * Configure the application or service to assume the IAM role programmatically. |
| Enable MFA for IAM users | * Sign in to the AWS Management Console using your root account or an IAM user with sufficient permissions. * Go to the IAM service and select the user for which you want to enable MFA. * Choose the “Security credentials” tab and navigate to the “Multi-factor authentication (MFA)” section. * Follow the instructions to configure MFA using either a virtual MFA |

**Temporary credentials such as IAM roles**

This document create an IAM roles Access the AWS Management Console and navigate to the IAM service When defining the role, specify an expiration time for the temporary credentials.

|  |  |
| --- | --- |
| **Role** | **Descriptions** |
| **Create an IAM Role:** | * **Access the AWS Management Console and navigate to the IAM service.** * **Click on "Roles" in the left-hand menu and select "Create Role."** * **Choose the appropriate trusted entity (AWS service, another AWS account, or a web identity) for role access.** |
| **Define Role Permissions:** | * **Attach managed policies or create custom policies to grant the minimum necessary permissions.** * **Adhere to the principle of least privilege, ensuring that the role has access only to required resources and actions.** |
| **Set Role Expiration Time:** | * **When defining the role, specify an expiration time for the temporary credentials.** * **Select an appropriate duration based on the expected usage timeframe.** |
| **Enable MFA for Role Assumption:** | * **Require multi-factor authentication (MFA) for users assuming the role.** * **Configure MFA settings in the IAM user policies and role's trust policy.** |
| **Review and Monitor Role Activity:** | * **Regularly review role activity in AWS CloudTrail logs.** * **Set up CloudWatch alarms to detect and alert on any unusual or unauthorized role assumption activities.** |

**Identity Federation and Access management**

This document, Identity Federation and Access Management as AWS Set up a trust relationship between on-premises IdP and AWS and Create an AWS account or access an existing AWS account

**Identity Federation**

|  |  |
| --- | --- |
| **Identity Federation** | **Description** |
| 1. Set up a trust relationship between on-premises IdP and AWS. | Establish trust between your organization's on-premises Identity Provider (IdP) and AWS. This involves configuring the IdP to recognize and trust AWS as a relying party. |
| 2. Configure IdP to issue SAML/OIDC assertions. | Configure your on-premises IdP to issue SAML (Security Assertion Markup Language) or OIDC (OpenID Connect) assertions for authenticated users. These assertions contain user identity information that will be used for federated access to AWS services. |
| 3. Create/update IAM role in AWS. | In the AWS Management Console, create an IAM (Identity and Access Management) role or update an existing role. This IAM role establishes trust with your IdP and allows users to assume the role for accessing AWS services. |
| 4. Map attributes in assertions to IAM roles/permissions. | Define the mapping of attributes from the assertions issued by your IdP to IAM roles or permissions in AWS. This mapping ensures that the federated user's attributes are correctly mapped to the appropriate IAM roles or permissions for access control. |
| 5. Test the federation setup. | Test the federation setup by logging in to your IdP using valid credentials and attempting to access AWS services using the federated login option. Verify that the user is successfully authenticated and authorized to access the expected AWS resources. |

**Access management**

|  |  |
| --- | --- |
| **Access management** | **Description** |
| 1. Create an AWS account or access an existing AWS account. | If you don't have an AWS account, create one on the AWS website. If you already have an account, sign in to the AWS Management Console using your account credentials. |
| 2. Set up AWS Identity and Access Management (IAM). | In the AWS Management Console, navigate to the IAM service. IAM enables you to securely control access to AWS services and resources. Set up IAM by creating an IAM user with administrative permissions or using the root account with caution. |
| 3. Define IAM users, groups, and roles. | Create IAM users for individuals who need access to AWS resources. Organize users into logical groups for easier management and assignment of permissions. Define IAM roles for different use cases, such as cross-account access or EC2 instance roles. |
| 4. Assign appropriate permissions to IAM entities. | Define and attach IAM policies to users, groups, and roles to grant permissions for specific AWS services and resources. IAM policies can be based on predefined policies, custom policies, or a combination of both. Follow the principle of least privilege, granting only the necessary permissions required for users to perform their tasks. |
| 5. Monitor and manage access using IAM policies and roles. | Regularly review and refine IAM policies to ensure they align with changing business needs and security requirements. Use IAM tools and features such as IAM Access Analyzer and IAM credential reports to monitor and manage access to AWS resources effectively. Implement multi-factor authentication (MFA) for added security. |

**AWS Management programmatic access**

|  |  |
| --- | --- |
| **Rules** | **Descriptions** |
| Use AWS Identity and Access Management (IAM): | * Create an IAM user for each individual who requires programmatic access. * Define granular permissions for each IAM user based on the principle of least privilege. * Enable multi-factor authentication (MFA) for IAM users. * Regularly review and update IAM user permissions as needed. * Disable or delete unused IAM users to maintain a clean and secure IAM environment. |
| Enable Multi-Factor Authentication (MFA): | * Access the AWS Management Console and navigate to the IAM service. * Select the IAM user for whom you want to enable MFA. * Go to the "Security credentials" tab and click on "Manage" for MFA. * Follow the prompts to set up MFA using a virtual or hardware device. * Test the MFA setup by signing in with the IAM user and providing the MFA code when prompted. |
| Implement strong password policies: | * Access the AWS Management Console and navigate to the IAM service. * Go to the "Account settings" tab and click on "Password policy." * Define a password policy that meets your organization's security requirements, including password complexity rules, length, and rotation intervals. * Enable password expiration and force users to create new passwords periodically. * Educate IAM users on creating and maintaining strong passwords. |
| Use IAM Roles and Temporary Security Credentials: | * Identify the applications, services, or users that require programmatic access. * Create an IAM role with the necessary permissions for the intended resource. * Define the trusted entities that can assume the IAM role (e.g., specific IAM users, AWS services). * Configure the duration for which the temporary security credentials will be valid. * Update your applications or services to assume the IAM role and retrieve temporary security credentials. |
| Regularly review and rotate access keys: | * Access the AWS Management Console and navigate to the IAM service. * Select the IAM user for whom you want to manage access keys. * Go to the "Security credentials" tab and locate the Access Keys section. * Remove any unused or unnecessary access keys. * Generate a new access key and update the necessary applications or services with the new credentials. * Set a reminder to periodically rotate access keys and follow the same procedure |