# IAM IN PRACTICE

*"How do I set up IAM for my organization?"*

## Overview

AWS Identity and Access Management (IAM) is a powerful and flexible web service for controlling access to AWS resources. IAM enables customers to leverage the agility and efficiency of the cloud while maintaining secure control of their organization's AWS infrastructure. IAM Administrators new to AWS can be sometimes overwhelmed by the options available as they face competing goals: securing the environment while quickly enabling new users to accomplish their jobs. Further complicating the task, the initial controls they implement must grow and adapt without disrupting productivity as the company navigates its path to the cloud

This document provides best practices and guidance to help IAM administrators quickly establish an initial set of controls that protect their infrastructure, empower users, and allow for growth and change in their organization's use of AWS. This document assumes a working knowledge of how to configure the IAM service.

## Guidelines and Best Practices

While it can take weeks or months to lay out a full access-control strategy for an organization, there are some universal best practices to apply immediately to ensure security in the cloud.[1] These practices are important for both new organizations and established organizations with mature security processes, as they help ensure that the initial strategy stays relevant as teams and resources grow in size and complexity.

- **Create groups that reflect organizational roles, not technical commonality.** Two users that require the same technical permissions to perform two different roles in an organization should be assigned to two different groups based on their roles rather than a single group based on the technical permission. The function of the roles will likely change over time, requiring different technical permissions. If the users are grouped by role, the permission changes can be highly targeted to the changing role–reducing the risk of inadvertently granting new privileges to extra users.

  For example, a Data Warehouse Admin and a Data Scientist might both need the ability to launch an Amazon Redshift cluster. In time, the organization decides to move their analytics from Amazon Redshift to Amazon Elastic MapReduce (EMR). If both employees were mapped to a group called *LaunchRedshiftCluster*, then the IAM Administrator would have to either a) increase the permissions of all the users in the group or b) break the group in two and reassign the users manually (introducing the risk of error). With two groups, the IAM Administrator merely removes the Amazon Redshift privilege from the Data Scientist group and adds the Amazon EMR privilege; the Data Warehouse Admin group remains unchanged. The latter option decreases overall effort, minimizes the blast radius of the change and reduces risk of human error.

- **Have a documented process for removing unnecessary users and credentials.** Know what steps are required to remove a user and write them down. Ideally, write a script to reduce the chance of error.

- **Enable MFA for privileged users.** Use MFA to lock down administrative accounts (e.g., the root account, IAM administrators, and system administrators). MFA adds a *something you have* factor to the *something you know* factor of authentication, reducing the risk of a security breach. This can be implemented using a hardware device or a virtual MFA app.[2]

- **Rotate credentials regularly.** Credentials are secrets and the longer a secret exists the more likely it is to be compromised. Rotating credentials regularly helps mitigate this risk. Using Temporary Security Tokens, for example through Amazon Elastic Compute Cloud (Amazon EC2) roles removes the need for persistent credentials entirely.

- **Use managed policies rather than group or user policies.** Although IAM policies can be directly associated with a user or group, use the newer capability of managed policies. Managed policies are reusable because they are decoupled from groups, making it easier to achieve technical commonality across different roles (as described above) without misusing groups.

[1] For a complete list of IAM best practices, refer to AWS documentation: http://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html

[2] For more information, see http://aws.amazon.com/iam/details/mfa/#Virtual_MFA_Applications

- **Make policies granular.** Add privileges to a group using multiple granular policies rather than one giant policy. For example, a Data Scientist might need access to write to Amazon Simple Storage Service (Amazon S3) to upload raw data, and also to Amazon EMR to launch clusters. Create a managed policy for S3 and a separate managed policy for Amazon EMR rather than combining the privileges into a single policy. This will help promote policy reuse and make it easier to manage permissions.
- **Use EC2 roles rather than access keys.** An access key is a secret and must be stored in a location that is secure but also can be easily referenced during the bootstrap process. Once acquired, the key must be securely stored on the EC2 instance yet retrievable in order to access AWS resources. These conflicts create a perpetual risk in the DevOps process—a problem only exacerbated by the need to regularly rotate keys. EC2 Roles eliminate the need for an access key, and the underlying technology of Temporary Security Tokens eliminates the need for key rotation.
- **Use roles rather than user credentials to grant cross-account access.** The safest and easiest way to grant access to users in different AWS accounts is to create a role with specific privileges and grant other accounts the right to assume that role. The administrator for the other account can then allow specific IAM users to switch to the role as necessary to use its permissions on a temporary basis. Using roles eliminates the responsibility of creating, managing, rotating, and securely delivering access keys for individual users from different accounts.
- **Use conditions to make policies more granular.** IAM conditions provide a wealth of possibilities to refine policies. These include locking down administrator accounts to only work from a specific IP address range, limiting developer permissions to specific subnets, granting a permission for a specific time window, and much more. Learn the various conditions available and leverage them to create better policies.

## First Steps

The following sections describe how to start using IAM, including how to secure an AWS account, create IAM users, groups, and policies, and how to prepare for future growth and change in AWS use.

### Securing the IAM Administrator Account

Before granting users the access they need, complete the following steps to move forward swiftly and securely.

1. Log in with the root account credentials, and configure baseline security settings according to the AWS Secure Initial Account Setup Solution Brief.
2. Use the IAM console to create a customized console login address. A custom console address will not only obscure the account number, but it will also provide a more user-friendly URL for users to use when accessing the AWS console.
3. Create a password policy.
4. Create an IAM Administrators group and assign it the managed policy *IAMFullAccess.*
5. Create an IAM Administrator user and add it to the IAM Administrators group.
6. Create a password for the IAM Administrator user.
7. Add virtual MFA to the IAM Administrator user.
8. Log out of the account, and log back in using the custom console URL and the new IAM Administrator credentials.

All new users and processes should now be set up using the new IAM Administrator account. Lock away the root account credentials and hardware device until needed to perform an account-level action that requires root credentials.

### Creating Users and Groups

With new administrator credentials configured, it's time to apply the general best practices. This sounds simple in concept, but can be challenging in actual execution–especially when starting out. Here are some steps to help get started.

1. Identify the first person to be granted access to AWS infrastructure. Explicitly state any associated business roles for that person. These business roles should be very granular and a person can fulfill several business roles.
2. Create an IAM group for each business role.

3. Identify the AWS permissions required to fulfill the tasks of each business role. Create managed policies for each task and assign them to the appropriate group.
4. Create an IAM user for the person and assign it to the groups representing the appropriate business roles. Assign a user name and password to the account. If this person needs to use the CLI or other tools to access the AWS environment, create an access key as well.
5. Complete these steps for all subsequent users, mapping their roles to existing groups and creating new groups if needed. Watch for situations where the second user fills only part of an existing role and consider splitting the associated group into two groups.

The following table shows examples of how some typical users might be mapped to groups and permissions.

| Employee | Group/Business Role | Permissions |
|---|---|---|
| Accounts Payable Clerk | Review Bills | Service: AWS Billing; Action: View*; ARN:* (no conditions) |
| Comptroller | Review Bills | Service: AWS Billing; Action: View*; ARN:* (no conditions) |
| Data Scientist | Run Data Experiment | Service: Amazon Elastic Map Reduce; Action: *; ARN: * (no conditions)<br>Service: Amazon S3; Action: Get*, List*, Put*; ARN: Input and output Buckets (no conditions) |
| Data Administrator | Prepare Data for Analysis | Service: Amazon S3; Action: Get*, List*, Put*; ARN: Input and output Buckets (no conditions) |
| Developer | Test Newly Developed Features | Service: Amazon EC2; Action: *Instances, *Volume, Describe*, CreateTags; Condition: Dev Subnets only |
| Tester | Run Test Scripts | Service: Amazon EC2; Action: *Instances, *Volume, Describe*, CreateTags; Condition: Test Subnets only |
| IT Manager | Review Deployed Infrastructure | Managed Policy: ReadOnlyAccess (Note – this policy will change as new services get added with no management needed) |

It is common for there to be some iteration when initially setting up policies as it is not always immediately apparent which permissions are needed to accomplish all of a given task. If the IAM administrator and user are in the same room when the user first attempts to implement their use case, then the administrator can quickly add missing privileges to the policy as they are discovered.

## Looking to the Future

Customers' perceptions of the cloud usually change drastically as they start to experience the significant changes it can bring to an IT environment. The guidelines outlined in this document will prepare an organization for common growth scenarios, such as:

- Federating existing users – As the number of employees using AWS increases, many companies choose to base authentication on their internal user directory rather than replicate all employees in IAM. AWS supports this federation through SAML and OIDC. Following the best practices of creating groups that reflect business roles and managing the policies separately, an organization will be ready to map those groups to business roles as defined through SAML attributes and assign the same privileges appropriately.
- Creating multiple accounts – Companies with extensive use of AWS often open multiple linked AWS accounts to help segregate billing, limit access, and minimize the blast radius of any security issues.[3] If administrators follow best practices to group users by business role rather than by technology, they can more easily map each group to the appropriate account. Since polices are managed independently of groups and users, administrators can create roles for cross-account access using the same policies that they already created.

## Resources

| | |
|---|---|
| IAM Documentation | http://aws.amazon.com/documentation/iam/<br>AWS webpage with links to detailed IAM documentation including the full IAM User Guide, and relevant CLI and API reference documentation |
| IAM Best Practices to Live By | https://www.youtube.com/watch?v=_wiGpBQGCjU<br>2015 AWS re:Invent session recording |
| AWS Secure Initial Account Setup | https://d0.awsstatic.com/aws-answers/AWS_Secure_Account_Setup.pdf<br>AWS Solution Brief on best practices for securing a new account |