IAM & AWS CLI

IAM: users and groups

Identity and access management, it is also a global service because it is needed in order to start up your AWS console after all if it is only available in one particular region then other region would not be able to work at all!

Root account is created by default when you register for an AWS account. They should not be used or shared. Only used for making other account with permissions because root account has access to everything.

Users are people within your organization and can be grouped.

Groups can only contain users not other groups.

Users don't have to belong to a group, and user can belong to multiple group.

Why groups and users?

Users or groups can be assigned a JSON document called policies which contains permission what a user/group are allowed to do.

IAM policy define the permissions of the users.

By default, a user are not allowed to do anything by default, you apply the least privilege principle: don't give more permissions than a user need to complete a task. For example if a user only need to work on a EC2 instance, then you only give permission to work on that EC2 instance, you don't give it more permission to say work on all of the EC2 instances.

User that's in a group will inherit the permission policy assigned to the group.

Tags

You can add tags to almost everything, they will help you identify and track down the resources that you created based on the tags you add to them.

Logging in as IAM user

To login as an IAM user, you would need either the account ID or if you created an alias for the account ID, you can use that. Then you would enter in the IAM user's username and password.

For root user you do not need to use the account ID, you can just log in using the email address.

IAM policies inheritance

If a user is part of a group then it will inherit the policy that is attached to the group. If a user is part of multiple group then it will inherit all of the policy that the user is part of.

You can also create something called **inline** policy for a user that is not part of a group. They are attached to the user directly not inherited from group.

IAM policy structure

A JSON document that consists of:

- Version: policy language version
- Id: An identifier for the policy, this is optional
- Statement: one or more individual statements, this is required

Further more, for each statement it consists of:

- Sid: an identifier for the statement, optional
- Effect: whether the statement is allow or denied access
- Principal: account/user/role to which this policy applied to
- Action: list of actions this policy allows or denies
- Resource: list of resources to which the actions applied to
- Condition: conditions for when this policy is in effect, optional



IAM password policy

Strong password = higher security for your account

You can set up policy to prevent password re-use

Also need multi factor authentication (MFA). You need a secondary device that you own to verify your identity.

Password + security device you own = good and strong security

Even if you lose your password or got stolen you account won't be compromised, because the hacker will need the same security device that you only have.

MFA device options in AWS

- Virtual MFA device: Google authenticator, Authy. Support multiple tokens on a single device. Another app that you install to authenticate yourself
- Universal 2nd factor security key. YubiKey by Yubico. It is a physical device.
- Hardware key fob MFA Device
- Hardware key fob MFA device for AWS GovCloud

AWS access keys

When you need to access management console it is protected by password + MFA.

CLI access is protected by access keys that you can generate from your account.

Software Developer Kit is also protected by access keys, meaning you need a valid access key in order to use CLI and SDKs.

Access keys is generated via AWS console. They should be treated as secrets, do not share access keys!

AWS CLI

A tool that allows you to interact with AWS service using commands in your command-line shell.

You can have direct access to public APIs to AWS services.

Alternative to management console.

AWS SDK

A library that is language specific that allow you access to manage AWS services programmatically. Supports lots of languages.

Mobile SDKs, and IoT device SDKs are also available.

AWS CLI is actually built using the Python AWS SDK.

IAM Roles for services

Some AWS service will need to perform actions on your behalf. You will then need to assign permissions to AWS services with IAM roles. They give the AWS resource the permission for a short period of time by assuming into the role, and have the appropriate permission to carry out whatever they need to do on AWS.

The IAM roles are just like users, but they are to be assigned to non-physical people but assigned to other AWS services.

EX: We make a EC2 instance and it might want to perform some action on AWS for example read some resources from the AWS. Then we will haveto assign an IAM role to give it permission to access AWS, otherwise, the EC2 instance does not have any permission to do so.

Common roles: EC2 instance roles, lambda function roles, roles for CloudFormation. But there are tons of services that supports IAM roles.

IAM Security Tools

IAM Credentials report: A report that lists all your account's users and the status of their various credentials

IAM Access Advisor: Shows the service permissions granted to a user and when those services were last accessed. You can then modify their permission according to their usage. Remember **principal of least privilege**.

IAM best practices

Do not use root account except when you are using it to make other AWS account.

One physical user = One AWS user

Assign users to groups and give groups policy

Use a strong password policy + MFA!

Create and use roles for giving permissions to AWS services, otherwise, they cannot access AWS

Use access keys for SDK + CLI

IAM Advanced

Revision #5
Created 2023-02-06 20:23:36 UTC by Tamarine
Updated 2023-07-27 01:54:33 UTC by Tamarine