

IAM Advanced

Organization

Let you manage multiple AWS account at the same time. There is one main account that's the management account, other account are member accounts.

Billing will all be sent to the management account. You get pricing benefits from using all services across the member accounts which is nice.

Share reserved instances and savings plans across accounts. Member account creation can be automated.

Organization unit

You first have Root organization unit (This is the most outer OU) in which your management account lives in. Then you can create other organization unit within the root one, kinda like sub groups. And within those subgroups that's where you can place member accounts. You can nest the organization unit within each other.

Benefit

You can enable CloudTrail for all accounts and have a central logging S3 account. Central CloudWatch logs as well.

Service control policies (SCP). IAM policies applied to OU or account to restrict users and roles. Applies to everything but the management account, so even if you deny a certain resource to the management account it will not apply. For SCP you can specify an allow list or block list. Allow list you explicitly allow actions while denying everything. Block list you allow everything then you deny certain actions.

You would apply SCP to each OU, and the account would have they inherit from their parent OU unless there is an explicit deny, even if there was an authorize. An deny would be a NO regardless if there is an authorize to that specific account.

IAM Condition

Used to conditionally apply IAM statement to a IAM policy. It is a field under the Statement key, with name "Condition". There are several conditions available for use.

1. aws:SourceIP: Don't allow access to those who has IP that are listed
2. aws:RequestedRegion: Don't allow access if you're not in certain region
3. ec2:ResourceTag: Allow access to those EC2 instances with the corresponding tag.
4. aws:MultiFactorAuthpresent: Only can carry out these access if they have MFA

IAM for S3

When writing IAM policy for S3 there is two kinds you can write.

Bucket level permission, defines which bucket you have access to.

Object level permission, defines which object you have access to, you would need to do / after the bucket to pick out those object that the policy applies to because you are doing it on object level.

Resource policies and aws:PrincipalOrgID

Restrict resource to only users under the specified organization.

IAM roles vs Resource-based policies

When a user, AWS service assumes a role it gives up the original permission that they had before.

If you use resource-based policies, policy that is attached to the resource not the user/role, then it doesn't have to give up the original permission that they had.

Lambda, SNS, SQS, CloudWatch, API Gateway uses resource-based policy, the resource defines who can access it and what kind of API it can make

Kinesis stream, ECS task use IAM role, so it will have to assume the role which define what it can do with what resources.

IAM permission boundaries

IAM permission boundary is like another IAM policy document that defines the upper bound that the IAM role / user's permission can get. Then you attach the actual IAM permission which must be equal or less than the permission defined in the boundary. If the permission you give to the user is outside of the boundary then it will receive no permission!

This can be used with combination with organization SCP. The junction of SCP, permission boundary, and IAM policy defines what the user can do.

The way the permission is evaluated is as follow:

1. **If there is an explicit deny then permission is denied. Even if there is an allow!**
2. Then it goes to SCP, if there isn't an explicit allow then it is denied, implicitly
3. If it has the resource based policy then it is evaluate

4. Then IAM that the role / user has
5. Then permission boundary is evaluated
6. Session policies is also evaluated

So all these policies do come into play and to allow / deny access

No explicit deny or explicit allow, then it is denied.

IAM Identity Center

Single sign on for all your AWS accounts. One login to access all applications.

One login into multiple AWS accounts basically you get to choose who to be logged in as. Identity center can be from AWS or from third party.

After you login you have permission set to define what the user can or cannot do.

You can define fine-grained permissions and assignment of policies.

Directory services

Microsoft active directory: A database that contains objects, users, accounts, computer, printer, file shares, and security groups.

There is centralized security. A domain controller to do verification on other machines that logs in.

AWS Directory services

AWS Managed microsoft AD: Mimics microsoft active directory so you can amange users. You can use an on-premise AD as another database for user logins, this is called "trust". The integration is out of the box.

AD connector: Proxy that redirect to on-premise AD, support MFA.

Simple AD: Standalone AD that you can use on the cloud.

However, if you have your own self-managed directory then you need to create two-way trust relationship using AWS Managed Microsoft AD. OR you can use AD connector.

AWS Control tower

Easy to set up a secure and compliant multi-account AWS environment. It uses AWS organization to make the accounts.

Guardrails are used to governance for your accounts that's set up from control tower.

Preventive guardrail uses SCP to restrict access

Detective guardrail uses AWS Config to identify non-compliance.

Basically it abstract away SCP and AWS Config for you if you don't want to deal with it.

Revision #2

Created 26 February 2023 04:02:09 by Tamarine

Updated 26 February 2023 17:40:56 by Tamarine