2 IAM

Identity Policies

- Identity Policies are JSON documents that define allow or deny permissions for an identity (user, group, or role) in AWS.
- statements are the main element of an identity policy. A statement has the following elements:
  - Effect: Whether the statement allows or denies access.
  - Action: The specific action or actions that the statement allows or denies.
  - Resource: The resource or resources to which the statement applies.
  - Condition: The conditions under which the statement applies.

Order of evaluation:

1. Explicit Deny
2. Explicit Allow
3. Default Deny (implicit) --> deny always wins

- Example:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FullAccess",
      "Effect": "Allow",
      "Action": ["s3:*"],
      "Resource": ["arn:aws:s3:::*"]
    },
    {
      "Sid": "DenyExampleBucket",
      "Effect": "Deny",
      "Action": ["s3:*"],
      "Resource": ["arn:aws:s3:::examplebucket,
arn:aws:s3:::examplebucket/*"]
    }
  ]
}
```

Inline Policies vs Managed Policies

- Managed Policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. They are the recommended way to manage permissions.
  - AWS Managed Policies: AWS managed policies are managed by AWS and are designed to provide permissions for many common use cases. AWS managed policies also provide you with the flexibility to attach the same managed policy to multiple identities.

  - Customer Managed Policies: You can create customer managed policies and attach them to multiple users, groups, and roles in your AWS account. Customer managed policies provide more precise control over your policies than AWS managed policies.
- Inline Policies are policies that are directly attached to a user, group, or role. They are usesd for exceptions to the managed policies. Example: you have a managed policy that allows access to all S3 buckets, but you want to deny access to a specific bucket for a specific user. You can create an inline policy for that user that denies access to that specific bucket.

IAM Users

IAM users are an identity used for anything requiring long-term access AWS access e.g. users, applications or service accounts. -> If you can select 1 thing, a named thing, then 99% of the time it's an IAM user identity you have to use.

Authentication = prove to IAM that Principal is who he claims to be. IAM users can be authenticated using:

- Username and password
- Access keys

*** EXAM: MAX 5,000 IAM Users per account *** *** EXAM: IAM User can be a member of 10 groups maximum *** System with more than 5,000 identities? Can't use IAM user for each identity. IAM Roles & Identity Federation fixes this (more later)

ARN

Amazon Resource Names (ARNs) are used to ***uniquely identify AWS resources within any AWS account.***

```
arn:partition:service:region:account-id:resource
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

ex: arn:aws:s3:::examplebucket (refrences the S3 bucket) ex: arn:aws:s3:::examplebucket/* (refrences the objects in the S3 bucket) --> difference between the bucket and the objects in the bucket is important for policies

Diffrence between `::` and `:*:`

- `::` omit the resource/service name because it is not needed to uniquely identify the resource. eg S3 bucket names are globally unique, so no need to define the region or account id in the bucket arn.
- `:*:` wildcard but can't be omitted, all resources of the type are included.

IAM Groups

IAM groups are containers for IAM users, they exist to make organizing large sets of IAM users easier. Groups are not identities, they are a way to attach policies to multiple users at once. You can't login using a group.

IAM Roles

Use cases roles:

- Give temporary permissions to an IAM user, group, or role. eg. Customer Service Rep needs temporary write access to an S3 bucket.
- External accounts (Facebook, MS Active Directory, etc) can't be used directly in AWS. Instead, you can create a role in AWS and grant the external account access to the role.
- If organisation has more than 5000 users (IAM user limit) you can't use IAM users, you can use roles instead (through ID federation).
- Web Identity Federation: Authenticate users of your application (1,000,000+) using a web identity provider like Amazon, Facebook, Google, etc.
- Cross-Account Access: Grant access to resources in one account to another account. Or give access to the whole account (using Organization).

Service linked roles

A service-linked role is a unique type of IAM role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf.

- Service-linked roles simplify the process of setting up a service because you don't have to manually add permissions for the service to complete actions on your behalf.

Service-linked role vs service role

Service-linked roles are different from service role:

- A ***service role*** is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM.
- A ***service-linked role*** is a type of service role that is linked directly to an AWS service. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

https://docs.aws.amazon.com/IAM/latest/UserGuide/using-service-linked-roles.html

--> you can't delete a service linked role until it's no longer required

PassRole

PassRole is an IAM permission that allows an IAM entity to pass a role to an AWS service. This permission is required when you create or update a service that uses a role. The service needs the permission to assume the role. The service-linked role for the service includes the permissions that the service needs to assume the role.

AWS Organizations

Manage multiple AWS accounts in a single organization. It allows you to:

- Create new AWS accounts
- Add existing accounts to the organization
- Organize accounts into groups

- Apply policies to groups of accounts
- Use consolidated billing
- Control access to AWS services and resources (SCP)

The main account is the MANAGEMENT account, it's the account that created the organization. It has full control over the organization and all accounts in the organization. It's the only account that can invite other accounts to join the organization. You can only have one MANAGEMENT account per organization.

The other accounts are MEMBER accounts. They are the accounts that are part of the organization. They can be moved between OUs, have SCPs applied to them, etc.

SCP

Service Control Policies are a feature of AWS Organizations which allow **restrictions** to be placed on MEMBER accounts in the form of boundaries.

- SCPs can be applied to the organization(root), to OU's or to individual accounts.
- Member accounts can be effected, the MANAGEMENT account cannot. SCPs have no effect on the management account. (not even SCP's on organization root level)
- SCPs DON'T GIVE permission - they just control what an account CAN and CANNOT grant via identity policies. The actual permissions are still granted via IAM policies.
- Allow list vs Deny list - By default SCPs allow all actions and you create a deny list that limits actions. (recommended because less policy overhead). You can also deny all actions and create an allow list (more policy overhead).

CloudWatch logs

CloudWatch Logs is a public service, usable from AWS or on-premise and even from other public cloud platforms. CloudWatch is a regional service.

- It is used to store, monitor and access logging data from AWS resources or any other application (using CloudWatch Agent or SDK). AWS Integrations: EC2, Lambda, CloudTrail, VPC Flow Logs, Route53, etc.
- It is often the default place where AWS Services can output their logging too. And then to be used by other services like CloudTrail, (a verifier: CloudWatch Alarms, Lambda, etc.)
- It can generate metrics based on logs = metric filters
- A log stream is a sequence of log events from the same source (eg an EC2 instance)
- A log group is a container for multiple log streams for the same type of logging (eg all logs from an EC2 instance)
- on log group you apply configuration settings like retention policy and permissions. These settings then apply to all log streams in the log group.
- on log group you can also create metric filters and alarms. These settings then apply to all log streams in the log group. A metric filter increments a metric when a log event matches a filter pattern. And a metric can have associated alarms, these alarms can be used to either notify the admin or trigger an action by other AWS services or external system.

*** EXAM: CloudWatch is a REGIONALLY resilient service ***

CloudTrail

Logs API calls/activities as CloudTrail events. It is enabled by default in AWS accounts and stores events for 90 days in Event History. To modify the default settings or to store logs for longer periods, you can create a trail. It's very often used to diagnose security or performance issues, or to provide quality account level traceability.

Events can be:

- ***Management events***:
  - control plane operations = operations on ressources in AWS accoun
  - enabled by default
- ***Data events***:
  - data plane operations = ressource operations performed on or in AWS ressource
  - not enabled by default and costs money
- ***Insights events***:
  - insights events are generated by CloudTrail Insights, a feature that helps you identify and respond to unusual activity in your AWS account.

By default only management events are logged, you have to enable data events logging. Because data events logging can generate a lot of logs, it's recommended to enable it only when needed.

CloudTrails is a **regional service**. There are 2 types of trails:

- ***Single region trail***: logs events from one region (if not us-east-1, global services are not logged)
- ***Multi-region trail***: logs events from multiple regions (if not us-east-1, global services are logged only if enabled) Global service events (IAM, STS, CloudFront) are logged where they are created (in us-east-1), so you have to create a multi-region trail and enable global service logging. Or you can create a sigle region trail in us-east-1 to log global service events.

Logging is NOT real time, there is a delay of a few minutes. If question on exam on real time logging, the answer is NOT CloudTrail.

It can be configured to store data indefinitely in S3 or CloudWatch Logs.

AWS Control Tower

WS Control Tower offers a straightforward way to set up and govern an AWS multi-account environment, following prescriptive best practices. AWS Control Tower orchestrates the capabilities of several other AWS services, including AWS Organizations, AWS Service Catalog, and AWS IAM Identity Center (successor to AWS Single Sign-On), to build a landing zone in less than an hour. Resources are set up and managed on your behalf.

AWS Control Tower orchestration extends the capabilities of AWS Organizations. To help keep your organizations and accounts from drift, which is divergence from best practices, AWS Control Tower applies preventive and detective controls (guardrails). For example, you can use guardrails to help ensure that security logs and necessary cross-account access permissions are created, and not altered.

- Think of Control Tower as another evolution of AWS Organizations

CT: Parts of Control Tower

- ***Landing Zone:*** Multi-account environment (what most people interact with). SSO/ID Fed, Centralized Logging, Auditing
- ***Guard Rails:*** Detect/Mandate rules/standards across all accounts
- ***Account Factory:*** Automates and standardizes new account creation
- ***Dashboard:*** Single page oversight of entire environment/org

CT: Landing Zone

A well-architected multi-account environment. Home Region is the one you deploy into and is always available.

- Built with AWS Orgz, Config, CloudFormation
- Security Organizational Unit (OU): Log Archive and Audit Accounts (CloudTrail / Config Logs)
- Sandbox OU: For testing and for less rigid security situations
- IAM ID Center (prev. AWS SSO): SSO, multiple accounts, ID Federation (using on-premise identity stores)
- Monitoring and Notifications: CloudWatch and SNS
- Service Catalog: End User account provisioning

CT: Guard Rails

- Rules for multi-account governance.
- Three types: Mandatory, Strongly Recommended, Elective
- GR functions in 2 Ways: -- Preventative: Stop you from doing things (AWS Org SCP). Enforced or not enabled. Eg. Allow/deny regions, disallow bucket policy changes (prevent things from happening) -- Detective: Compliance checks (AWS Config Rules) for maintaining Best Practices. Types: Clear, In Violation, Not Enabled. Eg. Detect / confirm if CloudTrail has been enabled on your account (identify things happening)

CT: Account Factory

- Automated account provisioning. Cloud Admins or End Users (w/ appropriate permissions)
- Guardrails automatically added
- Give admin permissions to a named user (IAM ID)
- Standard Account & Network standard configuration. Eg. IP addressing using VPC
- Allows accounts to be closed or repurposed
- Can be fully integrated with a business's Software Development Lifecycle

AWS Control Tower (continued)

- Under Control Tower, AWS Organizations creates two Organizational Units (OU): Foundational/Security and Sandbox OU's
- Within each Foundational OU and Security OU, two accounts are created in each: Audit and Log Archive accounts
- Account Factory: Create/configure/delete accounts using Templates: Account Baseline (template) and Network Baseline (template)
- For Guardrails, AWS uses AWS Congfig and SCP
- Drift: Divergence from best practices