Identity, Entitlement, and Access Management(IAM)

By :Ahsan Farooqui

Presentation Roadmap-IAM

- What is IAM?
- Terminologies
- IAM Standards in Cloud Computing
- How Federated IDM Works?
- Managing Users and Identities for Cloud Computing
- Hub & Spoke vs Free-Form
- Architecture and Process Decisions
- Authentication & Credential
- Entitlement and Access Management
- Cloud Impact on entitlements and Access Management
- Privilege Users Management
- AWS-IAM

Overview or What is IAM?

Gartner defines IAM as “the security discipline that enables the right individuals to access the right resources at the right times for the right reasons.”

- AWS
  - **AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.**

Terminologies in IAM?

- Entity: the person or “thing” that will have an identity. It could be an individual, a system, a device, or application code.
- Identity: the unique expression of an entity within a given namespace. An entity can have multiple digital identities,
  - such as a single individual having a work identity (or even multiple identities, depending on the systems), a social media identity, and a personal identity.
  - For example, if you are a single entry in a single directory server then that is your identity.
- Identifier: the means by which an identity can be asserted. For digital identities this is often a cryptological token. In the real world it might be your passport.

Terminologies in IAM?

- **Attributes: facets of an identity. Attributes can be relatively static (like an organizational unit) or highly dynamic (IP address, device being used, if the user authenticated with MFA, location, etc.).**
- **Persona: the expression of an identity with attributes that indicates context. For example, a developer who logs into work and then connects to a cloud environment as a developer on a particular project. The identity is still the individual, and the persona is the individual in the context of that project.**
- **Role: identities can have multiple roles which indicate context. “Role” is a confusing and abused term used in many different ways. For our purposes we will think of it as similar to a persona, or as a subset of a persona. For example, a given developer on a given project may have different roles, such as “super-admin” and “dev”, which are then used to make access decisions.**

Terminologies in IAM?

- Authentication: the process of confirming an identity. When you log in to a system you present a username (the identifier) and password (an attribute we refer to as an authentication factor).Also known as Authn.
- Multi factor Authentication (MFA): use of multiple factors in authentication. Common options include
  - one-time passwords generated by a physical or virtual device/token (OTP),
  - out-of- band validation through an OTP sent via text message, or confirmation from a mobile device,
  - biometrics, or plug-in tokens.
- Access control: restricting or granting access to a resource. Access management is the process of managing access to the resources.

Terminologies in IAM?

- Authorization: allowing an identity access to something (e.g. data or a function). Also known as Authz.
- Entitlement: mapping an identity (including roles, personas, and attributes) to an authorization. The entitlement is what they are allowed to do, and for documentation purposes we keep these in an entitlement matrix.
- Federated Identity Management: the process of asserting an identity across different systems or organizations. This is the key enabler of Single Sign On and also core to managing IAM in

- Authoritative source: the “root” source of an identity, such as the directory server that manages employee identities.
- Identity Provider: the source of the identity in federation. The identity provider isn’t always the authoritative source, but can sometimes rely on the authoritative source, especially if it is a broker for the process.
- Relying Party: the system that relies on an identity assertion from an identity provider.

IAM Standards in Cloud Computing

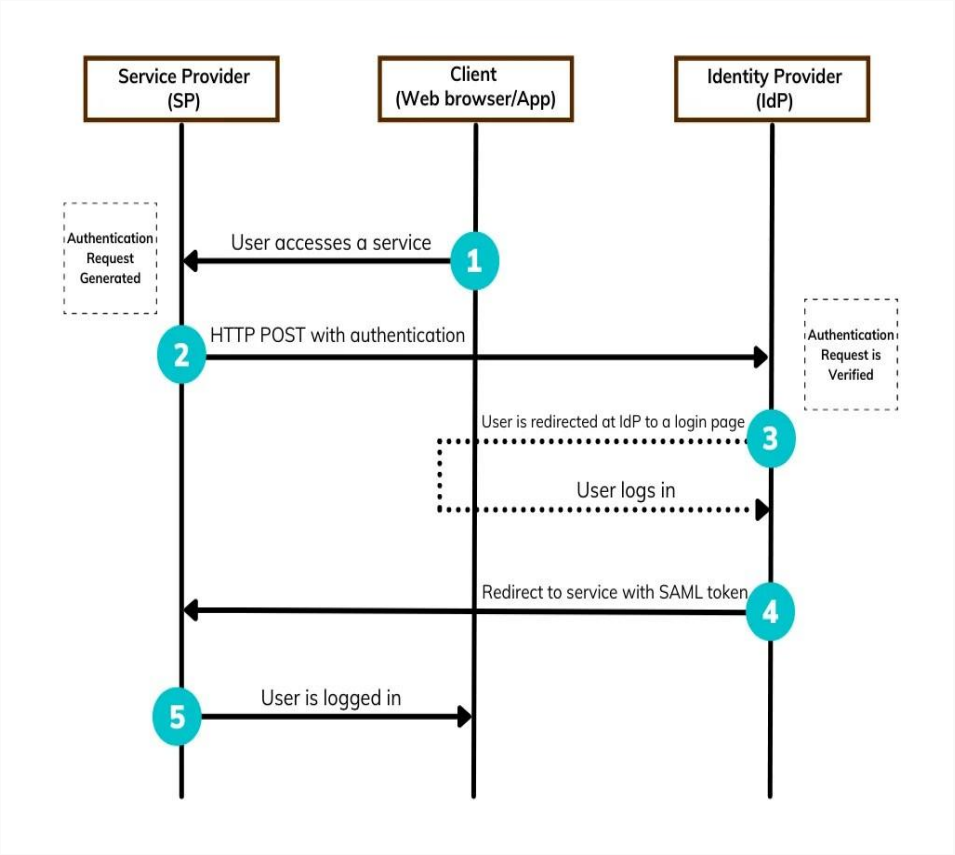
- Security Assertion Markup Language (SAML) 2.0 is an OASIS standard for federated identity
- OAUTH
- OpenID
- eXtensible Access Control Markup Language (XACML)
- System for Cross-domain Identity Management (SCIM)

IAM Standards in Cloud Computing-SAML

- **Security Assertion Markup Language (SAML) 2.0 is an OASIS standard for federated identity management that supports**
  - **both authentication and authorization.**
  - **It uses XML to make assertions between an identity provider and a relying party. Assertions can contain authentication statements, attribute statements, and authorization decision statements.**
  - **SAMLs very widely supported by both enterprise tools and cloud providers but can be complex to initially configure.**

How does SAML work?

- The identity provider and service providers exchange information about users, logins, and characteristics via SAML.
- When a user seeks to access services using Single Sign-On, the identity provider can provide SAML characteristics to the service provider.
- The service provider asks the identity provider for authorization and authentication. The user just has to log in once because both systems use the same language, SAML.
- Each identity provider and service provider need to agree upon the configuration for SAML.
- Both IdP and SP need to have the exact configuration for a successful authentication flow.



IAM Standards in Cloud Computing-OAUTH

- **OAuth is an IETF standard for authorization that is very widely used for web services (including consumer services). OAuth is designed to work over HTTP and is currently on version 2.0.**
  - **It is most often used for delegating access control/authorizations between services.**

How does OAuth work?

An OAuth Access Token transaction requires three players:

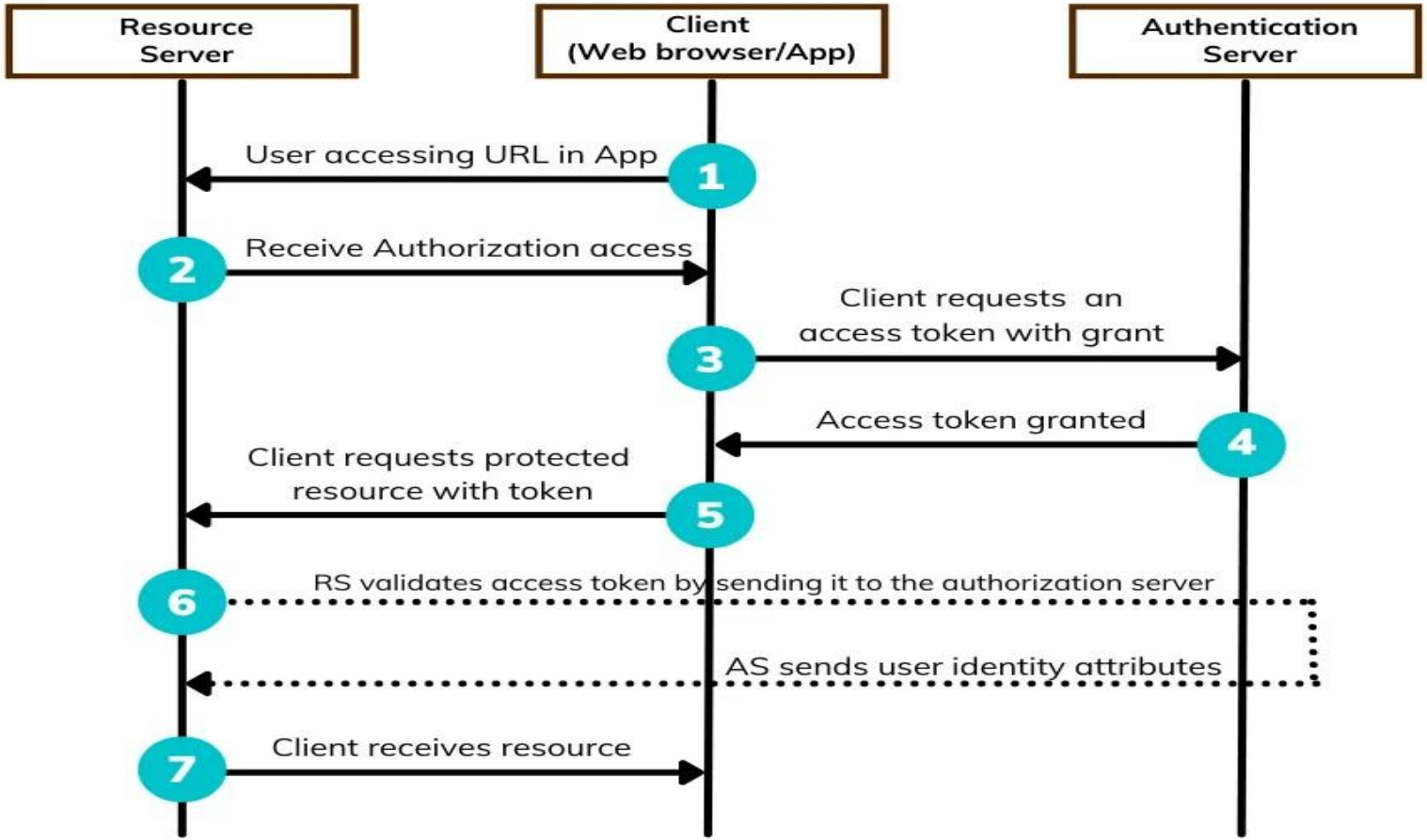
- **the end-user,**
- **the application (API), and the**
- **resource (service provider that has your login credentials).**

The transaction starts when the user specifies that he or she wants to utilize the API.

- **Application asks permission: By submitting the user's confirmed identity as proof, the application or API (application program interface) requests authorization from the resource.**

- **Application requests Access Token: Without having to provide usernames or passwords, the resource grants an Access Token to the API when the authorization has been validated.**

- **Application accesses resource: Tokens come with API access permissions. Permissions are called scopes, and each token has an approved scope for each API. The application can only access the resource to the degree that the scope permits.**



OAuth work Flow

IAM Standards in Cloud Computing-OpenID

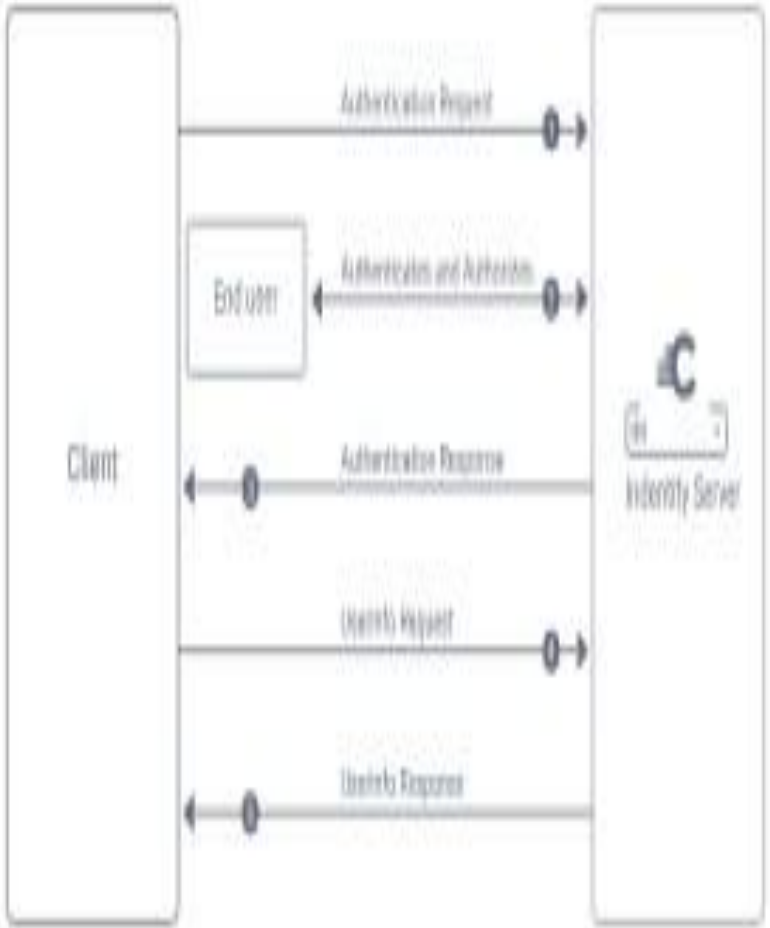
- **OpenID is a standard for federated authentication that is very widely supported for web services. It is based on HTTP with URLs used to identify the identity provider and the user/identity (e.g. identity.identity provider.com). The current version is OpenID Connect 1.0 and it is very commonly seen in consumer services.**

OpenID Connect (OIDC)

OpenID Connect (OIDC) is an open authentication protocol that works on top of the OAuth 2.0 framework. Targeted toward consumers, OIDC allows individuals to use single sign-on (SSO) to access relying party sites using OpenID Providers (OPs), such as an email provider or social network, to authenticate their identities. It provides the application or service with information about the user, the context of their authentication, and access to their profile information.

Asking for authorization and user authentication.

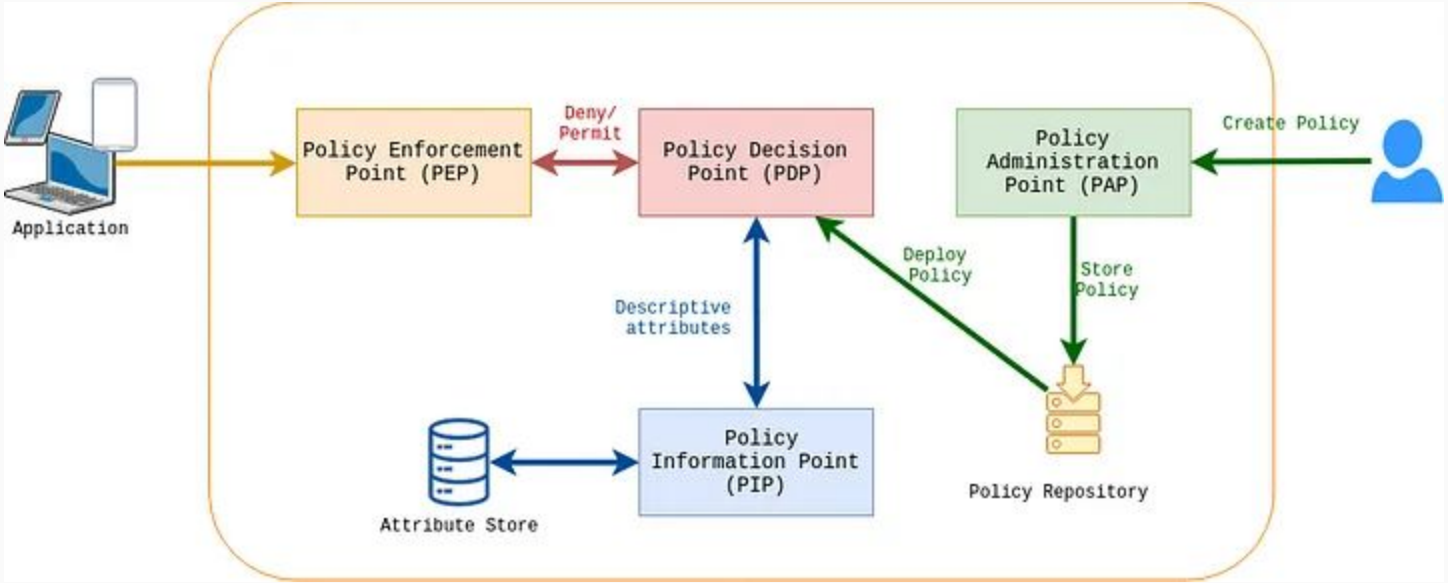
- The identity server then authenticates the end-user.
- The identity server returns a response to the client with the authentication (and authorization) result.
- Optionally, the client can then retrieve user details via the UserInfo endpoint.
- The identity server provides user details to the client.



IAM Standards in Cloud Computing

- eXtensible Access Control Markup Language (XACML) is a standard for defining attribute-based access controls/authorizations. It is a policy language for defining access controls at a Policy Decision Point and then passing them to a Policy Enforcement Point. It can be used with both SAML and OAuth since it solves a different part of the problem—i.e. deciding what an entity is allowed to do with a set of attributes, as opposed to handling logins or delegation of authority.
- System for Cross-domain Identity Management (SCIM) is a standard for exchanging identity information between domains. It can be used for provisioning and deprovisioning accounts in external systems and for exchanging attribute information.
  - One example might be that as a company onboards new employees and separates from existing employees, they are added and removed from the company's electronic employee directory. SCIM could be used to automatically add/delete (or, provision/de-provision) accounts for those users in external systems such as Google Workspace, Office 365, or Salesforce.com. Then, a new user account would exist in the external systems for each new employee, and the user accounts for former employees might no longer exist in those systems.

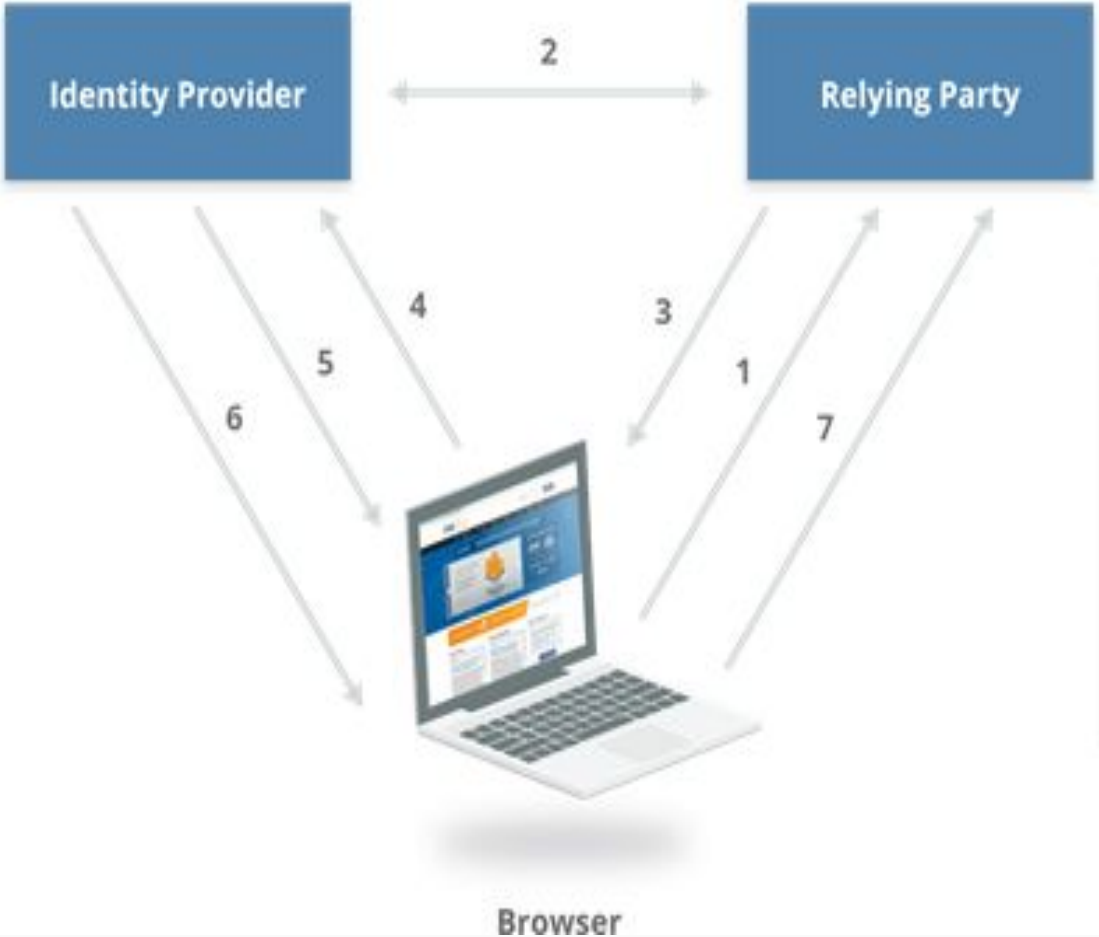
XACML Architecture



How Federated IdM Works?

How Federated Identity Management Works: Federation involves an identity provider making assertions to a relying party after building a trust relationship. At the heart are a series of cryptographic operations to build the trust relationship and exchange credentials.

A practical example is a user logging in to their work network, which hosts a directory server for accounts.That user then opens a browser connection to a SaaS application. Instead of logging in, there are a series of behind-the-scenes operations, where the identity provider (the internal directory server) asserts the identity of the user, and that the user authenticated, as well as any attributes. The relying party trusts those assertions and logs the user in without the user entering any credentials. In fact, the relying party doesn't even have a username or password for that user; it relies on the identity provider to assert successful authentication. To the user they simply go to the website for the SaaS application and are logged in, assuming they successfully authenticated with the internal directory.



1. User sends their OpenID URL
2. IP and RP set shared secret
3. Browser redirected to get token from provider
4. Request to IP for token for site
5. Login if needed
6. Token returned to browser
7. Token handed to requesting site

Managing users and identities for Cloud Computing

- The “identity” part of identity management focuses on the processes and technologies for
  - Registering
  - Provisioning
  - Propagating
  - Managing
  - Deprovisioning Identities

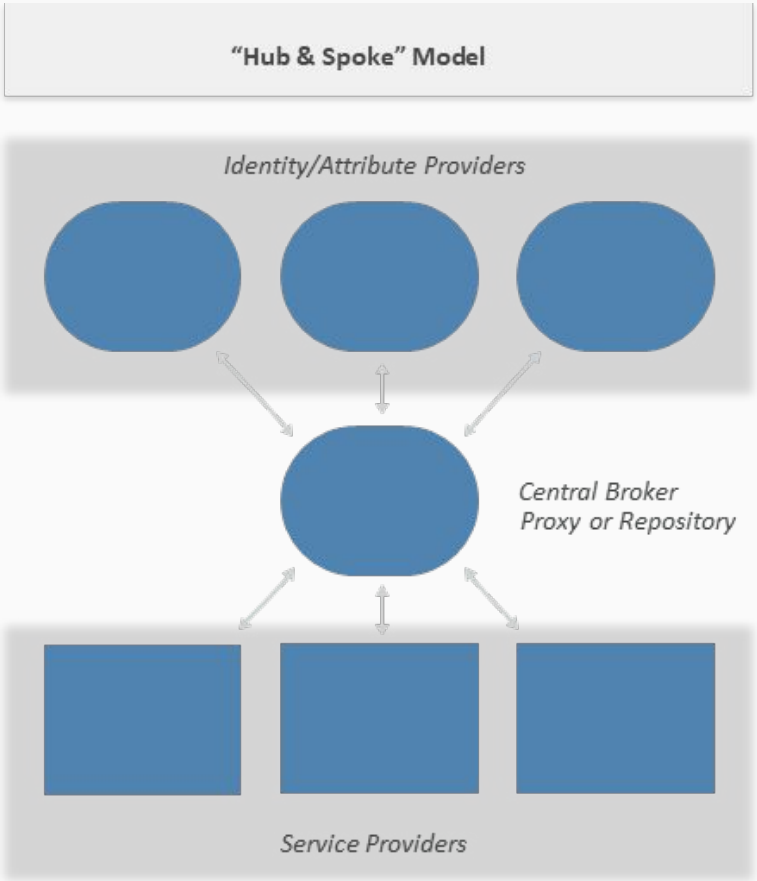
Managing users and identities for Cloud Computing

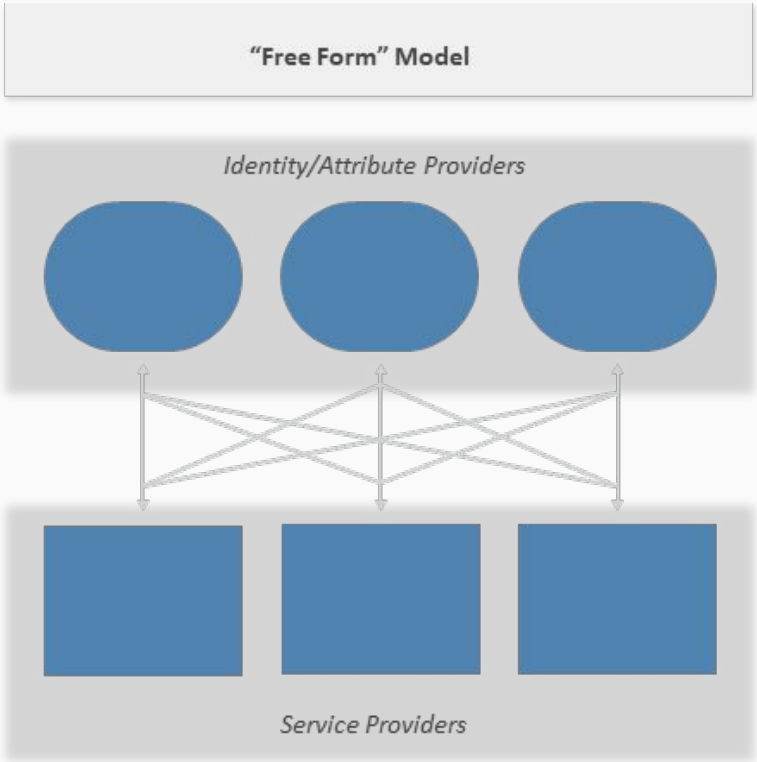
- Cloud providers need to nearly always support
  - internal identities, identifiers, and attributes for users who directly access the service,
  - while also supporting federation so that organizations don't have to manually provision and manage every user in the provider's system and issue everyone separate credentials.
  - Multiple IAM standards plus their own internal user/account management.
  - Enterprise markets provider will need to support federated identity, and most likely SAML .
- Cloud users need to decide where they want to manage their identities and which architectural models and technologies they want to support to integrate with cloud providers.
- As a cloud user, you can log in to a cloud provider and create all your identities in their system.
- When using federation, the cloud user needs to determine the authoritative source (internal directory server) that holds the unique identities they will federate.

Managing users and identities for Cloud Computing

- Cloud based directory support
  - Many cloud providers now support cloud-based directory servers that support federation internally and with other cloud services.
  - For example, more complex architectures can
    - synchronize or federate a portion of an organization's identities for an internal directory through an identity broker and then to a cloud-hosted directory, which then serves as an identity provider for other federated connections.
- Next decide
  - Directly use the authoritative source as the identity provider,
  - Or Integrate with Identity Broker
    - handle federating between identity providers and relying parties (which may not always be a cloud service).
    - They can be located on the network edge or even in the cloud in order toenable web-SSO.
- Architecture
  - Free-Form model
  - Hub and Spoke

Hub & Spoke vs Free-Form





Hub & Spoke vs Free-Form

Free-Form:internal identity providers/sources (often directory servers) connect directly to cloud providers.

The directory needs Internet access. This can be a problem, depending on existing topography, or it may violate security policies.

- It may require users to VPN back to the corporate network before accessing cloud services.
- Depending on the existing directory server, and especially if you have multiple directory servers in different organizational silos, federating to an external provider may be complex and technically difficult.

Hub and Spoke:internal identity providers/sources communicate with a central broker or repository that then serves as the identity provider for federation to cloud providers.

- Identity brokers handle federating between identity providers and relying parties (which may not always be a cloud service). They can be located on the network edge or even in the cloud in order to enable web-SSO.

Architecture and process decisions

How to manage identities for application code, systems, devices, and other services. You may leverage the same model and standards or decide to take a different approach within cloud deployments and applications. For example, the descriptions above skew towards users accessing services, but may not apply equally for services talking to services, systems or devices, or for application components within an IaaS deployment.

- **Defining the identity provisioning process and how to integrate that into cloud deployments. There may also be multiple provisioning processes for different use cases, although the goal should be to have as unified a process as possible.**
- **If the organization has an effective provisioning process in place for traditional infrastructure this should ideally be extended into cloud deployments. However, if existing internal processes are problematic then the organization should instead use the move to cloud as an opportunity to build a new, more effective process.**

Architecture and Process decisions

Provisioning and supporting individual cloud providers and deployments. There should be a formal process for adding new providers into the IAM infrastructure. This includes the process of establishing any needed federation connections, as well as:

- **Mapping attributes (including roles) between the identity provider and the relying party. Enabling required monitoring/logging, including identity-related security monitoring, such as behavioral analytics.**
- **Building an entitlement matrix (discussed more in the next section).**
- **Documenting any break/fix scenarios in case there is a technical failure of any of the federation (or other techniques) used for the relationship.**
- **Ensuring incident response plans for potential account takeovers are in place, including takeovers of privileged accounts.**
- **Implementing deprovisioning or entitlement change processes for identities and the cloud provider. With federation this requires work on both sides of the connection.**

Authentication and Credential

- Multi-factor authentication (MFA) offers one of the strongest options for reducing account takeovers. It isn't a panacea, but relying on a single factor (password) for cloud services is very high risk. When using MFA with federation, the identity provider can and should pass the MFA status as an attribute to the relying party.
- There are multiple options for MFA, including:
  - Hard tokens are physical devices that generate one time passwords for human entry or need to be plugged into a reader. These are the best option when the highest level of security is required.
  - Soft tokens work similarly to hard tokens but are software applications that run on a phone or computer. Soft tokens are also an excellent option but could be compromised if the user's device is compromised, and this risk needs to be considered in any threat model.

Authentication and Credential

- Authentication is the process of proving or confirming an identity. In information security authentication most commonly refers to the act of a user logging in, but it also refers to essentially any time an entity proves who they are and assumes an identity.
- Authentication is the responsibility of the identity provider.The biggest impact of cloud computing on authentication is a greater need for strong authentication using multiple factors. This is for two reasons:
  - Broad network access means cloud services are always accessed over the network, and often over the Internet. Loss of credentials could more easily lead to an account takeover by an attacker, since attacks aren't restricted to the local network.
  - Greater use of federation for Single Sign On means one set of credentials can potentially compromise a greater number of cloud services.

Authentication and Credential

- Out-of-band Passwords are text or other messages sent to a user's phone (usually) and are then entered like any other one time password generated by a token. Although also a good option, any threat model must consider message interception, especially with SMS.
- Biometrics are growing as an option, thanks to biometric readers now commonly available on mobile phones. For cloud services, the biometric is a local protection that doesn't send biometric information to the cloud provider and is instead an attribute that can be sent to the provider. As such the security and ownership of the local device needs to be considered.

Entitlements and Access Management

Entitlements maps identities to authorizations and any required attributes-Entitlements Matrix

Entitlement	SuperAdmin	Service-1 Admin	Service-2 Admin	Dev	SecurityAudit	Security-Admin
Service 1 List	X	X		X	X	X
Service 2 List	X		X	X	X	X
Service 1 Modify Network	X	X		X		X
Service 2 Modify Security Rule	X	X				X
Read Audit Logs	X				X	X

Entitlements process Example

- The cloud provider has an
  - API for launching new virtual machines.
  - That API has a corresponding authorization to allow launching new machines,
    - With additional authorization options for what virtual network a user can launch the VM within.
  - The cloud administrator creates an entitlement that says that users in the developer group can launch virtual machines in only their project network and only if they authenticated with MFA. The group and the use of MFA are attributes of the user's identity. That entitlement is written as a policy that is loaded into the cloud provider's system for enforcement.

Cloud impact on Entitlements,Authorization & Access Mgmt

- Cloud impacts entitlements, authorizations, and access management in multiple ways:
  - Cloud providers and platforms, like any other technology, will have their own set of potential authorizations specific to them. Unless the provider supports XACML (rare today) the cloud user will usually need to configure entitlements within the cloud platform directly.
  - The cloud provider is responsible for enforcing authorizations and access controls.
  - The cloud user is responsible for defining entitlements and properly configuring them within the cloud platform.
  - Cloud platforms tend to have greater support for the Attribute-Based Access Control (ABAC) model for IAM, which offers greater flexibility and security than the Role-Based Access Control (RBAC) model. RBAC is the traditional model for enforcing authorizations and relies on what is often a single attribute (a defined role). ABAC allows more granular and context aware decisions by incorporating multiple attributes, such as role, location, authentication method, and more.
  - ABAC is the preferred model for cloud-based access management.
- When using federation, the cloud user is responsible for mapping attributes, including roles and groups, to the cloud provider and ensuring that these are properly communicated during authentication.
- Cloud providers are responsible for supporting granular attributes and authorizations to enable ABAC and effective security for cloud users.

Privilege Users Management

In terms of controlling risk, few things are more essential than privileged user management.

- **The requirements mentioned above for strong authentication should be a strong consideration for any privileged user.**
- **In addition, account and session recording should be implemented to drive up accountability and visibility for privileged users.**

- **In some cases, it will be beneficial for a privileged user to sign in through a separate tightly controlled system using higher levels of assurances for credential control, digital certificates,physically and logically separate access points, and/or jump hosts.**

AWS -IAM

What is AWS-IAM?

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform

AWS-IAM Features

Shared access to your AWS account

You can grant other people permission to administer and use resources in your AWS account without having to share your password or access key.

Granular permissions

You can grant different permissions to different people for different resources. For example, you might allow some users complete access to Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, and other AWS services. For other users, you can allow read-only access to just some S3 buckets, or permission to administer just some EC2 instances, or to access your billing information but nothing else.

Identity information for assurance

If you use AWS CloudTrail, you receive log records that include information about those who made requests for

AWS-IAM Features

Secure access to AWS resources for applications that run on Amazon EC2

You can use IAM features to securely provide credentials for applications that run on EC2 instances. These credentials provide permissions for your application to access other AWS resources. Examples include S3 buckets and DynamoDB tables.

Multi-factor authentication (MFA)

You can add two-factor authentication to your account and to individual users for extra security. With MFA you or your users must provide not only a password or access key to work with your account, but also a code from a specially configured device. If you already use a FIDO security key with other services, and it has an AWS supported configuration, you can use WebAuthn for MFA security. For more information, see Supported configurations for using FIDO security keys.

Identity federation

AWS-IAM Features

Identity information for assurance

If you use AWS CloudTrail, you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

PCI DSS Compliance

Integrated with many AWS services

For a list of AWS services that work with IAM, see AWS services that work with IAM.

Eventually Consistent

IAM, like many other AWS services, is eventually consistent. IAM achieves high availability by replicating data across multiple servers within Amazon's data centers around the world. I.

Free to use

Accessing IAM

You can work with AWS Identity and Access Management in any of the following ways.

- **AWS Management Console**

  The console is a browser-based interface to manage IAM and AWS resources.

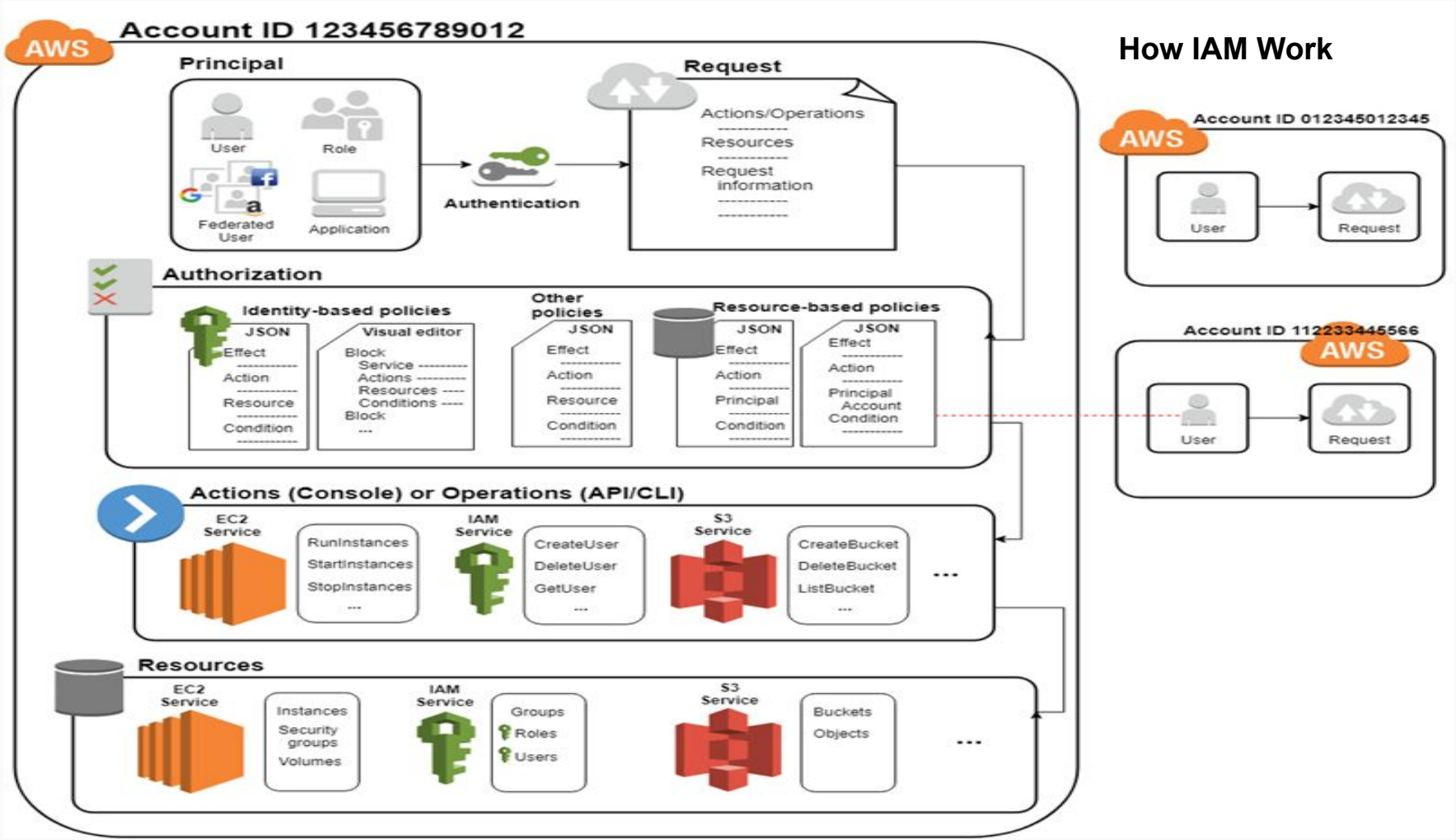
- **AWS Command Line Tools**

  You can use the AWS command line tools to issue commands at your system's command line to perform IAM and AWS tasks.

- **AWS SDKs**

  AWS provides SDKs (software development kits) that consist of libraries and sample code for various

How IAM Work


AWS
Account ID 123456789012
Principal
User
Role
Federated User
Application
Authentication
Request
Actions/Operations
Resources
Request information
Authorization
Identity-based policies
JSON
Effect
Action
Resource
Condition
Visual editor
Block
Service
Actions
Resources
Conditions
Other policies
Resource-based policies
Principal
Account
Actions (Console) or Operations (API/CLI)
EC2 Service
RunInstances
StartInstances
StopInstances
IAM Service
CreateUser
DeleteUser
GetUser
S3 Service
CreateBucket
DeleteBucket
ListBucket
Resources
Instances
Security groups
Volumes
Groups
Roles
Users
Buckets
Objects
Account ID 012345012345
Account ID 112233445566

How IAM Work?

First, a human user or an application uses their sign-in credentials to authenticate with AWS. Authentication is provided by matching the sign-in credentials to a principal (an IAM user, federated user, IAM role, or application) trusted by the AWS account.

Next, a request is made to grant the principal access to resources. Access is granted in response to an authorization request. For example, when you first sign in to the console and are on the console Home page, you are not accessing a specific service. When you select a service, the request for authorization is sent to that service and it looks to see if your identity is on the list of authorized users, what policies are being enforced to control the level of access granted, and any other policies that might be in effect. Authorization requests can be made by principals within your AWS account or from another AWS account that you trust. Once authorized, the principal can take action or perform operations on resources in your AWS account. For example, the principal could launch a new Amazon Elastic Compute Cloud instance, modify IAM group membership, or delete Amazon Simple Storage Service buckets.

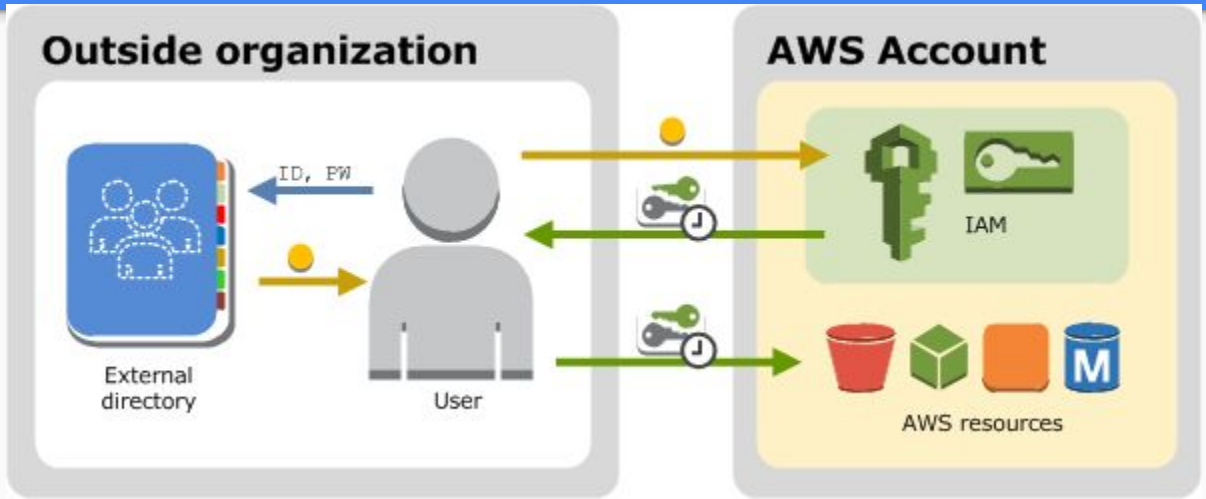
Users Identities in IAM?

You can give access to your AWS account to specific users and provide them specific permissions to access resources in your AWS account. You can use both IAM and AWS IAM Identity Center (successor to AWS Single Sign-On) to create new users or federate existing users into AWS. The main difference between the two is that IAM users are granted long-term credentials to your AWS resources while users in IAM Identity Center have temporary credentials that are established each time the user signs-in t

- **First time access only-Root user Credential**
  - **When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.**
- **IAM users and users in IAM Identity Center**
  - **IAM users are not separate accounts; they are users within your account. Each user can have its own password for access to the AWS Management Console.IAM users are granted long term credential while IAM identity center users are granted temporary or short term credential.**

Users Identities in IAM?

Federating Existing users



Access Management

The access management portion of AWS Identity and Access Management (IAM) helps you define what a principal entity is allowed to do in an account. A principal entity is a person or application that is authenticated using an IAM entity (user or role). Access management is often referred to as *authorization*. You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources

Policy A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal uses an IAM entity (user or role) to make a request. Permissions in the policies determine whether the request is allowed or denied.

Policies and Accounts

If you manage a single account in AWS, then you define the permissions within that account using policies. If you manage permissions across multiple accounts, it is more difficult to manage permissions for your users. You can use IAM roles, resource-based policies, or access control lists (ACLs) for cross-account permissions

Access Management

Policies and users

IAM users are identities in the service. When you create an IAM user, they can't access anything in your account until you give them permission. You give permissions to a user by creating an identity-based policy, which is a policy that is attached to the user or a group to which the user belongs.

Policies and Group

You can organize IAM users into *IAM groups* and attach a policy to a group. In that case, individual users still have their own credentials, but all the users in a group have the permissions that are attached to the group.

Federated users and roles

To assign permissions to federated users, you can create an entity referred to as a *role* and define permissions for the role. When a federated user signs in to AWS, the user is associated with the role and is granted the permissions that are defined in the role.

Access Management

Identity based policies

Identity-based policies are permissions policies that you attach to an IAM identity, such as an IAM user, group, or roles base *Identity-based policies* control what actions the identity can perform, on which resources, and under what conditions. Identity-based policies can be further categorized:

Managed policies – Standalone identity-based policies that you can attach to multiple users, groups, and roles in your AWS account. You can use two types of managed policies:

- **AWS managed policies – Managed policies that are created and managed by AWS. If you are new to using policies, we recommend that you start by using AWS managed policies.**
- **Customer managed policies – Managed policies that you create and manage in your AWS account. Customer managed policies provide more precise control over your policies than AWS managed policies.**

- **Inline policies – Policies that you create and manage and that are embedded directly into a single user, group, or role. In most cases, we don't recommend using inline policies.**

Access Management

Resources based policies

Resource-based policies are permissions policies that you attach to a resource such as an Amazon S3 bucket or an IAM role trust policy.

Resource-based policies control what actions a specified principal can perform on that resource and under what conditions. Resource-based policies are inline policies, and there are no managed resource-based policies. To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy.

The IAM service supports only one type of resource-based policy called a role trust policy, which is attached to an IAM role. Because an IAM role is both an identity and a resource that supports resource-based policies, you must attach both a trust policy and an identity-based policy to an IAM role. Trust policies define which principal entities (accounts, users, roles, and federated users) can assume the role. To learn how IAM roles are different from other resource-based policies,

What is ABAC &RBAC

Attribute based access control

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM resources, including IAM entities (users or roles) and to AWS resources. You can create a single ABAC policy or small set of policies for your IAM principals. These ABAC policies can be designed to allow operations when the principal's tag matches the resource tag. ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

Role based Access Control

The traditional authorization model used in IAM is called role-based access control (RBAC). RBAC defines permissions based on a person's job function, known outside of AWS as a *role*. Within AWS a role usually refers to an IAM role, which is an identity in IAM that you can assume. IAM does include managed policies for job functions that align permissions to a job function in an RBAC model.

Advantages of ABAC vs RBAC

- ABAC permissions scale with innovation. It's no longer necessary for an administrator to update existing policies to allow access to new resources. F
- ABAC requires fewer policies. Because you don't have to create different policies for different job functions, you create fewer policies. Those policies are easier to manage.
- Using ABAC, teams can change and grow quickly. This is because permissions for new resources are automatically granted based on attributes.
- Granular permissions are possible using ABAC. When you create policies, it's a best practice to grant least privilege.
- Use employee attributes from your corporate directory with ABAC. You can configure your SAML-based or web identity provider to pass session tags to AWS. When your employees federate into AWS, their attributes are applied to their resulting principal in AWS. You can then use ABAC to allow or deny permissions based on those attributes.

Amazon Elastic Compute Cloud (Amazon EC2)	Yes	Partial	No	Yes	Yes	Partial[1]
Amazon EC2 Auto Scaling	Yes	Yes	No	Yes	Yes	Yes
EC2 Image Builder	Yes	Yes	No	Yes	Yes	Yes
Amazon EC2 Instance Connect	Yes	Yes	No	No	Yes	No
AWS Elastic Beanstalk	Yes	Partial	No	Yes	Yes	Yes
Amazon Elastic Inference	Yes	Yes	No	No	Yes	No
AWS Elastic Load Balancing	Yes	Partial	No	Partial	Yes	Yes
AWS Lambda	Yes	Yes	Yes	Partial[2]	Yes	Partial[3]
Amazon Lightsail	Yes	Partial[4]	No	Partial[4]	Yes	Yes