AWS - Service Control Policies (SCPs)

Users and roles must still be granted permissions using IAM permission policies attached to them or to groups.

The SCPs filter the permissions granted by such policies, and the user can't perform any actions that the applicable SCPs don't allow.

Actions allowed by the SCPs can be used if they are granted to the user or role by one or more IAM permission policies.

When you attach SCPs to the root, OUs, or directly to accounts, all policies that affect a given account are evaluated together using the same rules that govern IAM permission policies:

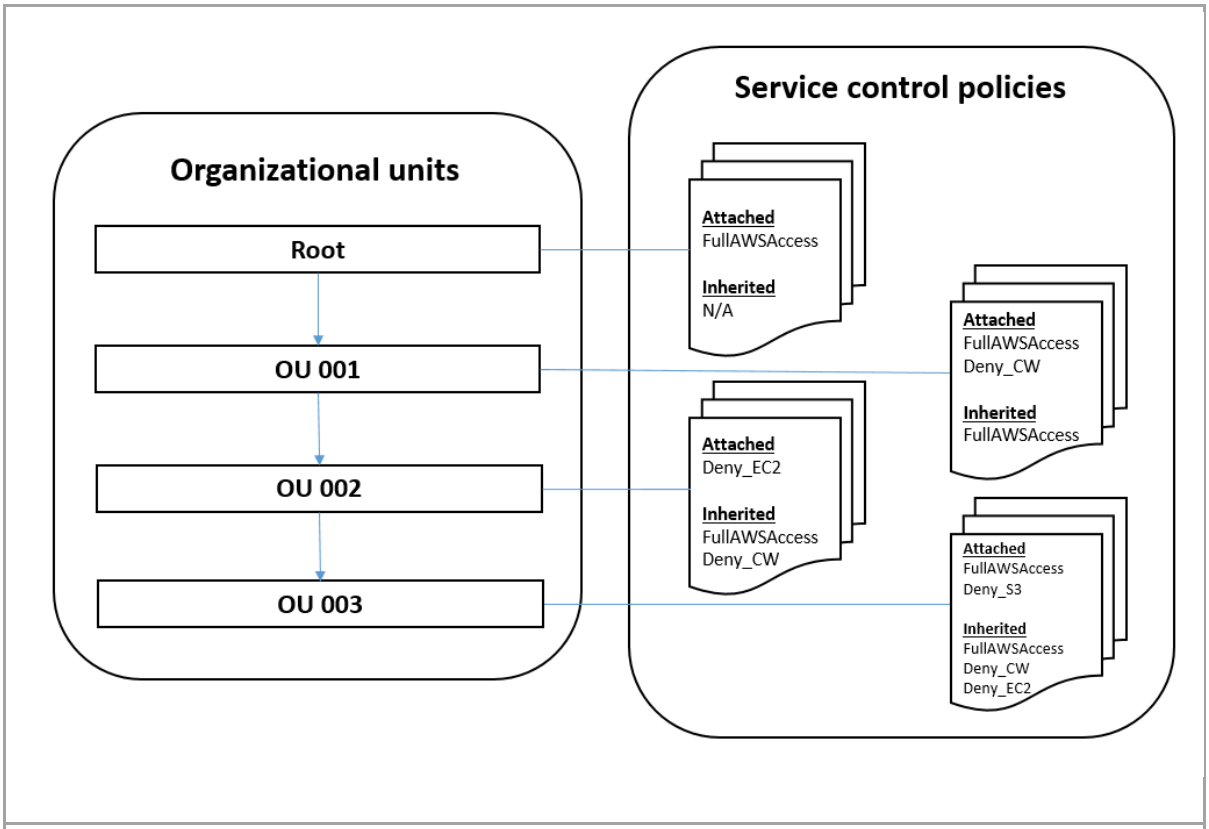
- Any action that has an explicit Deny in an SCP can't be delegated to users or roles in the affected accounts. An explicit Deny statement overrides any Allow that other SCPs might grant.

- Any action that has an explicit Allow in an SCP (such as the default "*" SCP or by any other SCP that calls out a specific service or action) can be delegated to users and roles in the affected accounts.

- Any action that isn't explicitly allowed by an SCP is implicitly denied and can't be delegated to users or roles in the affected accounts.

By default, an SCP named FullAWSAccess is attached to every root, OU, and account. This default SCP allows all actions and all services. So in a new organization, until you start creating or manipulating the SCPs, all of your existing IAM permissions continue to operate as they did. As soon as you apply a new or modified SCP to a root or OU that contains an account, the permissions that your users have in that account become filtered by the SCP. Permissions that used to work might now be denied if they're not allowed by the SCP at every level of the hierarchy down to the specified account.

As stated in the documentation of AWS Organizations, ***SCPs DO NOT affect any service-linked role. Service-linked roles enable other AWS services to integrate with AWS Organizations and can't be restricted by SCPs.***



A service control policy (SCP) is a policy that specifies the services and actions that users and roles can use in the specified AWS accounts.

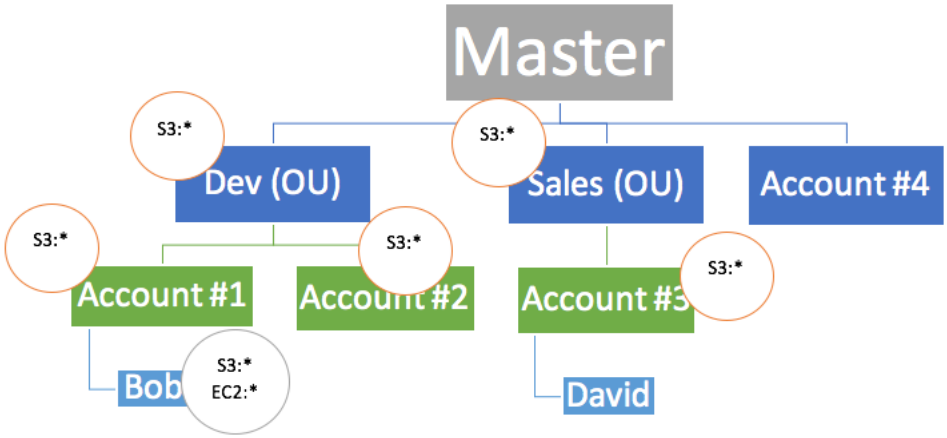
SCPs are similar to IAM permission policies except that they don't grant any permissions.

Instead, SCPs specify the maximum permissions for an organization, organizational unit (OU), or account.

When you attach an SCP to your organization root or an OU, the SCP limits permissions for entities in member accounts.

Even if a user is granted full administrator permissions with an IAM permission policy, any access that is not explicitly allowed or that is explicitly denied by the SCPs affecting that account is blocked.

For example, if you assign an SCP that allows only database service access to your "database" account, then any user, group, or role in that account is denied access to any other service's operations. SCPs are available only when you enable all features in your organization.



References:

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scp.html

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_about-scps.html

Service Control Policies (SCP) vs IAM Policies:

https://tutorialsdojo.com/aws-cheat-sheet-service-control-policies-scp-vs-iam-policies/

Comparison of AWS Services Cheat Sheets:

https://tutorialsdojo.com/comparison-of-aws-services-for-udemy-students/

AWS - IAM Service Linked Accounts

IAM Service Linked Accounts

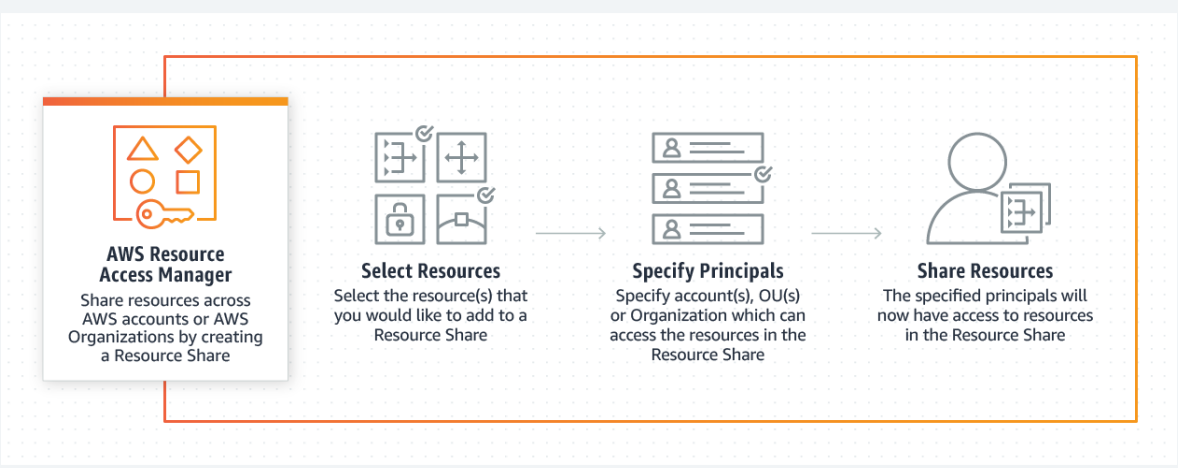
https://docs.aws.amazon.com/IAM/latest/UserGuide/using-service-linked-roles.html

- 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.
- The linked service also defines how you create, modify, and delete a service-linked role. A service might automatically create or delete the role.
- It might allow you to create, modify, or delete the role as part of a wizard or process in the service. Or it might require that you use IAM to create or delete the role.
- Regardless of the method, service-linked roles make setting up a service easier because you don't have to manually add the necessary permissions for the service to complete actions on your behalf.

Integrate AWS RAM and AWS Organizations

AWS - IAM Service Linked Accounts (TRUSTED roles)

You can use ***trusted access*** to enable an AWS service that you specify, called the *trusted service*, to perform tasks in your organization and its accounts on your behalf.
This involves granting permissions to the trusted service but does not otherwise affect the permissions for IAM users or roles.
When you enable access, the trusted service can create an IAM role called a service-linked role in every account in your organization. That role has a permissions policy that allows the trusted service to do the tasks that are described in that service's documentation. This enables you to specify settings and configuration details that you would like the trusted service to maintain in your organization's accounts on your behalf.



AWS Resource Access Manager (AWS RAM) enables you to share specified AWS resources that you own with other AWS accounts. To enable trusted access with AWS Organizations:

From the AWS RAM CLI, use the enable-sharing-with-aws-organizations command.

Name of the IAM service-linked role that can be created in accounts when trusted access is enabled: *AWSResourceAccessManagerServiceRolePolicy*.

References:

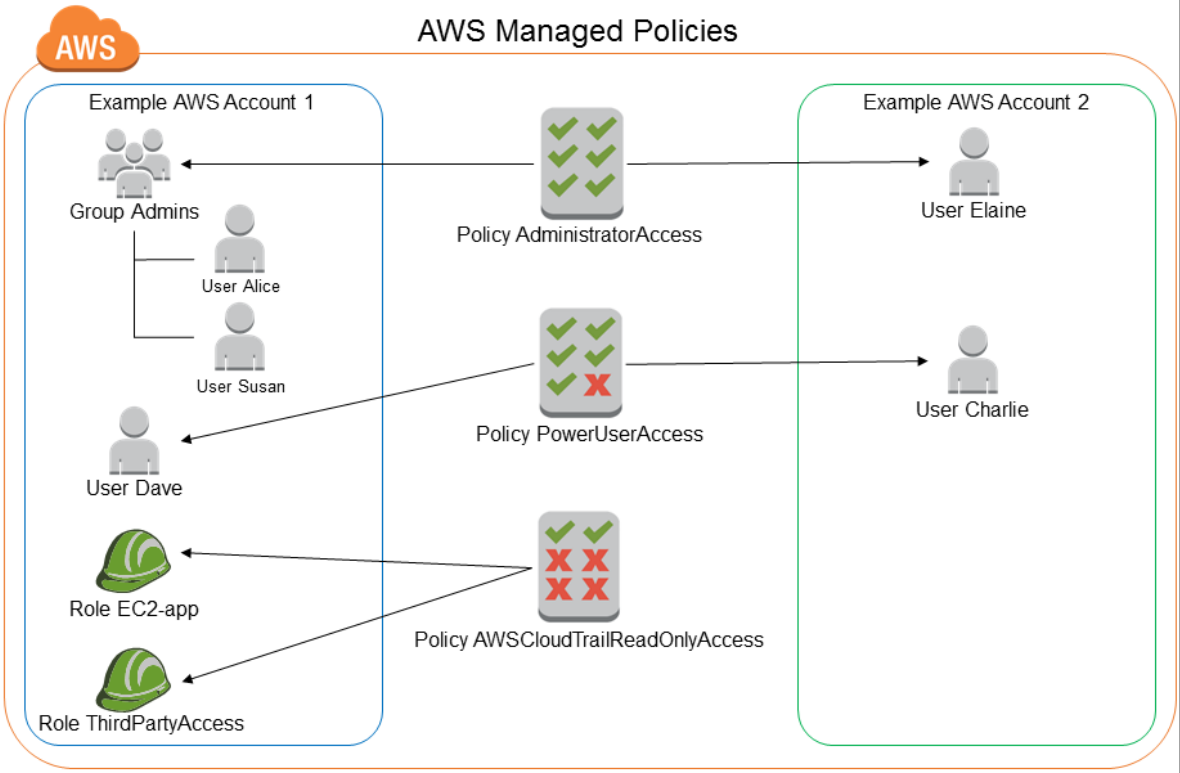
https://docs.aws.amazon.com/organizations/latest/userguide/orgs_integrate_services.html

https://docs.aws.amazon.com/organizations/latest/userguide/services-that-can-integrate-ram.html

https://aws.amazon.com/blogs/security/introducing-an-easier-way-to-delegate-permissions-to-aws-services-service-linked-roles/

AWS - Managed Policies

AWS managed policies for job functions are designed to closely align to common job functions in the IT industry. You can use these policies to easily grant the permissions needed to carry out the tasks expected of someone in a specific job function. These policies consolidate permissions for many services into a single policy that's easier to work with than having permissions scattered across many policies.



There are a lot of available AWS Managed Policies that you can directly attach to your IAM Users, such as Administrator, Billing, Database Administrator, Data Scientist, Developer Power User, Network Administrator, Security Auditor, System Administrator and many others.

AdministratorAccess

- For Administrators, you can use the AWS managed policy name: **AdministratorAccess** if you want to provision full access to a specific IAM User.
- This will enable the user to delegate permissions to every service and resource in AWS as this policy grants all actions for all AWS services and for all resources in the account.

PowerUserAccess

- For Developer Power Users, you can use the AWS managed policy name: **PowerUserAccess** if you have users who perform application development tasks.
- This policy will enable them to create and configure resources and services that support AWS aware application development.
- **The first statement** of this policy uses the *NotAction* element to allow all actions for all AWS services and for all resources except AWS Identity and Access Management and AWS Organizations.

- **The second statement grants** IAM permissions to create a service-linked role. This is required by some services that must access resources in another service, such as an Amazon S3 bucket. It also grants Organizations permissions to view information about the user's organization, including the master account email and organization limitations.

Web Identity Federation

- With web identity federation, you don't need to create custom sign-in code or manage your own user identities.
- Instead, users of your app can sign in using a well-known identity provider (IdP) —such as
  - Login with Amazon, Facebook, Google, or
  - any other OpenID Connect (OIDC)-compatible IdP,
- receive an authentication token, and then exchange that token for temporary security credentials in AWS that map to an IAM role with permissions to use the resources in your AWS account.
- Using an IdP helps you keep your AWS account secure because you don't have to embed and distribute long-term security credentials with your application.

Reference:

http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_oidc.html

Check out this AWS IAM Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-aws-identity-and-access-management-iam/

WebPortal - On-premise LDAP

You are a Software Engineer for a leading call center company in Seattle. Their corporate web portal is deployed to AWS and is linked to their corporate data center via a link aggregation group (LAG) which terminates at the same AWS Direct Connect endpoint and connected on a private virtual interface (VIF) in your VPC. The portal must authenticate against their on-premises LDAP server. Each Amazon S3 bucket can only be accessed by a logged-in user if it belongs to that user.

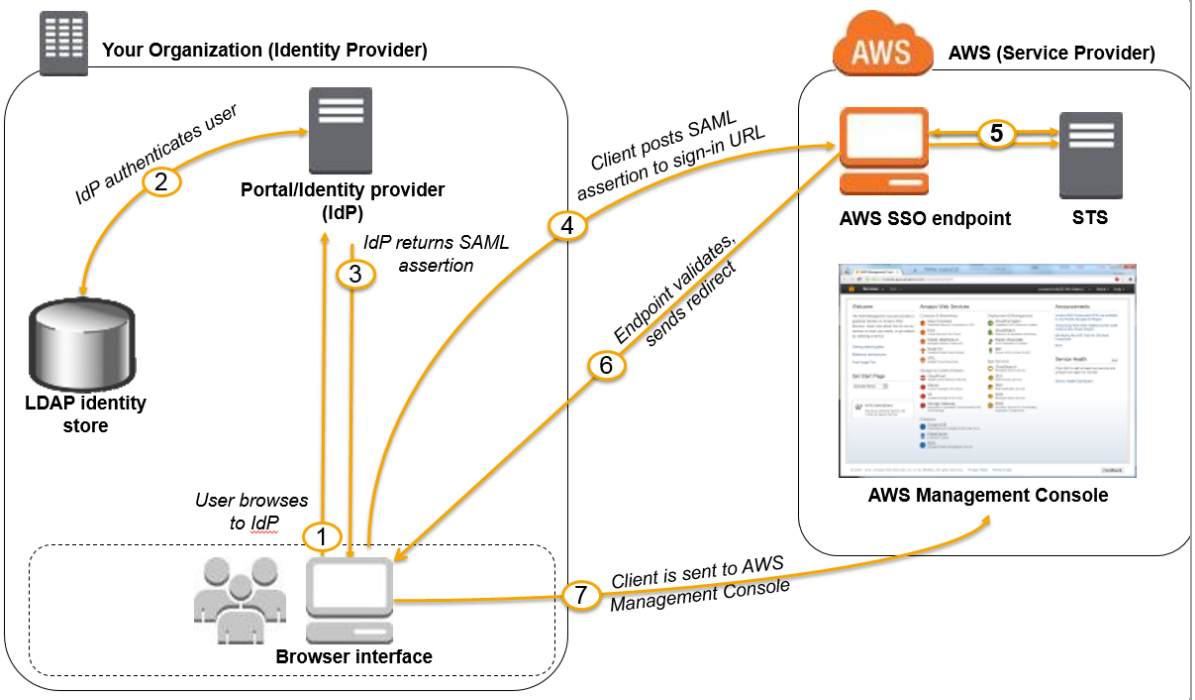
How will you implement this architecture in AWS? (Select TWO.)

- Create an identity broker that assumes an IAM role, and retrieve temporary AWS security credentials via IAM Security Token Service (STS). The application gets the AWS temporary security credentials from the identity broker to gain access to the appropriate S3 bucket.

-
- The application first authenticates against LDAP to retrieve the name of an IAM role associated with the user. It then assumes that role via call to IAM Security Token Service (STS). Afterwards, the application can now use the temporary credentials from the role to access the appropriate S3 bucket.

SAML2.0 - Identity Federation

Since the company is using Microsoft Active Directory which implements Security Assertion Markup Language (SAML), you can set up a SAML-Based Federation for API Access to your AWS cloud.
In this way, you can easily connect to AWS using the login credentials of your on-premises network.



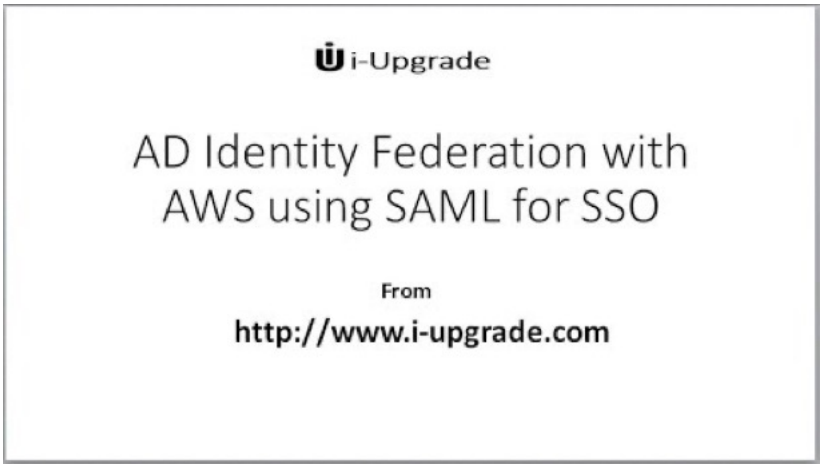
- AWS supports identity federation with SAML 2.0, an open standard that many identity providers (IdPs) use.
- This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS APIs without you having to create an IAM user for everyone in your organization.
- By using SAML, you can simplify the process of configuring federation with AWS, because you can use the IdP's service instead of writing custom identity proxy code.

- Before you can use SAML 2.0-based federation as described in the preceding scenario and diagram, you must configure your organization's IdP and your AWS account to trust each other.
- The general process for configuring this trust is described in the following steps.

- Inside your organization, you must have an IdP that supports SAML 2.0, like Microsoft Active Directory Federation Service (AD FS, part of Windows Server), Shibboleth, or another compatible SAML 2.0 provider.

https://aws.amazon.com/blogs/security/enabling-federation-to-aws-using-windows-active-directory-adfs-and-saml-2-0/

How to Integrate AD with AWS Using SAML - SSO



References:

http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_saml.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers.html

Check out this AWS IAM Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-aws-identity-and-access-management-iam/