**Domain: 10. Security, Identity, & Compliance**

**Topic: IAM – Identity and Access Management**

## Service Overview

*AWS Identity and Access Management (IAM) helps you securely control access to Amazon Web Services (AWS) and your account resources. IAM can also keep your account credentials private. With IAM, you can create multiple IAM users under the umbrella of your AWS account or enable temporary access through identity federation with your corporate directory. In some cases, you can also enable access to resources across AWS accounts.*

*Without IAM, however, you must either create multiple AWS accounts—each with its own billing and subscriptions to AWS products—or your employees must share the security credentials of a single AWS account. In addition, without IAM, you cannot control the tasks a particular user or system can do and what AWS resources they might use.*

Almost everything is tied with IAM in AWS!

## Use cases / Considerations

* **Fine-grained access control to AWS resources** - IAM enables [your users](https://aws.amazon.com/iam/features/manage-users/) to [control access](https://aws.amazon.com/iam/features/manage-permissions/) to AWS service APIs and to specific resources. IAM also enables you to add [specific conditions](https://aws.amazon.com/iam/features/manage-permissions/) such as time of day to control how a user can use AWS, their originating IP address, whether they are using SSL, or whether they have authenticated with a [multi-factor authentication device](https://aws.amazon.com/iam/features/mfa/). IAM is secure by default; users have no access to AWS resources until permissions are explicitly granted.
* **Temporary credentials -** In addition to defining access permissions directly to users and groups, IAM lets you create roles. Roles allow you to define a set of permissions and then let authenticated users or EC2 instances assume them, increasing your security posture by granting temporary access to the resources you define.
* **Multi-factor authentication for highly privileged users -** protect your AWS environment by using [AWS MFA](https://aws.amazon.com/iam/features/mfa/), a security feature available at no extra cost that augments user name and password credentials. MFA requires users to prove physical possession of a hardware MFA token or MFA-enabled mobile device by providing a valid MFA code.
* **Analyze access** - IAM helps you [analyze access](https://aws.amazon.com/iam/features/analyze-access/) across your AWS environment. Your security teams and administrators can quickly validate that your policies only provide the intended public and cross-account access to your resources. You can also easily identify and refine your policies to allow access to only the services being used. This helps you to better adhere to the principle of least privilege.
* **Integrate with your corporate directory** - IAM can be used to grant your employees and applications [federated access](https://aws.amazon.com/identity/federation/) to the AWS Management Console and AWS service APIs, using your existing identity systems such as Microsoft Active Directory. You can use any [identity management](http://docs.aws.amazon.com/IAM/latest/UserGuide/IdP-solution-providers.html) solution that supports [SAML 2.0](https://aws.amazon.com/identity/saml/), or feel free to use one of our federation samples ([AWS Console SSO](http://aws.amazon.com/code/4001165270590826) or [API federation](http://aws.amazon.com/code/1288653099190193)).
* **Seamlessly integrated into AWS services** - IAM is integrated into most AWS services. This provides the ability to define access controls from one place in the AWS Management Console that will take effect throughout your AWS environment.

## Cautions

*Roles and policies are global AWS resources, their names must be unique across your AWS account.*

## Details:

## <https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_iam-quotas.html>

## Pricing considerations

IAM is a feature of your AWS account offered at no additional charge. You will be charged only for the use of other AWS services by your users.

## Explanation of the Solution\Useful links

*IAM Basics* [*https://tutorialsdojo.com/aws-identity-and-access-management-iam/*](https://tutorialsdojo.com/aws-identity-and-access-management-iam/)

[*IAM principles and features*](https://aws.amazon.com/iam/features/)

*Using roles:* [*https://docs.aws.amazon.com/IAM/latest/UserGuide/id\_roles\_use.html*](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use.html)

*Policy evaluation logic -* [*https://docs.aws.amazon.com/IAM/latest/UserGuide/reference\_policies\_evaluation-logic.html*](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html)

[*AWS IAM FAQs*](https://aws.amazon.com/iam/faqs/)

*[https://docs.aws.amazon.com/general/latest/gr/root-vs-iam.html#aws\_tasks-that-require-root](https://docs.aws.amazon.com/general/latest/gr/root-vs-iam.html" \l "aws_tasks-that-require-root)*

## Tutorials and guides

* [Creating an IAM admin user and group](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-admin-group.html)
* [Creating a delegated user](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-delegated-user.html)
* [How IAM users sign in](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_how-users-sign-in.html)
* [Tutorial: Delegate Access to the Billing Console](http://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_billing.html)
* [Tutorial: Delegate Access Across AWS Accounts Using IAM Roles](http://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_cross-account-with-roles.html)
* [Tutorial: Create and Attach Your First Customer Managed Policy](http://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_managed-policies.html)
* [Tutorial: Enable Your Users to Configure Their Own Credentials and MFA Settings](http://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_users-self-manage-mfa-and-creds.html)
* [How do I assume an IAM role using the AWS CLI?](https://aws.amazon.com/premiumsupport/knowledge-center/iam-assume-role-cli/)

Video:

* Become an IAM policy master in 60 minutes or less <https://youtu.be/YQsK4MtsELU>
* IAM ninja - <https://www.youtube.com/watch?v=aISWoPf_XNE>

**Task: Creating and using of IAM roles/policies.**

**Problem to Be Solved**

*Study how to create IAM user/roles/policies and use them for different AWS services, e.g. assume a role to use it in the AWS CLI.*

## Implementation Details

*Pre-requisites:*

* *Already created S3 bucket*
* *Installed jq -* [*https://stedolan.github.io/jq/download/*](https://stedolan.github.io/jq/download/)
* *Installed AWS cli v1 -* [*https://docs.aws.amazon.com/cli/latest/userguide/install-cliv1.html*](https://docs.aws.amazon.com/cli/latest/userguide/install-cliv1.html)
* [*https://awspolicygen.s3.amazonaws.com/policygen.html*](https://awspolicygen.s3.amazonaws.com/policygen.html) *or* [*https://aws.amazon.com/blogs/security/use-the-new-visual-editor-to-create-and-modify-your-aws-iam-policies/*](https://aws.amazon.com/blogs/security/use-the-new-visual-editor-to-create-and-modify-your-aws-iam-policies/)

*Create a new user.*

*Attach to the new user AWS managed policy: S3 full access.*

*Attach to the new user an inline policy, that should contain the follow “Actions”:*

* *ssm:ReadOnly*
* *kms:Decrypt*
* *iam:ListRoles*
* *sts:AssumeRole*

*It’s highly recommended to read about the last two “Actions” in the Tutorials and guides.*

*Attach to the new user the Customer managed policy named “KmsDecryptPolicy”, that policy should contain “Action” - kms:Decrypt*

*Hint: Use policy generator*

*After attaching all of that policies to the new user, add him permission for programmatic api access*

*Add Role named “S3ReadOnlyRole”, attach to the role AWS managed policy: S3 read-only*

*Create a json and assume role policy that defines the trust relationship of the IAM role:*

*For example,*

{ "Version": "2012-10-17", "Statement": { "Effect": "Allow", "Principal": { "AWS": "arn:aws:iam::123456789012:user/new\_created\_user" }, "Action": "sts:AssumeRole" } }

*Verification:*

*1. credentials retrieval from AWS CLI for the newly created user*

*2. authorize in AWS CLI using credentials from the previous step*

*3. write file to an existing s3 bucket to test full access using AWS CLI*

*4. perform role assuming using command* aws sts assume-role

*5. authorize in AWS CLI using credentials from the previous step*

*6. make sure that you are authorized by using role*

*7. make a try to write file from step 2*

*8. copy file from s3, that was uploaded in step, using AWS CLI*

*Solution levels (could be done in a sequence):*

*Basic - create IAM roles and user using AWS Console and policy generator, proceed with verification from cli*

*Medium - create everything using AWS cli (hint: use policies in json forma from Basic level) and proceed with verification*

*Pro - create everything using IaC approach with CloudFormation/Terraform, proceed with verification from cli*

## Benefits / Outcomes / Pros and Cons / Summary

1. *Almost everything in AWS is tied with IAM.*
2. *Usage of AWS managed and custom managed policies.*
3. *Understanding of IAM roles precedence.*
4. *Understanding of service-managed roles.*
5. *Assuming roles practice and key points.*
6. *Instance-profile*
7. *Resource-based policies*

## Tearing down

*Delete all resources and delete IAM roles/policies, since you cannot delete policy or role that is in use.*