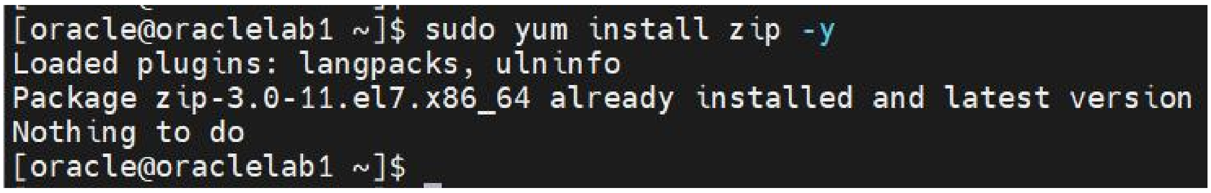
Step 1: Install Required Tools

You need **zip** for compression and AWS CLI for uploading. Run these commands as **root** or with **sudo**:

# Install zip (usually installed, but good to check)

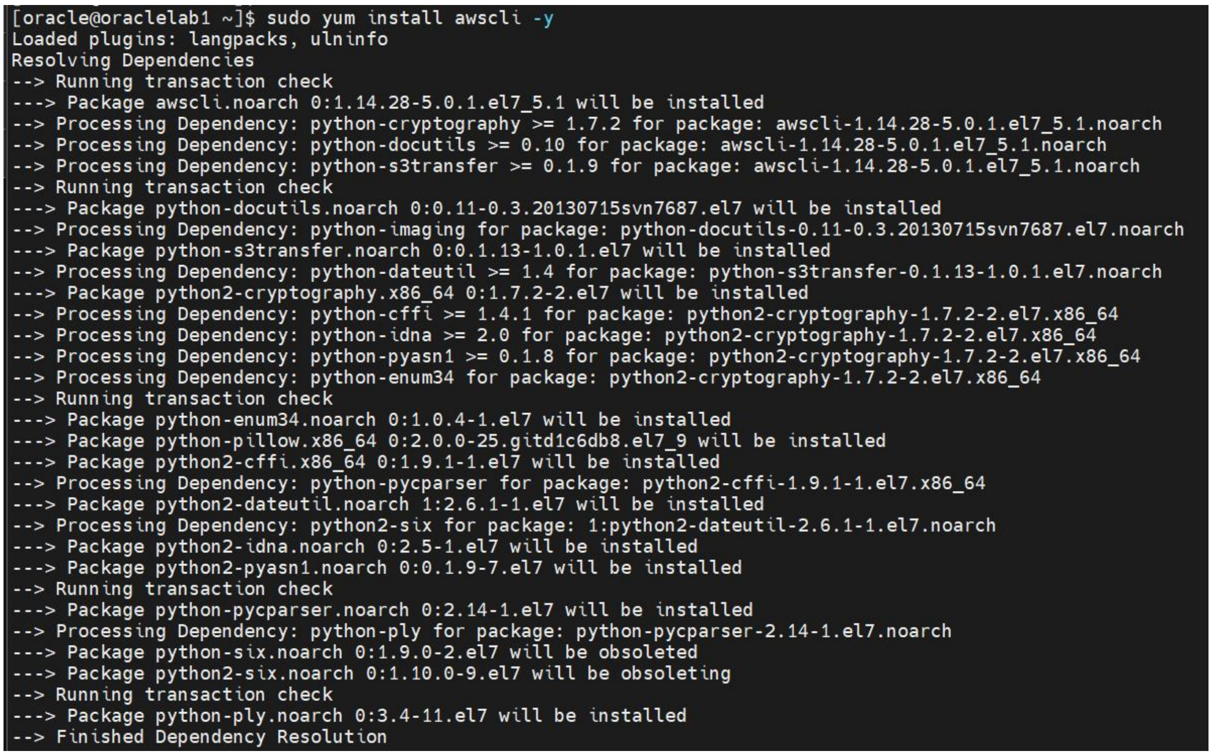
sudo yum install zip -y



```
[oracle@oraclelab1 ~]$ sudo yum install zip -y
Loaded plugins: langpacks, ulninfo
Package zip-3.0-11.el7.x86_64 already installed and latest version
Nothing to do
[oracle@oraclelab1 ~]$ _
```

# Install AWS CLI

sudo yum install awscli -y



```
[oracle@oraclelab1 ~]$ sudo yum install awscli -y
Loaded plugins: langpacks, ulninfo
Resolving Dependencies
--> Running transaction check
---> Package awscli.noarch 0:1.14.28-5.0.1.el7_5.1 will be installed
--> Processing Dependency: python-cryptography >= 1.7.2 for package: awscli-1.14.28-5.0.1.el7_5.1.noarch
--> Processing Dependency: python-docutils >= 0.10 for package: awscli-1.14.28-5.0.1.el7_5.1.noarch
--> Processing Dependency: python-s3transfer >= 0.1.9 for package: awscli-1.14.28-5.0.1.el7_5.1.noarch
--> Running transaction check
---> Package python-docutils.noarch 0:0.11-0.3.20130715svn7687.el7 will be installed
--> Processing Dependency: python-imaging for package: python-docutils-0.11-0.3.20130715svn7687.el7.noarch
---> Package python-s3transfer.noarch 0:0.1.13-1.0.1.el7 will be installed
--> Processing Dependency: python-dateutil >= 1.4 for package: python-s3transfer-0.1.13-1.0.1.el7.noarch
---> Package python2-cryptography.x86_64 0:1.7.2-2.el7 will be installed
--> Processing Dependency: python-cffi >= 1.4.1 for package: python2-cryptography-1.7.2-2.el7.x86_64
--> Processing Dependency: python-idna >= 2.0 for package: python2-cryptography-1.7.2-2.el7.x86_64
--> Processing Dependency: python-pyasn1 >= 0.1.8 for package: python2-cryptography-1.7.2-2.el7.x86_64
--> Processing Dependency: python-enum34 for package: python2-cryptography-1.7.2-2.el7.x86_64
--> Running transaction check
---> Package python-enum34.noarch 0:1.0.4-1.el7 will be installed
---> Package python-pillow.x86_64 0:2.0.0-25.gitd1c6db8.el7_9 will be installed
---> Package python2-cffi.x86_64 0:1.9.1-1.el7 will be installed
--> Processing Dependency: python-pycparser for package: python2-cffi-1.9.1-1.el7.x86_64
---> Package python2-dateutil.noarch 1:2.6.1-1.el7 will be installed
--> Processing Dependency: python2-six for package: 1:python2-dateutil-2.6.1-1.el7.noarch
---> Package python2-idna.noarch 0:2.5-1.el7 will be installed
---> Package python2-pyasn1.noarch 0:0.1.9-7.el7 will be installed
--> Running transaction check
---> Package python-pycparser.noarch 0:2.14-1.el7 will be installed
--> Processing Dependency: python-ply for package: python-pycparser-2.14-1.el7.noarch
---> Package python-six.noarch 0:1.9.0-2.el7 will be obsoleted
---> Package python2-six.noarch 0:1.10.0-9.el7 will be obsoleting
--> Running transaction check
---> Package python-ply.noarch 0:3.4-11.el7 will be installed
--> Finished Dependency Resolution
```

Step 2: Configure AWS CLI

Before uploading, you must configure your AWS credentials.

aws configure

It will ask for:

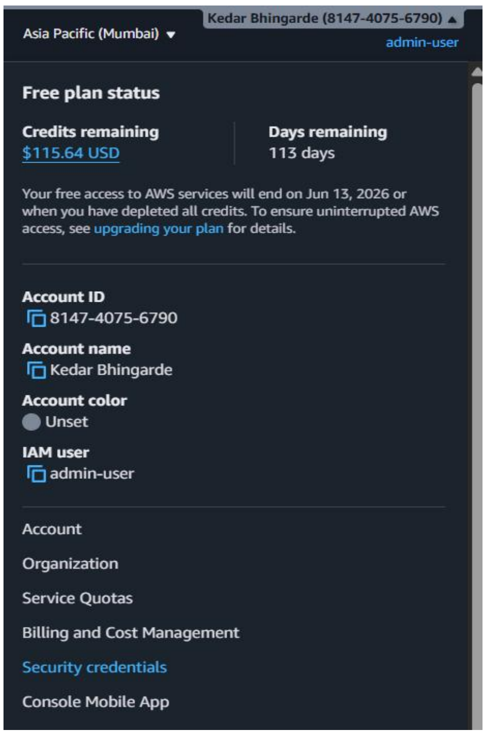
1. **AWS Access Key ID: AKIA33MSVLE3B7HOMVZQ**
2. **AWS Secret Access Key: 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**
3. **Default region name: ap-south-1**

4. **Default output format: [Press Enter for default]**

To find above details follow below steps:

Step 1: Access Security Credentials

Click on **Security credentials** in the menu shown in your image.


Asia Pacific (Mumbai) ▾

Kedar Bhingarde (8147-4075-6790) ▴

admin-user

Free plan status

Credits remaining
$115.64 USD

Days remaining
113 days

Your free access to AWS services will end on Jun 13, 2026 or when you have depleted all credits. To ensure uninterrupted AWS access, see upgrading your plan for details.

Account ID
8147-4075-6790

Account name
Kedar Bhingarde

Account color
Unset

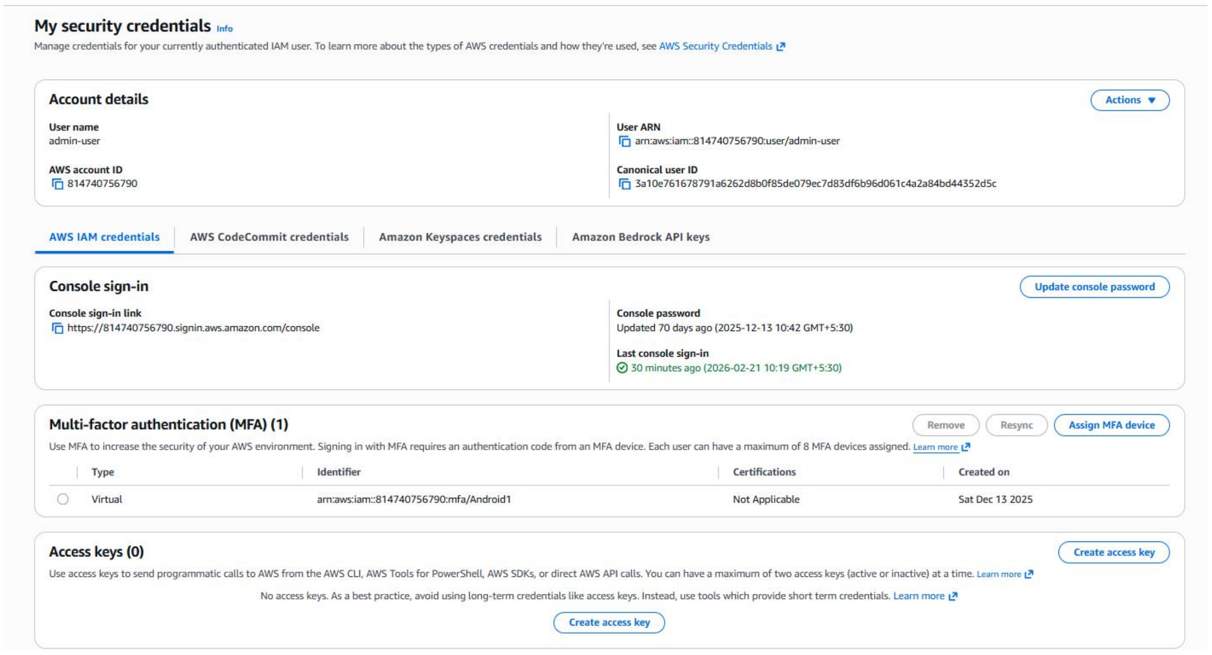
IAM user
admin-user

Account
Organization
Service Quotas
Billing and Cost Management
Security credentials
Console Mobile App

Step 2: Generate the Access Keys

Once that page loads, follow these instructions to create your keys:

- Scroll down until you see a section titled **Access keys**.



My security credentials Info

Manage credentials for your currently authenticated IAM user. To learn more about the types of AWS credentials and how they're used, see AWS Security Credentials

Account details

User name admin-user	**User ARN** arn:aws:iam::814740756790:user/admin-user
AWS account ID 814740756790	**Canonical user ID** 3a10e761678791a6262d8b0f85de079ec7d83df6b96d061c4a2a84bd44352d5c

Console sign-in

Console sign-in link https://814740756790.signin.aws.amazon.com/console

Console password Updated 70 days ago (2025-12-13 10:42 GMT+5:30)

Last console sign-in 30 minutes ago (2026-02-21 10:19 GMT+5:30)

Multi-factor authentication (MFA) (1)

Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. Learn more

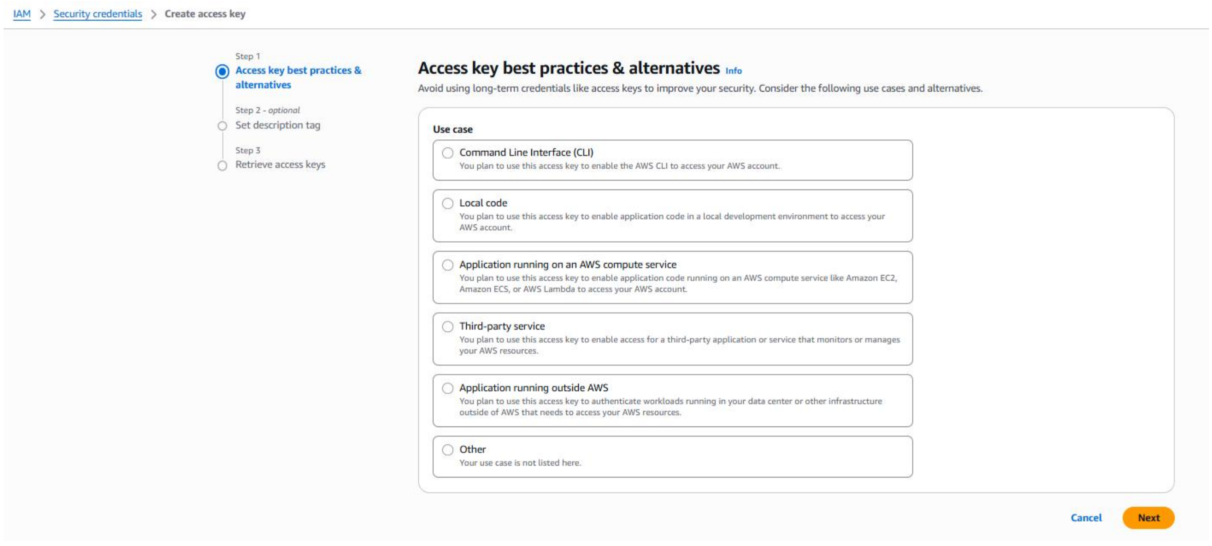
	Type	Identifier	Certifications	Created on
○	Virtual	arn:aws:iam::814740756790:mfa/Android1	Not Applicable	Sat Dec 13 2025

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. Learn more

- Click the **Create access key** button.



IAM > Security credentials > Create access key

Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

○ Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

○ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

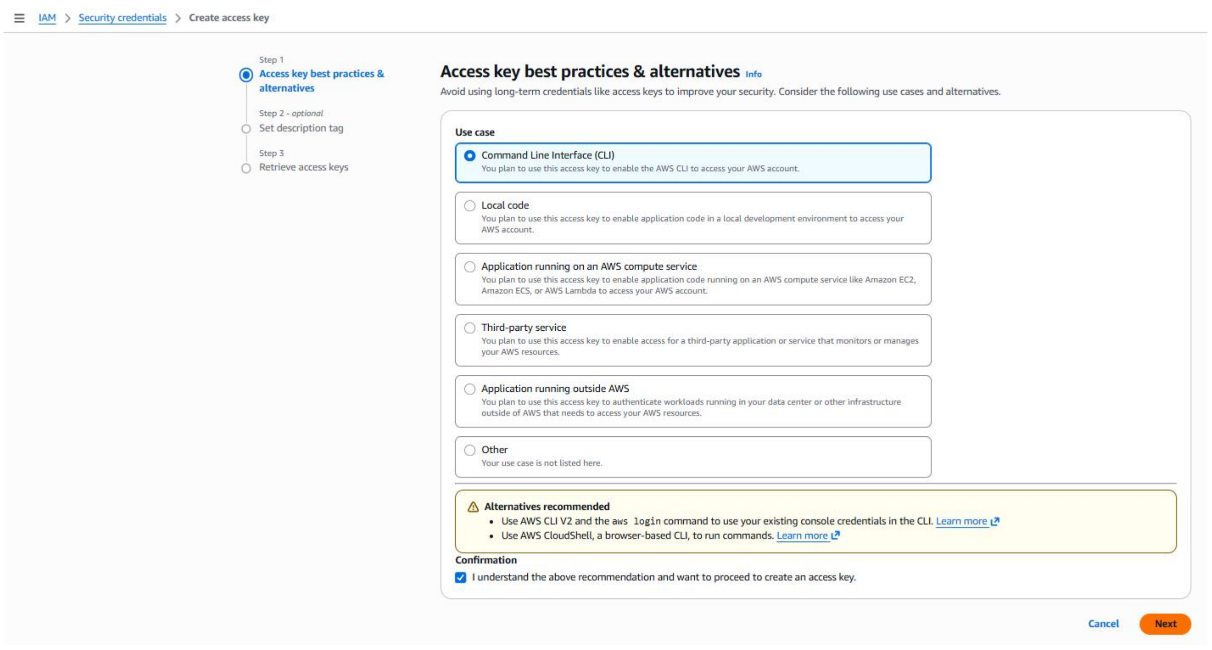
○ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

○ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

○ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

○ Other
Your use case is not listed here.

- You will be asked for a "Use case." Select **Command Line Interface (CLI)**.
- Check the box at the bottom that says "I understand the security recommendations..." and click **Next**.


IAM > Security credentials > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

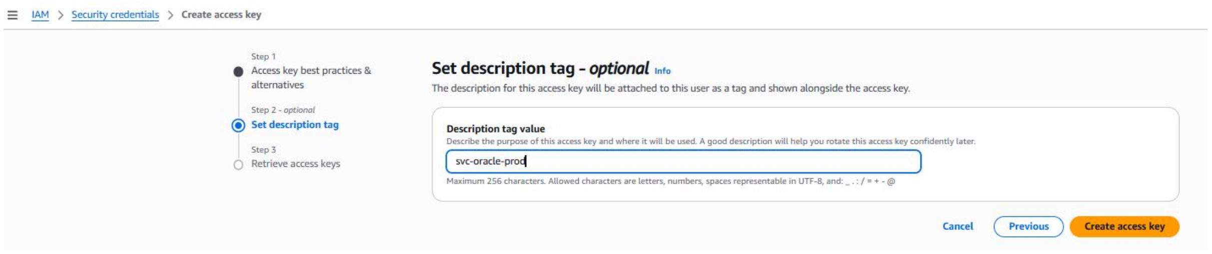
- [✓ Command Line Interface (CLI)] You plan to use this access key to enable the AWS CLI to access your AWS account.
- Local code: You plan to use this access key to enable application code in a local development environment to access your AWS account.
- Application running on an AWS compute service: You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- Third-party service: You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- Application running outside AWS: You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
- Other: Your use case is not listed here.

Alternatives recommended
- Use AWS CLI V2 and the `aws login` command to use your existing console credentials in the CLI. Learn more
- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more

Confirmation

[✓] I understand the above recommendation and want to proceed to create an access key.

(Optional) Give it a description tag, like "MyLaptopCLI," and click **Create access key**.


IAM > Security credentials > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Set description tag - *optional* Info

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value

Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

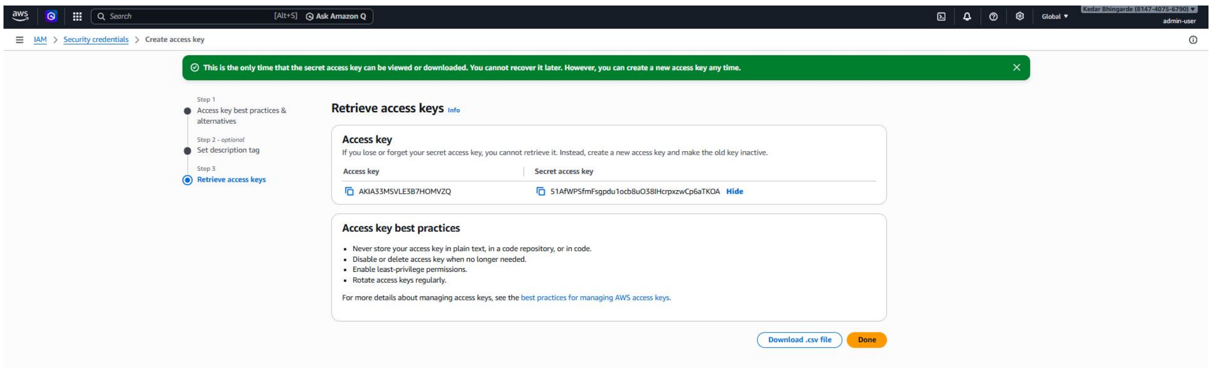
svc-oracle-prod

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Step 3: Save Your Keys

This is the most important part:

- **Access Key ID:** This will be a string of capital letters and numbers.
- **Secret Access Key:** Click **Show** to reveal it.
- **Action:** Click **Download .csv file**. AWS will never show you the "Secret" key again after you leave this screen. If you lose it, you'll have to delete this key and make a new one.



IAM > Security credentials > Create access key

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

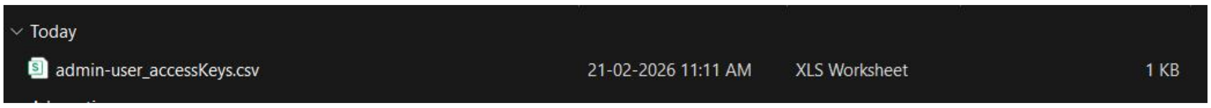
Access key	Secret access key
AKIA33MSVLE3B7HOMVZQ	51AfWPSfmFsgpdu1ocb8uO38IHcrpxzwCp6aTKOA Hide

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

admin-user_accessKeys.csv

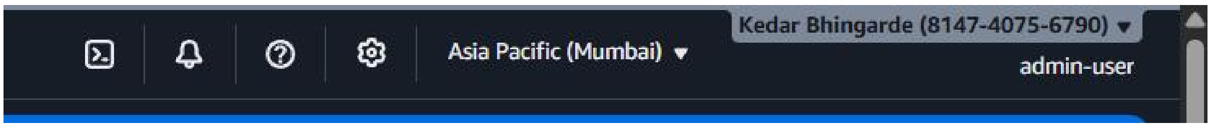

Today
admin-user_accessKeys.csv 21-02-2026 11:11 AM XLS Worksheet 1 KB

Step 4: Find your Region

Look at the URL in your browser address bar or the top right corner of your screen (usually next to the "Global" or "N. Virginia" text).

- If it says **N. Virginia**, your region is us-east-1.
- If it says **Mumbai**, your region is ap-south-1.

https://814740756790-wv76zkgm.us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/security_credentials/access-key-wizard

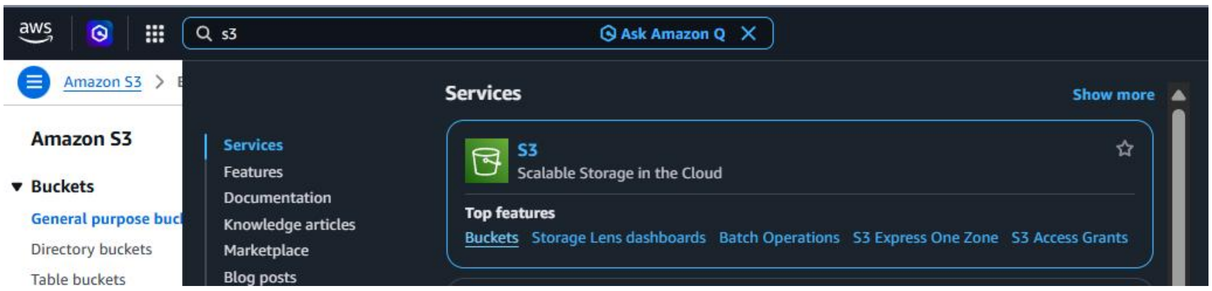


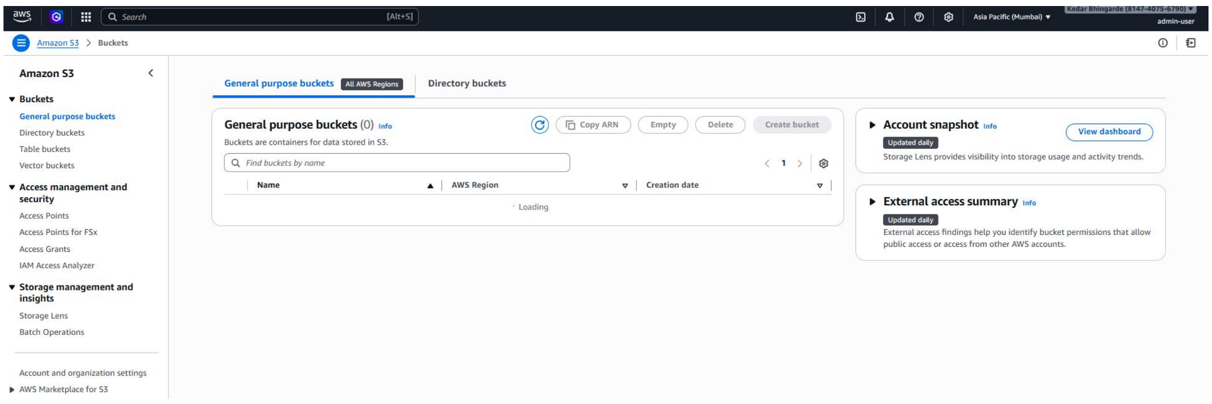
```
Default region name: ap-south-1
```

Lets' Create s3 bucket

Step 1: Go to the S3 Dashboard

1. Log in to your AWS Management Console.
2. In the search bar at the top, type S3 and select it.
3. You should see the page shown in your first image. Click the orange Create bucket button on the right side.




Search [Alt+S]

Asia Pacific (Mumbai) ▾ Kedar Bhingarde (8147-4075-6790) ▾ admin-user

Amazon S3 > Buckets

Amazon S3

▾ Buckets
- General purpose buckets
- Directory buckets
- Table buckets
- Vector buckets

▾ Access management and security
- Access Points
- Access Points for FSx
- Access Grants
- IAM Access Analyzer

▾ Storage management and insights
- Storage Lens
- Batch Operations

Account and organization settings

▸ AWS Marketplace for S3

General purpose buckets | All AWS Regions | Directory buckets

General purpose buckets (0) Info

Copy ARN | Empty | Delete | Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

Name	AWS Region	Creation date

Loading

▸ Account snapshot Info — View dashboard

Updated daily

Storage Lens provides visibility into storage usage and activity trends.

▸ External access summary Info

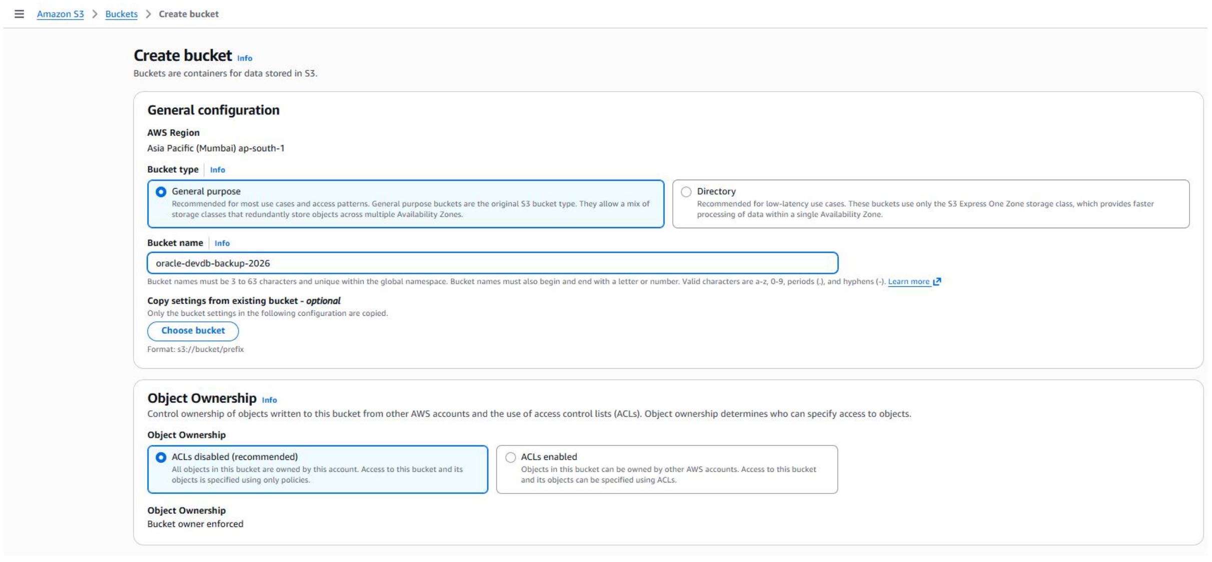
Updated daily

External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

Step 2: Configure Bucket Settings (General Configuration)

You will see a configuration page similar to your second image. Fill in the following:

1. Bucket name:
   - Choose a unique name (e.g., **oracle-devdb-backup-2026**).
   - Note: The name must be globally unique across all AWS customers, not just your account. If it says "Bucket name already exists," try adding numbers or your company name.
2. AWS Region:
   - Choose a region close to you (e.g., **US East (N. Virginia) us-east-1**). Keep this region in mind for your **aws configure** setup later.


Amazon S3 > Buckets > Create bucket

Create bucket Info

Buckets are containers for data stored in S3.

General configuration

AWS Region

Asia Pacific (Mumbai) ap-south-1

Bucket type Info

[✓ General purpose] Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Directory — Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name Info

oracle-devdb-backup-2026

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). Learn more

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Choose bucket

Format: s3://bucket/prefix

Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Object Ownership

[✓ ACLs disabled (recommended)] All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled — Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

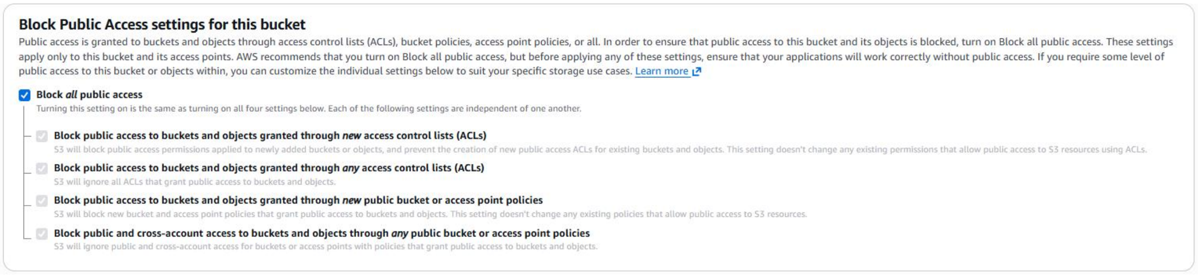
Object Ownership

Bucket owner enforced

Step 3: Block Public Access Settings

- Block Public Access settings for this bucket:
  - It is recommended to leave this as Block all public access (Checked).

- Since this is a database backup, you want it private. Do not uncheck these boxes unless you have a specific reason to make it public.



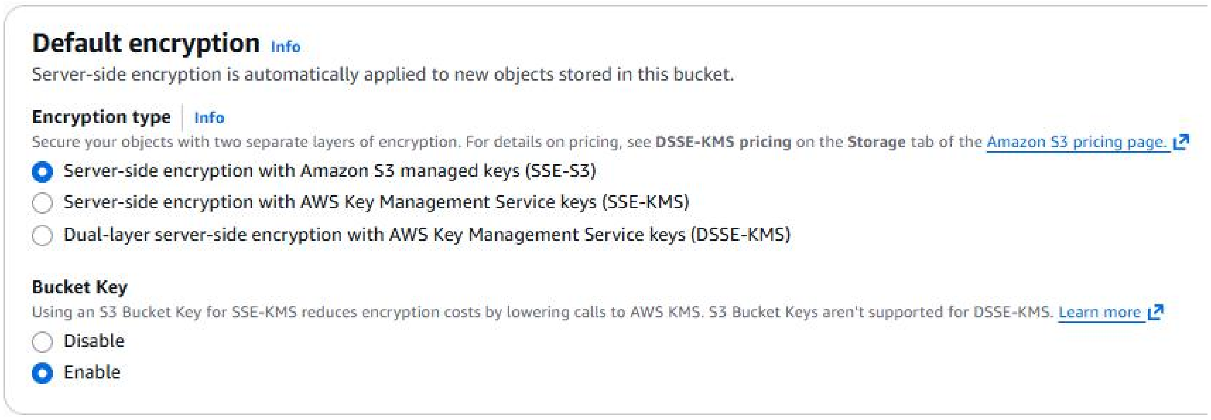
Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

- [✓] **Block *all* public access**
  Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.
  - [✓] **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
    S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
  - [✓] **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
    S3 will ignore all ACLs that grant public access to buckets and objects.
  - [✓] **Block public access to buckets and objects granted through *new* public bucket or access point policies**
    S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
  - [✓] **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
    S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Step 4: Bucket Versioning and Encryption

1. Bucket Versioning:
   - Select Disable. (You don't need versioning for simple backups; it costs extra).
2. Default encryption:
   - Select Enable.
   - Encryption key type: SSE-S3 is sufficient and easiest for backups.



Default encryption Info

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type Info

Secure your objects with two separate layers of encryption. For details on pricing, see **DSSE-KMS pricing** on the **Storage** tab of the Amazon S3 pricing page.

- ● Server-side encryption with Amazon S3 managed keys (SSE-S3)
- ○ Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- ○ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

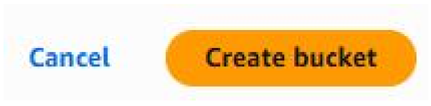
Bucket Key

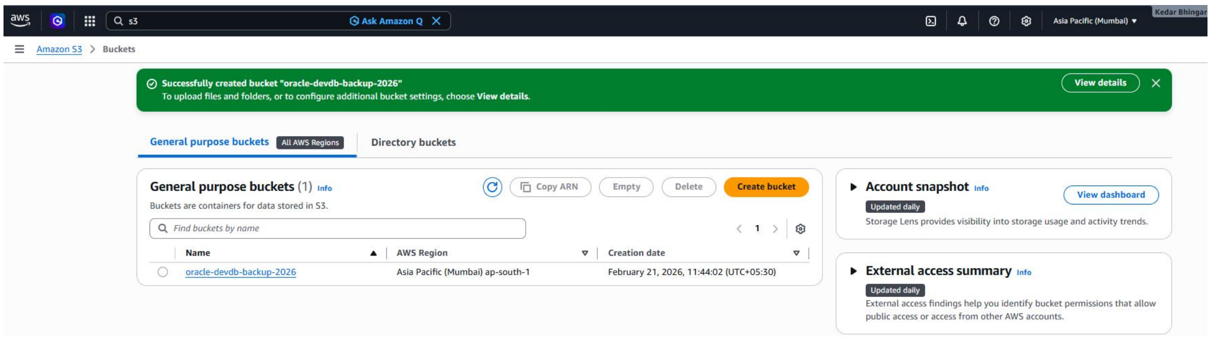
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. Learn more

- ○ Disable
- ● Enable

Step 5: Create the Bucket

1. Scroll to the very bottom of the page.
2. Click the orange Create bucket button.





Successfully created bucket "oracle-devdb-backup-2026"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

General purpose buckets (1) Info
Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
oracle-devdb-backup-2026	Asia Pacific (Mumbai) ap-south-1	February 21, 2026, 11:44:02 (UTC+05:30)

Account snapshot Info
Storage Lens provides visibility into storage usage and activity trends.

External access summary Info
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

Step 6: Update Your Script

Once the bucket is created (e.g., named **oracle-devdb-backup-2026**), update your script to match the name.

Open your script:

vi /u01/app/oracle/backup/run_backup.sh

Change the **S3_BUCKET** line to match your new bucket name:

S3_BUCKET="s3://oracle-devdb-backup-2026" # Use the actual name you created

Save and exit (**:wq**).

Important: AWS CLI Configuration

Before running the script, ensure your AWS CLI has permission to write to this bucket. Run this command if you haven't already:

aws configure

```
[oracle@oraclelab1 ~]$ cat /home/oracle/backup_to_s3.sh

#!/bin/bash


# ==========================================

# Configuration

# ==========================================

# Path confirmed by user

BACKUP_DIR="/u01/app/oracle/backup"


# AWS Configuration

S3_BUCKET="s3://oracle-devdb-back-up-2026"   # <--- CHANGE THIS

ZIP_PASSWORD="Password@123"     # <--- CHANGE THIS


# Environment Setup

export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
export PATH=$PATH:$ORACLE_HOME/bin
export ORACLE_SID=DEVDB

DATE_STAMP=$(date +%Y%m%d_%H%M%S)
BACKUP_PREFIX="devdb_backup_${DATE_STAMP}"

echo "========================================"
echo "Starting Backup: $(date)"
echo "Destination: $BACKUP_DIR"
echo "========================================"

# Ensure directory exists
mkdir -p "$BACKUP_DIR"

# ========================================
# Step 1: RMAN Backup
# ========================================
echo "Running RMAN Backup..."

rman target / <<EOF
RUN {
  ALLOCATE CHANNEL ch1 DEVICE TYPE DISK FORMAT
'${BACKUP_DIR}/${BACKUP_PREFIX}_%U.bak';
  BACKUP DATABASE;
  BACKUP CURRENT CONTROLFILE FORMAT '${BACKUP_DIR}/${BACKUP_PREFIX}_ctrl.bak';
  SQL 'ALTER SYSTEM ARCHIVE LOG CURRENT';
  BACKUP ARCHIVELOG ALL FORMAT '${BACKUP_DIR}/${BACKUP_PREFIX}_arch_%U.bak';
  RELEASE CHANNEL ch1;
}
EXIT;
```

```
EOF

if [ $? -ne 0 ]; then
   echo "ERROR: RMAN backup failed. Aborting script."
   exit 1
fi

echo "RMAN Backup Complete."

# ==========================================
# Step 2: Zip with Password
# ==========================================
echo "Compressing and Encrypting..."

ZIP_FILE="${BACKUP_DIR}/${BACKUP_PREFIX}.zip"

# -j: Junk paths (don't store full directory structure in zip)
# -r: Recursive
# -P: Password
zip -r -j -P "${ZIP_PASSWORD}" "${ZIP_FILE}" "${BACKUP_DIR}"/*.bak

if [ $? -ne 0 ]; then
   echo "ERROR: Zip failed. Check disk space in /u01."
   exit 1
fi

# Clean up raw .bak files to free space immediately
echo "Cleaning up temporary files..."
rm -f "${BACKUP_DIR}"/*.bak
```

```
# ==========================================
# Step 3: Upload to AWS S3
# ==========================================
echo "Uploading to S3 Bucket: ${S3_BUCKET}..."

aws s3 cp "${ZIP_FILE}" "${S3_BUCKET}/"

if [ $? -ne 0 ]; then
   echo "ERROR: S3 Upload failed. Check 'aws configure' setup."
   exit 1
fi

echo "Upload Successful."
echo "=========================================="
echo "Backup Finished: $(date)"
echo "File sent to S3: ${S3_BUCKET}/$(basename ${ZIP_FILE})"
echo "=========================================="


[oracle@oraclelab1 ~]$
```