Managing AWS Credentials for Boto3

1. How to Get AWS Credentials on AWS

To use AWS services programmatically, you need an **Access Key ID** and **Secret Access Key**. Follow these steps to obtain them:

Steps to Create AWS Credentials

1. **Log in to AWS Console**: Go to AWS IAM (https://console.aws.amazon.com/iam/)
2. **Navigate to IAM (Identity & Access Management)**
3. **Click on "Users"** in the left menu
4. **Select Your User** or create a new one
5. **Go to the "Security Credentials" Tab**
6. **Under "Access Keys," click "Create Access Key"**
7. **Download the `.csv` file** or copy the Access Key and Secret Key immediately

Important: AWS will only show the Secret Key once. If you lose it, you must generate a new key.

---

2. Setting Up AWS Credentials Locally

Once you have AWS credentials, you need to configure them on your local machine.

Option 1: Using AWS CLI (Recommended)

If you have the AWS CLI installed, configure credentials with:

```
aws configure
```

You'll be prompted to enter:

```
AWS Access Key ID: <YOUR_ACCESS_KEY>
AWS Secret Access Key: <YOUR_SECRET_KEY>
Default region name: us-east-1  # or your preferred AWS region
Default output format: json     # or text/table
```

This creates a credentials file stored at: - **Linux/macOS:** `~/.aws/credentials` - **Windows:** `C:\Users\YourUser\.aws\credentials`

To verify setup, run:

```
aws sts get-caller-identity
```

This should return your **AWS Account ID**.

Option 2: Manually Creating the Credentials File

If you don't want to use `aws configure`, manually create the credentials file:

Step 1: Locate the AWS Credentials File

Create (or edit) the following file: - **Linux/macOS:** `~/.aws/credentials` - **Windows:** `C:\Users\YourUser\.aws\credentials`

Step 2: Add Your Credentials

Open the file and add:

```
[default]
aws_access_key_id=YOUR_ACCESS_KEY
aws_secret_access_key=YOUR_SECRET_KEY
region=us-east-1  # Replace with your AWS region
```

To use multiple accounts, add additional profiles:

```
[my-profile]
aws_access_key_id=YOUR_OTHER_ACCESS_KEY
aws_secret_access_key=YOUR_OTHER_SECRET_KEY
region=us-west-2
```

To use a specific profile:

```
aws configure --profile my-profile
```

To verify:

```
aws sts get-caller-identity --profile my-profile
```

---

3. Using AWS Credentials in a Virtual Environment

When using **Python virtual environments (venv)**, AWS credentials are not automatically included, so they must be set up.

Option 1: Using Environment Variables (Temporary)

Each time you activate a virtual environment, set the credentials manually:

```
export AWS_ACCESS_KEY_ID=YOUR_ACCESS_KEY
export AWS_SECRET_ACCESS_KEY=YOUR_SECRET_KEY
export AWS_DEFAULT_REGION=us-east-1
```

For Windows (Command Prompt):

```
set AWS_ACCESS_KEY_ID=YOUR_ACCESS_KEY
set AWS_SECRET_ACCESS_KEY=YOUR_SECRET_KEY
set AWS_DEFAULT_REGION=us-east-1
```

For PowerShell:

```
$env:AWS_ACCESS_KEY_ID="YOUR_ACCESS_KEY"
$env:AWS_SECRET_ACCESS_KEY="YOUR_SECRET_KEY"
$env:AWS_DEFAULT_REGION="us-east-1"
```

To check if credentials are set:

```
echo $AWS_ACCESS_KEY_ID  # Linux/macOS
```

```
echo $env:AWS_ACCESS_KEY_ID  # Windows PowerShell
```

Option 2: Using the AWS Profile Inside Virtual Environments

Instead of setting credentials every time, use AWS profiles in your virtual environment:

```
pip install boto3
export AWS_PROFILE=my-profile  # Or use set for Windows
```

To verify:

```
import boto3
session = boto3.Session()
print(session.client("sts").get_caller_identity())
```

This will return your **AWS Account ID**, confirming your credentials are set up correctly.

Final Verification

To confirm AWS credentials are working in your environment, run:

```
import boto3
sts_client = boto3.client("sts")
account_id = sts_client.get_caller_identity()["Account"]
print(f"Authenticated AWS Account ID: {account_id}")
```

This will print your **AWS Account ID**, ensuring everything is correctly configured.

Following these steps ensures smooth authentication with AWS when using Boto3. 🚀