**AWS SAM (Serverless Application Model) CLI:**

AWS SAM provides you with a command line tool, the AWS SAM CLI, that makes it easy for you to create and manage serverless applications. You need to install and configure a few things in order to use the AWS SAM CLI.

**Installing the AWS SAM CLI on Linux**

The AWS SAM command line interface (CLI) is supported on 64-bit versions of recent distributions of CentOS, Fedora, Ubuntu, and Amazon Linux 2. To install the AWS SAM CLI, you must extract or "unzip" the downloaded package. If your operating system doesn't have the built-in unzip command, use an equivalent.

To install and configure the prerequisites for using the AWS SAM CLI on your Linux host, follow these steps:

* Create an AWS account.
* Configure AWS Identity and Access Management (IAM) permissions and AWS credentials.
* Install Docker. Note: Docker is a prerequisite only for testing your application locally or using the --use-container option.
* Install the AWS SAM CLI.

**Installing the AWS SAM CLI on Windows**

Follow these steps to install and configure the prerequisites for using the AWS SAM command line interface (CLI) on your Windows host:

* Create an AWS Identity and Access Management (AWS) account.
* Configure IAM permissions and AWS credentials.
* Install Docker. Note: Docker is a prerequisite only for testing your application locally or using the --use-container option.
* Install the AWS SAM CLI.

**Installing the AWS SAM CLI on macOS**

Follow these steps to install and configure the prerequisites for using the AWS SAM command line interface (CLI) on your macOS host:

* Create an AWS account.
* Configure AWS Identity and Access Management (IAM) permissions and AWS credentials.
* Install Docker. Note: Docker is a prerequisite only for testing your application locally or using the --use-container option
* Install Homebrew.
* Install the AWS SAM CLI.

AWS SAM CLI commands

* sam build
* sam delete
* sam deploy
* sam init
* sam local generate-event
* sam local invoke
* sam local start-api
* sam local start-lambda
* sam logs
* sam package
* sam pipeline bootstrap
* sam pipeline init
* sam publish
* sam sync
* sam traces
* sam validate

**SAM BUILD:**

The sam build command processes your AWS SAM template file, application code, and any applicable language-specific files and dependencies. The command also copies build artifacts in the format and location expected for subsequent steps in your workflow. You specify dependencies in a manifest file that you include in your application, such as requirements.txt for Python functions, or package.json for Node.js functions.

The format of your application's build artifacts depends on its package type. You specify your AWS Lambda function's package type with the PackageType property. The options are:

Zip – A .zip file archive, which contains your application code and its dependencies. If you package your code as a .zip file archive, you must specify a Lambda runtime for your function.

Image – A container image, which includes the base operating system, runtime, and extensions, in addition to your application code and its dependencies.

**Usage:**

sam build [OPTIONS] [RESOURCE\_LOGICAL\_ID]

**SAM DELETE:**

Deletes an AWS SAM application by deleting the AWS CloudFormation stack, the artifacts that were packaged and deployed to Amazon S3 and Amazon ECR, and the AWS SAM template file.

Also checks whether there is an Amazon ECR companion stack deployed, and if so prompts the user about deleting that stack and Amazon ECR repositories. If --no-prompts is specified, then companion stacks and Amazon ECR repositories are deleted by default.

**Usage:**

sam delete [OPTIONS]

**SAM DEPLOY:**

Deploys an AWS SAM application.

By default when you use this command, the AWS SAM CLI assumes that your current working directory is your project's root directory. The AWS SAM CLI first tries to locate a template file built using the sam build command, located in the .aws-sam subfolder, and named template.yaml. Next, the AWS SAM CLI tries to locate a template file named template.yaml or template.yml in the current working directory.

To override the AWS SAM CLI's default behavior, specify the --template option. When you specify this option, the AWS SAM CLI deploys only that AWS SAM template and the local resources that it points to.

To turn on the guided interactive mode, specify the --guided option. This mode shows you the parameters required for deployment, provides default options, and optionally saves these options in a configuration file in your project directory. When you perform subsequent deployments of your application using sam deploy, the AWS SAM CLI retrieves the required parameters from the configuration file.

Objects declared in the Parameters section of the AWS SAM template file appear as additional interactive mode prompts. The AWS SAM CLI prompts you to provide values for each parameter. For examples of these objects and the corresponding prompts, see the Examples section later in this topic.

Serverless applications that you configure with code signing generate more interactive mode prompts. The AWS SAM CLI asks whether you want your code to be signed, and if so, prompts you to enter signing profile names and owners. For examples of these prompts, see the Examples section later in this topic.

For more information about settings that the AWS SAM CLI optionally stores when you specify the --guided option, see AWS SAM CLI configuration file.

To deploy AWS Lambda functions through AWS CloudFormation, you must have an Amazon Simple Storage Service (Amazon S3) bucket for the Lambda deployment package. The AWS SAM CLI creates and manages this Amazon S3 bucket for you. AWS SAM enables encryption for all files stored in Amazon S3.

If your application includes any function or layer resources declared with PackageType: Image, then you can instruct the AWS SAM CLI to automatically create the required Amazon Elastic Container Registry (Amazon ECR) repositories for you. To have the AWS SAM CLI create these repositories, use either the --resolve-image-repos option or the --guided option, and then respond to the subsequent prompt with Y.

**Usage:**

sam deploy [OPTIONS] [ARGS]...

**SAM INIT:**

Initializes a serverless application with an AWS SAM template. The template provides a folder structure for your AWS Lambda functions, and is connected to event sources such as APIs, Amazon Simple Storage Service (Amazon S3) buckets, or Amazon DynamoDB tables. This application includes everything that you need to get started and to eventually extend it into a production-scale application.

For some sample applications, you can choose the package type of the application, either Zip or Image. For more information about Lambda package types, see Lambda deployment packages in the AWS Lambda Developer Guide.

**Usage:**

sam init [OPTIONS]

**SAM LOGS:**

Fetches logs that are generated by your Lambda function.

When your functions are a part of an AWS CloudFormation stack, you can fetch logs by using the function's logical ID when you specify the stack name.

**Usage:**

sam logs [OPTIONS]

**AWS CLI (COMMAND LINE INTERFACE):**

The AWS Command Line Interface (AWS CLI) is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

The AWS CLI v2 offers several new features including improved installers, new configuration options such as AWS IAM Identity Centre (successor to AWS SSO), and various interactive features.

**Install or update the AWS CLI:**

**Follow these steps from the command line to install the AWS CLI on Linux.**

* curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"
* unzip awscliv2.zip
* sudo ./aws/install

**Follow these steps from the command line to install the AWS CLI on MacOS.**

* curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
* sudo installer -pkg AWSCLIV2.pkg -target /

**Follow these steps from the command line to install the AWS CLI Windows.**

* Download and run the AWS CLI MSI installer for Windows (64-bit**):**

https://awscli.amazonaws.com/AWSCLIV2.msi

Alternatively, you can run the msiexec command to run the MSI installer.

msiexec.exe /i <https://awscli.amazonaws.com/AWSCLIV2.msi>

* To confirm the installation, open the **Start** menu, search for cmd to open a command prompt window, and at the command prompt use the aws --version command.

**Prerequisites to use the AWS CLI version 2:**

To access AWS services with the AWS CLI, you need an AWS account, IAM credentials, and an IAM access key pair. When running AWS CLI commands, the AWS CLI needs to have access to those AWS credentials.

To increase the security of your AWS account, we recommend that you do not use your root account credentials. You should create an IAM user to provide access credentials to the tasks you'll be running in AWS.

Step 1: Sign up to AWS

Step 2: Create an IAM user account

Step 3: Create an access key ID and secret access key