AWS CLI Cheatsheet

By Dejan Panovski • Updated on Jul 31, 2026 •

Quick reference for AWS CLI commands, profiles, and output filtering

The AWS CLI puts every AWS API behind a single aws command, so you can manage instances, buckets, secrets, and logs from your shell. This cheatsheet covers AWS CLI version 2 syntax for authentication, profiles, output filtering, and the services Linux administrators reach for most.

Installation and Setup

Install AWS CLI version 2 and set the basic configuration. Full walkthrough in installing and configuring the AWS CLI on Linux .

`curl https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip -o awscliv2.zip`	Download the x86_64 installer
`curl https://awscli.amazonaws.com/awscli-exe-linux-aarch64.zip -o awscliv2.zip`	Download the ARM64 installer
`unzip awscliv2.zip`	Extract the installer
`sudo ./aws/install`	Install AWS CLI version 2
`sudo ./aws/install --update`	Update from a freshly extracted installer
`aws --version`	Show the installed version
`aws configure`	Interactive credential and region setup
`aws configure list`	Show active settings and where they come from
`aws configure get region`	Read a single config value
`aws configure set region eu-central-1`	Write a single config value
`complete -C aws_completer aws`	Enable Bash command completion
`aws ec2 help`	Open the reference for a service

Profiles and Authentication

Static access keys configured with `aws configure` live in `~/.aws/credentials`, while other settings live in `~/.aws/config`. Prefer temporary credentials from `aws login` or IAM Identity Center.

`aws login`	Sign in with AWS console credentials
`aws login --profile dev`	Sign in to a named profile
`aws login --remote`	Sign in from a host without a browser
`aws logout --profile dev`	Clear cached login credentials for a profile
`aws configure --profile dev`	Configure a profile with an access key
`aws s3 ls --profile dev`	Run one command as a profile
`export AWS_PROFILE=dev`	Use a profile for the whole shell
`aws configure list-profiles`	List configured profiles
`aws configure sso`	Set up an IAM Identity Center profile
`aws sso login --profile dev`	Start or refresh an SSO session
`aws sso logout`	Clear cached SSO credentials
`aws sts get-caller-identity`	Show the account and identity in use
`aws sts assume-role --role-arn arn --role-session-name cli`	Get temporary role credentials
`export AWS_REGION=eu-west-1`	Override the region for the shell

Common Options

Frequently used options. Availability depends on the service and operation.

`--region eu-west-1`	Target a specific region
`--profile dev`	Use a named profile
`--output json`	Set the response format
`--no-cli-pager`	Print output instead of opening a pager
`--dry-run`	Check permissions without acting on supported EC2 operations
`--cli-auto-prompt`	Prompt for parameters interactively
`--no-paginate`	Return only the first page from a paginated operation
`--page-size 100`	Set the API page size for a paginated operation
`--max-items 20`	Limit output from a paginated operation
`--cli-input-json file://params.json`	Read parameters for a modeled API operation
`--generate-cli-skeleton`	Print a parameter template for a modeled API operation
`--debug`	Print the full request and response trace

Output Formatting and Queries

The `--query` option uses JMESPath and runs client side. Service-specific options such as `--filters` run server side, which can reduce response size and improve response time for large data sets. Pipe `--output json` into `jq` when a query gets hard to read.

`aws s3api list-buckets --output table`	Human-readable table
`aws s3api list-buckets --output text`	Tab-delimited text
`aws s3api list-buckets --output yaml`	YAML response
`aws s3api list-buckets --query "Buckets[].Name"`	Return one field
`aws ec2 describe-instances --query "Reservations[].Instances[].[InstanceId,State.Name]" --output text`	Return several fields as columns
`aws ec2 describe-instances --query "Reservations[].Instances[?State.Name=='running'].InstanceId"`	Filter results client side
`aws ec2 describe-instances --filters Name=instance-state-name,Values=running`	Filter results server side
`aws ec2 describe-volumes --query "Volumes[0:5]"`	Slice the result list
`aws ec2 wait instance-running --instance-ids i-0abc123`	Block until a state is reached

S3 Buckets and Objects

Everyday transfers with the high-level `aws s3` commands. More examples in aws s3 commands .

`aws s3 ls`	List all buckets
`aws s3 ls s3://bucket --recursive --summarize`	List objects with a size total
`aws s3 cp file.txt s3://bucket/`	Upload a file
`aws s3 cp s3://bucket/file.txt .`	Download a file
`aws s3 cp ./dir s3://bucket/dir --recursive`	Upload a directory
`aws s3 sync ./dir s3://bucket/dir`	Copy only new and changed files
`aws s3 sync ./dir s3://bucket/dir --delete --dryrun`	Preview a mirror that removes stale files
`aws s3 rm s3://bucket/dir/ --recursive --dryrun`	Preview a recursive delete
`aws s3 mb s3://bucket`	Create a bucket
`aws s3 rb s3://bucket`	Remove an empty bucket
`aws s3 presign s3://bucket/file.txt --expires-in 3600`	Generate a temporary download link

EC2 Instances

Launch, inspect, and control instances.

`aws ec2 describe-instances`	List instances and their details
`aws ec2 describe-instances --instance-ids i-0abc123`	Details for one instance
`aws ec2 describe-instance-status`	Health and scheduled events
`aws ec2 run-instances --image-id ami-0abc --instance-type t3.micro --key-name mykey`	Launch an instance
`aws ec2 start-instances --instance-ids i-0abc123`	Start a stopped instance
`aws ec2 stop-instances --instance-ids i-0abc123`	Stop an instance
`aws ec2 reboot-instances --instance-ids i-0abc123`	Reboot an instance
`aws ec2 terminate-instances --instance-ids i-0abc123`	Permanently terminate an instance
`aws ec2 create-tags --resources i-0abc123 --tags Key=Name,Value=web`	Tag a resource
`aws ec2 describe-images --owners 099720109477 --filters "Name=name,Values=ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-*"`	Find official Ubuntu 24.04 AMIs
`aws ec2 get-console-output --instance-id i-0abc123`	Read the instance console log

Security Groups and Key Pairs

Network access and SSH keys for EC2. Private key files are unencrypted, so keep them readable only by your user.

`aws ec2 describe-security-groups`	List security groups
`aws ec2 create-security-group --group-name web --description "Web tier"`	Create a security group
`aws ec2 authorize-security-group-ingress --group-id sg-0abc --protocol tcp --port 22 --cidr 203.0.113.10/32`	Allow inbound traffic
`aws ec2 revoke-security-group-ingress --group-id sg-0abc --protocol tcp --port 22 --cidr 203.0.113.10/32`	Remove an inbound rule
`aws ec2 delete-security-group --group-id sg-0abc`	Delete a security group
`aws ec2 create-key-pair --key-name mykey --query KeyMaterial --output text > mykey.pem`	Create a key pair and save the private key
`chmod 400 mykey.pem`	Restrict private key permissions
`aws ec2 describe-key-pairs`	List key pairs
`aws ec2 delete-key-pair --key-name mykey`	Delete a key pair
`aws ec2 describe-vpcs`	List VPCs
`aws ec2 describe-subnets --filters Name=vpc-id,Values=vpc-0abc`	List subnets in a VPC
`aws ec2 allocate-address`	Reserve an elastic IP
`aws ec2 associate-address --instance-id i-0abc123 --allocation-id eipalloc-0abc`	Attach an elastic IP

EBS Volumes and Snapshots

Block storage and backups.

`aws ec2 describe-volumes`	List volumes
`aws ec2 create-volume --size 20 --availability-zone eu-central-1a`	Create a volume
`aws ec2 attach-volume --volume-id vol-0abc --instance-id i-0abc123 --device /dev/sdf`	Attach a volume
`aws ec2 detach-volume --volume-id vol-0abc`	Detach a volume
`aws ec2 delete-volume --volume-id vol-0abc`	Permanently delete a volume
`aws ec2 create-snapshot --volume-id vol-0abc --description "nightly"`	Snapshot a volume
`aws ec2 describe-snapshots --owner-ids self`	List your snapshots
`aws ec2 copy-snapshot --region eu-west-1 --source-region eu-central-1 --source-snapshot-id snap-0abc`	Copy a snapshot from eu-central-1 to eu-west-1
`aws ec2 delete-snapshot --snapshot-id snap-0abc`	Permanently delete a snapshot

IAM Users and Roles

Identities, policies, and access keys. Creating an access key prints its secret once, so store it securely and never commit it to version control.

`aws iam list-users`	List IAM users
`aws iam create-user --user-name dev`	Create a user
`aws iam delete-user --user-name dev`	Permanently delete a user after removing dependencies
`aws iam list-roles`	List roles
`aws iam get-role --role-name deploy`	Show a role and its trust policy
`aws iam attach-user-policy --user-name dev --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess`	Attach a managed policy
`aws iam list-attached-user-policies --user-name dev`	List policies attached to a user
`aws iam create-access-key --user-name dev`	Create an access key
`aws iam list-access-keys --user-name dev`	List access keys and creation dates
`aws iam update-access-key --user-name dev --access-key-id AKIA123 --status Inactive`	Disable a key before deleting it
`aws iam delete-access-key --user-name dev --access-key-id AKIA123`	Permanently delete an access key
`aws iam get-account-summary`	Account-wide IAM counts and limits

Systems Manager and Secrets

Shell access without SSH, plus configuration and secret storage. The `start-session` command needs the Session Manager plugin installed locally. Pass secret values from protected files so they do not enter shell history, and keep decrypted output out of shared logs.

`aws ssm start-session --target i-0abc123`	Open a shell on an instance
`aws ssm describe-instance-information`	List instances managed by SSM
`aws ssm send-command --instance-ids i-0abc123 --document-name AWS-RunShellScript --parameters commands="uptime"`	Run a command on an instance
`aws ssm get-command-invocation --command-id 1a2b --instance-id i-0abc123`	Read the output of a sent command
`aws ssm put-parameter --name /app/db_url --value file://db-url.txt --type SecureString`	Store an encrypted parameter from a file
`aws ssm get-parameter --name /app/db_url --with-decryption`	Read and decrypt a parameter
`aws ssm get-parameters-by-path --path /app --recursive`	List parameters under a path
`aws secretsmanager list-secrets`	List secrets
`aws secretsmanager get-secret-value --secret-id prod/db --query SecretString --output text`	Print a decrypted secret value
`aws secretsmanager create-secret --name prod/db --secret-string file://secret.json`	Create a secret from a file

CloudWatch Logs and Alarms

Read application and service logs from the terminal.

`aws logs describe-log-groups`	List log groups
`aws logs describe-log-streams --log-group-name /aws/lambda/fn`	List streams in a group
`aws logs tail /aws/lambda/fn --follow`	Stream new log events live
`aws logs tail /aws/lambda/fn --since 1h --format short`	Show the last hour of logs
`aws logs filter-log-events --log-group-name /aws/lambda/fn --filter-pattern ERROR`	Search log events
`aws logs create-log-group --log-group-name /app/web`	Create a log group
`aws logs put-retention-policy --log-group-name /app/web --retention-in-days 30`	Set log retention
`aws logs delete-log-group --log-group-name /app/web`	Permanently delete a log group and its logs
`aws cloudwatch describe-alarms`	List alarms and their state
`aws cloudwatch describe-alarms --state-value ALARM`	Show only firing alarms

Lambda Functions

Deploy and invoke functions. AWS CLI version 2 expects base64 input for blob parameters by default, so literal JSON passed to `--payload` needs `--cli-binary-format raw-in-base64-out`.

`aws lambda list-functions`	List functions
`aws lambda get-function --function-name fn`	Show configuration and code location
`aws lambda invoke --function-name fn out.json`	Invoke a function
`aws lambda invoke --function-name fn --cli-binary-format raw-in-base64-out --payload '{"key":"value"}' out.json`	Invoke with a JSON payload
`aws lambda update-function-code --function-name fn --zip-file fileb://fn.zip`	Deploy new code
`aws lambda update-function-configuration --function-name fn --timeout 30`	Change a setting
`aws lambda publish-version --function-name fn`	Publish an immutable version
`aws lambda list-versions-by-function --function-name fn`	List published versions
`aws lambda delete-function --function-name fn`	Permanently delete a function

ECR Container Images

Authenticate Docker against a registry with `aws ecr get-login-password --region eu-central-1` piped into `docker login --username AWS --password-stdin 123456789012.dkr.ecr.eu-central-1.amazonaws.com`. Container commands are in the Docker cheatsheet .

`aws ecr get-login-password --region eu-central-1`	Print a registry password for Docker
`aws ecr describe-repositories`	List repositories
`aws ecr create-repository --repository-name app`	Create a repository
`aws ecr list-images --repository-name app`	List image tags and digests
`aws ecr describe-images --repository-name app`	Show image size and push date
`aws ecr batch-delete-image --repository-name app --image-ids imageTag=old`	Delete an image
`aws ecr delete-repository --repository-name app --force`	Permanently delete a repository and its images