

INFRASTRUCTURE AS CODE · FIELD GUIDE

AWS IaC with Python

Understand CloudFormation. Use AWS CDK safely. Build and operate reproducible infrastructure.

How to use this guide

This guide is both a learning path and a reference. Read topics 02 through 05 first, then build a small project and use the remaining topics while you work.

01	Tool selection	03	02	The mental model	04
03	Set up your workstation	05	04	CDK project with Python	06
05	Secure CDK stack	07	06	Constructs and tokens	08
07	Compose stacks	09	08	Tests and policy as code	10
09	CloudFormation anatomy	11	10	Intrinsic functions and conditions	12
11	Deployments and drift	13	12	IAM and secrets	14
13	Lifecycle and refactoring	15	14	Multi-account and regions	16
15	CI/CD	17	16	Operations and cost	18
17A	AWS SAM	19	17B	Terraform and tool selection	20
18A	Failure patterns	21	18B	Diagnostic workflow	22
19	CDK commands	23	20	CloudFormation commands	25
21	30-day learning path	27	22	Glossary and sources	28

Practice rule

Every change follows the same loop: **Code** → **synthesis** → **test** → **diff** → **deployment** → **verification**. Do not make parallel console changes to the same resources.

Getting started

Topics 03 through 13, followed by exercise 1 in topic 27.

Practice

Topics 14 through 22 for security, operations and failures.

Reference

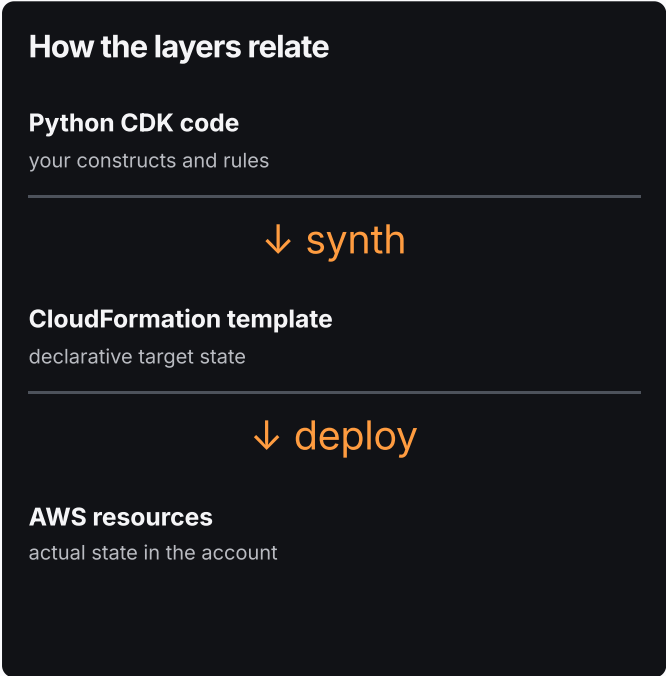
Keep topics 23 through 26 and 28 open beside your terminal.

01 · DECIDE

Choose a tool by operating model

For an AWS-only project using Python, CDK v2 is usually the best starting point. CloudFormation remains the execution and state foundation.

TOOL	STRENGTHS	SUITABLE FOR	TRADE-OFF
AWS CDK v2 RECOMMENDED	Python, abstractions, reuse, tests, native AWS integration	AWS-only environments, more complex systems, development teams	The synthesis and token model must be understood
CloudFormation	Declarative, AWS-native, no external state backend	Small stacks, existing YAML/JSON templates, platform teams	More repetition; large templates become difficult to read
AWS SAM	Concise serverless syntax, local test and deployment commands	Lambda, API Gateway, event-driven applications	Focused on serverless; CloudFormation remains underneath
Terraform	Multi-cloud, broad provider ecosystem	Hybrid and multi-cloud environments, established Terraform teams	Separate state management and HCL instead of Python



Choose CDK when ...

- Python is your working language.
- you want to encapsulate secure standards.
- you need reusable constructs.
- you want to test infrastructure with unit tests.

Use CloudFormation directly when ...

- an existing template estate already exists.
- the stack stays small and declarative.
- no programming logic is required.

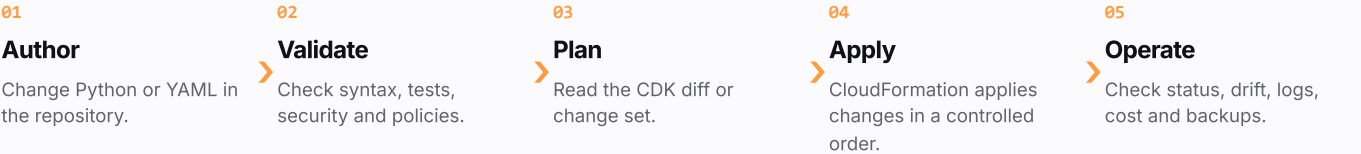
Do not mix tools without a boundary

A resource has exactly one IaC owner. CDK, Terraform and manual changes must not manage the same resource at the same time.

Working definition: IaC describes a versioned and verifiable target state. A provisioning script by itself does not meet that requirement.

02 · FOUNDATIONS

From source code to actual AWS state



The five states you compare

Source	Code in the Git commit
Synthesis	generated CloudFormation template
Change Set	planned operations
Stack State	CloudFormation managed state
Actual State	actual resource properties

Differences between the stack definition and actual state are called **Drift**.

CloudFormation reasons about resources

- **Logical ID:** Identity in the template. Changing it can trigger replacement.
- **Physical ID:** actual name or ARN in AWS.
- **Dependency:** determines creation and deletion order.
- **Update behavior:** in place, interruption or replacement.
- **Rollback:** attempts to return to the previous state after a failure.

Terms to understand early

Stack Deployment and lifecycle boundary.	Construct CDK building block made from one or more resources.	Asset File, Lambda code or container image used for deployment.	Bootstrap CDK support resources for each account and region.
Token A value resolved only during synthesis or deployment.	Context Lookup or configuration value used for deterministic synthesis.	Change Set Preview of planned CloudFormation changes.	Drift Manual or external deviation from the IaC target state.

Completion criteria

Code committed, tests green, diff reviewed, deployment successful, application verified, alarms quiet and no unexpected drift.

03 · PREPARATION

Local tools, profile and target environment

Prerequisites

- Python 3.11 or a newer version defined by the project
- Node.js for the CDK CLI
- AWS CLI v2
- AWS CDK CLI v2
- Git and an editor with Python and YAML support
- SSO, OIDC or short-lived roles instead of static keys

Verify identity before every change

```
aws sts get-caller-identity
aws configure list
aws configure get region

# Explicitly set values when using multiple environments:
$env:AWS_PROFILE = "sandbox-admin"
$env:AWS_REGION = "eu-central-1"
```

Never guess: Explicitly confirm the account ID, role and region before a deployment.

PowerShell · new Python CDK project

local

```
# Create the working directory and initialise the CDK app
New-Item -ItemType Directory aws-iac-lab
Set-Location aws-iac-lab
cdk init app --language python

# Activate the virtual environment
.\.venv\Scripts\Activate.ps1
python -m pip install --upgrade pip
python -m pip install -r requirements.txt

# Bootstrap the target environment once for CDK
cdk bootstrap aws://123456789012/eu-central-1

# Check early
cdk doctor
cdk synth
```

Pin versions

`requirements.txt` or commit a lockfile. Keep the CLI and library compatible.

Commit context

`cdk.context.json` when lookups are used. This keeps synthesis reproducible.

Ignore generated artifacts

`.venv/`, `cdk.out/`, caches and local secrets do not belong in Git.

Bootstrap is infrastructure

The bootstrap stack contains asset storage and roles, among other resources. Use trusted accounts, suitable permissions boundaries and consistent qualifiers.

04 · STRUCTURE

Project structure for maintainable CDK stacks

Recommended structure

```
aws-iac-lab/
├─ app.py
├─ cdk.json
├─ cdk.context.json
├─ requirements.txt
├─ requirements-dev.txt
├─ pytest.ini
├─ src/
│  └─ network_stack.py
│  └─ data_stack.py
│  └─ app_stack.py
│     └─ constructs/
│        └─ secure_bucket.py
├─ tests/
│  └─ test_data_stack.py
│  └─ test_app_stack.py
└─ README.md
```

app.py

Composition, environments, global tags and stack dependencies. Keep extensive resource logic elsewhere.

src/*_stack.py

Divide stacks by lifecycle and ownership: network, data, application and observability.

src/constructs/

Reusable secure building blocks with a clear API and tests.

tests/

Template assertions and rules. Test outcomes, not implementation details.

Minimal entry point

app.py

Python

```
import os
import aws_cdk as cdk
from src.data_stack import DataStack

app = cdk.App()
environment = cdk.Environment(
    account=os.getenv("CDK_DEFAULT_ACCOUNT"),
    region=os.getenv("CDK_DEFAULT_REGION", "eu-central-1"),
)

stack = DataStack(
    app,
    "LabData",
    env=environment,
    termination_protection=True,
    description="OeXYZ AWS IaC learning data stack",
)
cdk.Tags.of(stack).add("Project", "aws-iac-lab")
cdk.Tags.of(stack).add("ManagedBy", "cdk")

app.synth()
```

Deterministic

The same commit and context produce the same template.

No runtime logic

CDK code runs before deployment. It does not respond to later AWS events.

05 · PYTHON

Example: encrypted S3 data stack

src/data_stack.py

AWS CDK v2 · Python

```
from aws_cdk import (
    CfnOutput, RemovalPolicy, Stack,
    aws_kms as kms,
    aws_s3 as s3,
)
from constructs import Construct

class DataStack(Stack):
    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        key = kms.Key(
            self,
            "DataKey",
            alias="alias/aws-iac-lab-data",
            description="KMS key for the AWS IaC learning data stack",
            enable_key_rotation=True,
            removal_policy=RemovalPolicy.RETAIN,
        )

        bucket = s3.Bucket(
            self,
            "DataBucket",
            block_public_access=s3.BlockPublicAccess.BLOCK_ALL,
            encryption=s3.BucketEncryption.KMS,
            encryption_key=key,
            enforce_ssl=True,
            versioned=True,
            removal_policy=RemovalPolicy.RETAIN,
        )

        CfnOutput(
            self,
            "BucketArn",
            value=bucket.bucket_arn,
            description="ARN of the encrypted data bucket",
        )
```

No fixed bucket name

CloudFormation generates a unique physical name. This improves portability and prevents collisions.

Data is retained

`RETAIN` protects data when the stack is deleted or the resource is replaced. Deletion then becomes a deliberate separate operation.

Key rotation enabled

The KMS key and bucket encryption are explicit. Access is granted later with `grant_*`.

Development versus production

For short-lived learning resources, you may deliberately use `RemovalPolicy.DESTROY`. For S3, you must also understand `auto_delete_objects=True` and the cost impact. In production data stacks, `RETAIN` is the safe baseline.

Verification steps for this example

```
python -m pytest
cdk synth --strict
cdk diff LabData
cdk deploy LabData --require-approval broadening
```

06 · ABSTRACTION

Construct levels and deployment tokens

L1

Close to CloudFormation

`CfnBucket`, `CfnRole`. Direct representation of the resource specification. Maximum control, little convenience.

L2

Intent-based

`Bucket`, `Function`, `Vpc`. Safe helpers, grants and useful APIs. The default choice.

L3

Pattern

Multiple resources for one use case. Fast, but more opinionated and broader in scope.

Tokens: values from the future

```
bucket = s3.Bucket(self, "Data")

# No physical name exists while the Python code runs:
name = bucket.bucket_name

CfnOutput(self, "Name", value=name)
```

The bucket name is a token. CDK serializes a reference into the template, so string output during synthesis may show placeholders.

Escape hatch

```
cfn_bucket = bucket.node.default_child
cfn_bucket.add_property_override(
    "Metadata.Owner", "platform"
)
```

Use only when the L2 construct does not yet expose a new property. Test the override and remove it later.

Design the construct API

Inputs

- domain options instead of raw property collections
- secure defaults
- small, typed props

Outputs

- interfaces such as `IBucket`
- ARNs and tokens only when needed
- no secrets in outputs

Boundaries

- one clear responsibility
- no hidden cross-account side effects
- documented cost and deletion impact

A construct ID is identity

Renaming or moving a construct can change its logical ID and replace a resource. Before refactoring, always read `cdk diff`.

Review note: An abstraction may encapsulate repetition. Its cost, deletion and permission impact must remain visible.

Stack boundaries and coupling

Network

VPC, subnets, endpoints and central security groups.

RARELY CHANGED

Data

KMS, S3, databases and backups.

STATEFUL

App

Compute, API, queues and roles.

FREQUENTLY CHANGED

Ops

Logs, alarms, dashboards and topics.

OBSERVED

Strong, typed reference

app.py

same account and region

```
data = DataStack(app, "Data", env=env)

api = ApiStack(
    app,
    "Api",
    data_bucket=data.bucket,
    env=env,
)

# CDK creates the required export/import relationship.
```

api_stack.py

Least privilege

```
def __init__(..., data_bucket: s3.IBucket, **kwargs):
    super().__init__(..., **kwargs)
    fn = lambda_.Function(...)
    data_bucket.grant_read(fn)
```

Choose coupling deliberately

Direct reference	Typed and convenient, but creates deployment coupling.
SSM parameter	Loose coupling through a name or ARN. Useful for independent deployments.
CloudFormation export	Native, but the export cannot be removed while imports still exist.
Lookup	For existing resources; commit context and refresh changes deliberately.

Cross-stack deadlock

An export reference in use cannot be deleted directly. Decouple and deploy the consumer first, then remove the export.

Dependency rules

Automatic

Resource attributes and grants usually create the appropriate dependency.

Explicit

`stack.add_dependency(other)` only for genuine ordering requirements without a reference.

Avoid

Cycles. Move a shared resource into a third stack or couple it loosely.

Template assertions with pytest

tests/test_data_stack.pypytest

```
import aws_cdk as cdk
from aws_cdk.assertions import Match, Template
from src.data_stack import DataStack

def test_bucket_is_encrypted_and_versioned():
    app = cdk.App()
    stack = DataStack(app, "TestData")
    template = Template.from_stack(stack)

    template.has_resource_properties(
        "AWS::S3::Bucket",
        {
            "VersioningConfiguration": {
                "Status": "Enabled"
            },
            "PublicAccessBlockConfiguration":
                Match.object_like({
                    "BlockPublicAcls": True,
                    "BlockPublicPolicy": True,
                })
        },
    )
    template.resource_count_is("AWS::S3::Bucket", 1)
```

laC test pyramid

Unit	Template assertions, fast, on every commit
Policy	cfn-guard, cdk-nag and organization rules
Snapshot	detect broad changes; never update blindly
Integration	deploy to a sandbox and verify AWS behavior
Smoke	verify the endpoint or resource after deployment

Enable cdk-nag

```
from aws_cdk import Aspects
from cdk_nag import AwsSolutionsChecks

Aspects.of(app).add(AwsSolutionsChecks())
```

Pin the dependency separately. Use suppressions only with a specific justification and the smallest possible scope.

Validation gates

01	02	03	04	05	06
Format	Types	Tests	Synth	Policy	Diff
ruff / black	> mypy	> pytest	> strict	> nag / guard	> review

Test the desired properties

"The bucket is encrypted and not public" is more stable than "The logical ID is exactly DataBucketE3889A50."

Anatomy of a CloudFormation template

template.yamlCloudFormation

```
AWSTemplateFormatVersion: "2010-09-09"
Description: OeXYZ AWS IaC learning storage stack

Parameters:
  Environment:
    Type: String
    AllowedValues: [dev, stage, prod]
    Default: dev

Conditions:
  IsProduction: !Equals [!Ref Environment, prod]

Resources:
  DataBucket:
    Type: AWS::S3::Bucket
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              SSEAlgorithm: AES256
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
      VersioningConfiguration:
        Status: Enabled
      Tags:
        - Key: Environment
          Value: !Ref Environment
        - Key: ManagedBy
          Value: cloudformation

Outputs:
  DataBucketArn:
    Description: ARN of the encrypted data bucket
    Value: !GetAtt DataBucket.Arn
    Export:
      Name: !Sub "${AWS::StackName}-DataBucketArn"
```

Parameters Actual deployment inputs. Use constraints instead of unrestricted values.	Mappings Static mapping, such as a region to an AMI or configuration.	Conditions Create resources or properties conditionally.	Resources The only required section. Keep logical IDs stable.
Outputs Non-sensitive results and optional exports.	Metadata Tool metadata, never secrets.	Rules Validate parameter combinations before creation.	Transform Apply SAM, Include or macros before processing.

10 · LANGUAGE

Intrinsic functions, pseudo parameters and conditions

SHORT FORM	MEANING	EXAMPLE
<code>!Ref</code>	Parameter value or primary value of a resource	<code>!Ref Environment</code>
<code>!GetAtt</code>	Resource attribute	<code>!GetAtt DataBucket.Arn</code>
<code>!Sub</code>	Build a string from variables	<code>!Sub "\${AWS::StackName}-logs"</code>
<code>!Join</code>	Join a list with a delimiter	<code>!Join [":", [a, b]]</code>
<code>!FindInMap</code>	Read a value from a mapping	<code>!FindInMap [Config, !Ref AWS::Region, Size]</code>
<code>!ImportValue</code>	Import an export from another stack	<code>!ImportValue shared-vpc-id</code>
<code>!If</code>	Set a property value conditionally	<code>!If [IsProd, 3, 1]</code>
<code>!Select</code>	Select an item from a list	<code>!Select [0, !GetAZs ""]</code>
<code>!GetAZs</code>	Availability Zones in a region	<code>!GetAZs !Ref AWS::Region</code>
<code>!Base64</code>	Base64-encode text	<code>!Base64 "#!/bin/bash"</code>

Pseudo parameters

```
AWS::AccountId
AWS::Region
AWS::Partition
AWS::StackId
AWS::StackName
AWS::URLSuffix
AWS::NoValue
AWS::NotificationARNs
```

`AWS::Partition` instead of a hard-coded `aws` when templates must work across partitions.

Conditional property

```
Conditions:
  IsProd: !Equals [!Ref Environment, prod]

Resources:
  Function:
    Type: AWS::Lambda::Function
    Properties:
      ReservedConcurrentExecutions: !If
        - IsProd
        - 50
        - !Ref AWS::NoValue
```

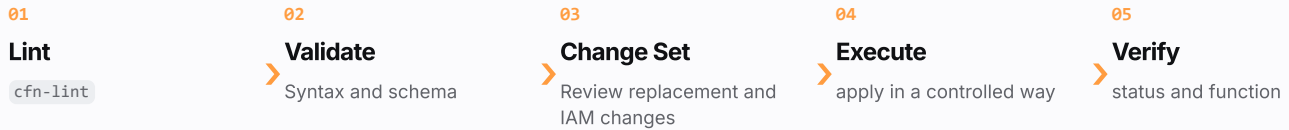
`AWS::NoValue` removes the property completely instead of setting an empty value.

Do not nest YAML short forms arbitrarily

For complex intrinsic functions, use the long form or a mapping. Readability matters more than the shortest syntax.

11 · CONTROL

Deployment with a change set and drift check



Direct development workflow

```
cfn-lint template.yaml

aws cloudformation validate-template \
  --template-body file://template.yaml

aws cloudformation deploy \
  --template-file template.yaml \
  --stack-name lab-data \
  --parameter-overrides Environment=dev \
  --capabilities CAPABILITY_NAMED_IAM \
  --no-fail-on-empty-changeset
```

In PowerShell, use a backtick for line continuation or enter the command on one line. The example uses backslashes to save space.

Explicit change set

```
aws cloudformation create-change-set \
  --stack-name lab-data \
  --change-set-name review-001 \
  --change-set-type UPDATE \
  --template-body file://template.yaml \
  --capabilities CAPABILITY_NAMED_IAM

aws cloudformation describe-change-set \
  --stack-name lab-data \
  --change-set-name review-001

aws cloudformation execute-change-set \
  --stack-name lab-data \
  --change-set-name review-001
```

For a new stack, use `--change-set-type CREATE`.

Detect drift

```
aws cloudformation detect-stack-drift \
  --stack-name lab-data

aws cloudformation describe-stack-resource-drifts \
  --stack-name lab-data
```

Find the root cause

```
aws cloudformation describe-events \
  --stack-name lab-data \
  --filters FailedEvents=true
```

Investigate specific failures before “Resource creation cancelled.”

IAM grants and secret references

CDK: grants instead of hand-written policies

```
secret = secretsmanager.Secret.from_secret_name_v2(
    self, "DbSecret", "prod/app/database"
)

fn = lambda_.Function(...)
bucket.grant_read(fn)
queue.grant_send_messages(fn)
secret.grant_read(fn)
```

Grants create targeted identity or resource policies. The secret is not read during synthesis.

CloudFormation: dynamic reference

```
MasterUserPassword:
  '{{resolve:secretsmanager:prod/app/database:
    SecretString:password}}'
```

The value remains outside the template. No output, logging or plaintext parameter.

Least-privilege check

- Which specific action is required?
- Which ARN can it be restricted to?
- Can a condition be used?
- Who may assume the role?
- Who may `iam:PassRole` use it?
- Is the KMS key policy also correct?

Prohibited patterns

- `Action: "*" and Resource: "*" without a demonstrated need`
- Access keys in the repository or in parameters
- Secrets in outputs, user data, logs or stack names
- broad trust policies

Review areas for each stack

AREA	SECURE BASELINE	REVIEW
Identity	SSO/OIDC, short-lived sessions, separate roles	<code>sts get-caller-identity</code> , CloudTrail
Data	Encryption, retention, backups and versioning	Template tests and a restore test
Network	private paths, minimal ingress rules and VPC endpoints	Security group review and reachability
Deployment	Service role, permissions boundary and approval gates	Change set and Access Analyzer
Detection	Logs, alarms, CloudTrail, Config or drift checks	Alarm test and periodic review

Secret safety

Do not use commands that retrieve secret values. For local processes, resolve dynamic Secrets Manager references with `asm-exec` at runtime so values never enter the working context.

13 · CHANGES

Replacement, import and safe refactoring

CHANGE	POSSIBLE IMPACT	SAFE APPROACH
Change a construct ID or path	New logical ID; the resource is recreated	<code>cdk diff</code> , with refactoring separated from property changes
Change an immutable property	Replacement, potentially causing data loss or downtime	Review the change set; plan migration and rollback
Remove a stateful resource	Deletion or retention depending on policy	<code>RETAIN</code> , backup and document the ownership transition
Remove a cross-stack export	Deployment is blocked while an import exists	Decouple the consumer first and deploy separately
Make a manual change in AWS	Drift; later updates may overwrite the change	Bring the emergency change back into IaC and resolve the drift

Protection layers

- **Termination Protection:** protects against stack deletion.
- **Removal Policy:** behavior when removed from the stack.
- **UpdateReplacePolicy:** behavior during replacement.
- **Stack Policy:** protects selected resources from updates.
- **Backup:** protects data independently of the stack.

Import existing resources

1. Document the current state and owner.
2. Model the resource with matching properties.
3. Do not include any other changes in the same step.
4. `cdk diff` and review the import mapping.
5. `cdk import` run it.
6. Normalize it separately afterward.

Three-deployment rule for risky changes

DEPLOY 1

Compatibility

Support the old and new structures at the same time. Provide a backup and migration path.

DEPLOY 2

Switch over

Move the consumer to the new resource or reference. Verify function.

DEPLOY 3

Clean up

Remove the old reference or resource. Handle retained data separately.

`cdk destroy` is not a cleanup script

First review the diff, protection policies, dependencies, data retention and cost of remaining retained resources.

Account and region boundaries

Management

Organizations and governance.
No workloads.

Security

Central logs, security and audit functions.

Non-Prod

Development, tests and experiments with budgets.

Prod

Strict roles, approvals, backups and alarms.

CDK for each target environment

```
targets = {
  "dev": cdk.Environment(
    account="111111111111",
    region="eu-central-1",
  ),
  "prod": cdk.Environment(
    account="222222222222",
    region="eu-central-1",
  ),
}

for stage, env in targets.items():
  AppStage(
    app,
    f"App-{stage}",
    env=env,
    stage_name=stage,
  )
```

Account IDs are configuration, not secrets. Still keep roles and regions central and verifiable.

Boundaries and mechanisms

Account	Blast radius, billing, quotas and IAM boundary
Region	Latency, data residency and service availability
Stack	Deployment and lifecycle
Stage	related stacks in one environment
StackSets	standardized rollout across accounts and regions
SCP	maximum permitted actions in Organizations

Deployment principles

Same artifact version

A tested commit moves through environments. Do not rebuild it for each environment.

Treat differences as data

Make capacity, domains and features configurable. Do not fork the architecture by copy and paste.

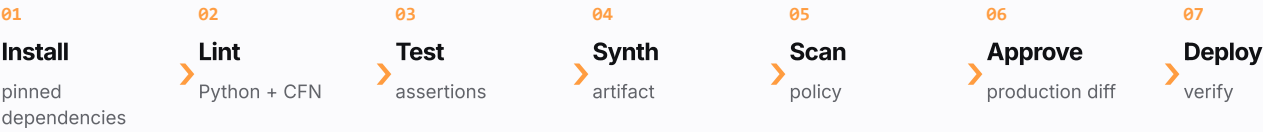
Treat production deliberately

Manual approval, maintenance window, rollback plan and observability before deployment.

Bootstrap each account and region

Every CDK target environment needs the appropriate bootstrap resources. Keep trust relationships and permissions boundaries narrow.

CI/CD gates for Infrastructure as Code



Pipeline gates

Pull request	Formatting, linting, types, unit tests, synthesis, policy and diff artifact
Merge	create an immutable artifact and record its signature and provenance
Dev	deploy automatically, integration test and smoke test
Stage	same build, realistic data flows, load tests and restore tests
Prod	diff approval, change window, deployment, alarms and verification

OIDC instead of static keys

The CI system obtains short-lived AWS credentials through a trusted identity provider.

- Restrict trust to the repository, branch and workflow.
- Use separate roles for synthesis, diff and deployment.
- `iam:PassRole` only for defined roles.
- Use the session name and tags for auditing.
- Do not use long-lived access keys as CI secrets.

Artifacts

`cdk.out`, template, asset hashes, test reports, policy results and diff in a traceable form.

Concurrency

Allow only one writer per stack. Pipeline concurrency controls or locking prevent colliding deployments.

Rollback

CloudFormation rollback is not a complete application rollback. Plan data migrations separately.

Use self-mutation deliberately

A pipeline that changes its own infrastructure needs especially narrow permissions, staging, version control and a documented recovery path.

Operational requirements after deployment

Operational Excellence

Runbooks, small reversible changes, automated verification and drift checks.

Security

Least privilege, encryption, auditing, separate accounts and secret references.

Reliability

Backups, restore tests, Multi-AZ where needed, quotas and failure modes.

Performance

Measurable requirements, appropriate services, load tests and scaling limits.

Cost Optimization

Tags, budgets, right-sizing, retention, development shutdowns and ownership.

Sustainability

Demand-based capacity, managed services, efficient regions and architectures.

Required tags

```
Project      = aws-iac-lab
Environment  = dev | stage | prod
Owner        = platform-team
CostCenter   = learning
ManagedBy    = cdk
DataClass    = internal
Criticality   = low | medium | high
```

Define tag policies centrally. Tagging matters for cost and ownership, but does not replace an IAM or account boundary.

Operational readiness

- Dashboards and alarms exist.
- Log retention and data classification are defined.
- A backup has been restored successfully.
- Service quotas match the load.
- The error budget and SLO are known.
- The runbook covers diagnosis and escalation.
- A cost alarm and budget are active.
- Ownership and the on-call path are clear.
- Drift checks run periodically.

A diff is not a cost forecast

A small template difference can have a large cost impact. Evaluate pricing, data volume, requests, logs, NAT traffic and retention separately.

Operational approval: The system is observable, recoverable, understood in cost terms and assigned to a responsible team.

17 · SERVERLESS

When AWS SAM makes sense

SAM extends CloudFormation for Lambda, API Gateway, events and related services. Its transform produces regular CloudFormation resources.

template.yaml

AWS SAM

```
Transform: AWS::Serverless-2016-10-31
Resources:
  ApiFunction:
    Type: AWS::Serverless::Function
    Properties:
      Runtime: python3.13
      Handler: app.handler
      CodeUri: src/
      Timeout: 10
    Events:
      Api:
        Type: Api
        Properties:
          Path: /health
          Method: get
```

SAM fits here

- Lambda and events form the core.
- Local builds and emulation are important.
- The team prefers declarative YAML.
- Only a few general infrastructure components are needed.

CDK fits here

- Many services are composed.
- Python abstractions and unit tests are central.
- Multiple stacks and accounts belong together.

Local workflow

```
sam validate --lint
sam build
sam local start-api
sam deploy --guided
```

Read the transform

Inspect the generated CloudFormation template, especially IAM, APIs and implicit resources.

Check the runtime

The example runtime is version-dependent. Verify the currently supported Lambda runtimes before use.

Limits of local testing

Emulation does not replace integration tests in a real AWS sandbox.

Documentation-dependent reference

SAM is presented for context in this CDK-centred guide. Commands and runtimes must be checked against the installed SAM CLI and current AWS documentation.

When Terraform makes sense

Terraform makes sense when multi-cloud, hybrid infrastructure or an existing platform standard matters more than native Python CDK integration.

The core workflow

```
terraform fmt -check
terraform init
terraform validate
terraform plan -out=tfplan
terraform apply tfplan
```

The saved plan is reviewed, and that exact plan is applied.

State is operational infrastructure

Backend	remote, encrypted and available
Locking	only one writer per state
Access	smallest possible group, auditable
Backup	versioning and recovery tested
Secrets	a sensitive marker is not a substitute for encryption

Decide in 30 seconds

QUESTION	ANSWER	STARTING POINT
Only AWS and Python expertise?	Yes	AWS CDK v2
Almost exclusively Lambda, API Gateway and events?	Yes	AWS SAM
Small existing YAML template?	Yes	CloudFormation
Multiple clouds or providers?	Yes	Terraform
Already using a standardised platform tool?	Yes	Improve the existing standard

Pin providers

Plan upgrades separately and commit the lock file.

Version modules

Keep APIs small, defaults safe and migrations documented.

Separate ownership

Terraform and CloudFormation must never manage the same resource at the same time.

18 · TROUBLESHOOTING

Common failure patterns

SYMPTOM	TYPICAL CAUSE	NEXT STEP
<code>NoCredentials</code> <code>ExpiredToken</code>	SSO session expired, wrong profile or role	<code>aws sts get-caller-identity</code> , check the profile and region
<code>AccessDenied</code>	missing action, resource, condition, trust or KMS key policy	read the exact API action and ARN; correlate with CloudTrail
<code>AlreadyExists</code>	physical name conflicts or the resource exists outside the stack	let AWS generate the name or import the resource in a controlled way
<code>ROLLBACK_COMPLETE</code>	creation failed and was rolled back	inspect failure events; deliberately prepare a new create
<code>UPDATE_ROLLBACK_FAILED</code>	rollback could not restore the resource	repair the affected resource, then continue the rollback
Export cannot be deleted	another stack imports the export	decouple and deploy the consumer first
Dependency cycle	mutual references between resources or stacks	extract the shared resource or use loose coupling
Asset publish failed	bootstrap, Docker, path or permission issue	<code>cdk doctor</code> , check the bootstrap and asset path
Unexpected replacement	logical ID or immutable property changed	stop the deployment; read the diff and specification
Stack drifted	manual change or external controller	clarify ownership; correct the IaC or actual state
S3 remains after destroy	<code>RETAIN</code> or a non-empty or versioned bucket	handle data and the resource separately and deliberately

Template level

Incorrect property, dependency, IAM definition or incompatible change in the IaC code.

Environment level

Quota, missing permission, name collision, service state or existing resource.

Application level

The deployment succeeds, but a health check, data migration or application test fails.

Do not change the template for a purely environmental problem

Classify the failure first. Otherwise, a code change can mask quotas, role errors or inconsistent state.

Diagnostic workflow for failed deployments

Diagnostic sequence

1. Confirm the account, role, region and stack.
2. Find the first specific failure event.
3. Collect every specific failure that occurred in parallel.
4. Classify the issue at template, environment or application level.
5. Design the smallest safe correction.
6. Read the diff, deploy again and verify.

read-only diagnosticsAWS CLI v2

```
aws sts get-caller-identity

aws cloudformation describe-events \
  --stack-name LabData \
  --filters FailedEvents=true

aws cloudformation describe-stack-resources \
  --stack-name LabData

aws cloudtrail lookup-events \
  --lookup-attributes \
    AttributeKey=EventName,AttributeValue=CreateBucket
```

“Resource creation cancelled” is usually an effect, not the cause

Look for an event with a specific error message. When several genuine failures exist, capture all permission or quota gaps together.

Evidence pack for a clean handover

Context Time, account, region, stack, commit and pipeline run.	Plan CDK diff or change set, expected replacements and IAM changes.	Events All specific failure events, not just the final message.
Audit Relevant CloudTrail events and invoked API actions.	Impact Affected users, data, regions and running resources.	Next step Smallest reversible correction plus verification.

CLI prerequisite

`describe-events` is the current CloudFormation API with flexible filters. Older AWS CLI versions may not support this command yet.

AWS CDK v2: daily workflow

```
# Project and version
$ cdk init app --language python
$ cdk doctor
$ cdk --version

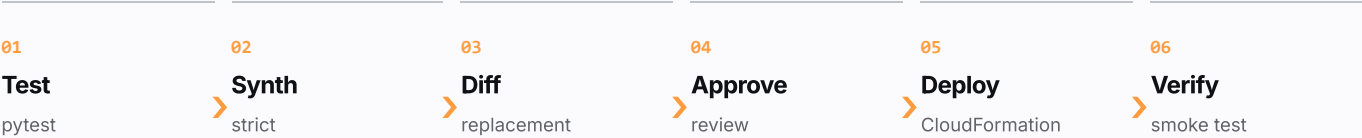
# Environment
$ cdk bootstrap aws://ACCOUNT/REGION
$ cdk bootstrap --show-template

# Explore
$ cdk list
$ cdk context
$ cdk metadata StackName
```

```
# Quality and plan
$ python -m pytest
$ cdk synth --strict
$ cdk diff StackName
$ cdk diff --security-only

# Deployment
$ cdk deploy StackName
$ cdk deploy --all
$ cdk deploy --require-approval broadening
$ cdk deploy --outputs-file outputs.json
```

Production approval sequence



Before every deployment

`aws sts get-caller-identity` run this and visibly confirm the account, role and region.

Validated baseline

For version 1.2, these commands were validated using AWS CDK CLI `2.1136.0` as the reference.

CDK commands for import, drift and recovery

```
# Fast local development
$ cdk watch StackName

# Existing resources
$ cdk import StackName
$ cdk import --resource-mapping mapping.json

# Drift
$ cdk drift StackName
$ cdk drift StackName --fail
```

```
# Diagnostics and recovery
$ cdk deploy StackName --verbose
$ cdk rollback StackName
$ cdk rollback StackName --orphan LogicalId

# Remove
$ cdk destroy StackName
```

OPTION	WHEN TO USE IT	BOUNDARY
<code>--profile NAME</code>	explicit AWS profile	still verify identity with STS
<code>--context k=v</code>	set CDK context	never use it for secrets
<code>--exclusively</code>	deploy only the selected stack	dependencies must already be compatible
<code>--concurrency N</code>	independent stacks in parallel	consider quotas and shared resources
<code>--hotswap</code>	fast local development	bypasses the regular CloudFormation path; never use in production
<code>--no-rollback</code>	targeted diagnostics	leaves partial state; not the default

An import contains only the import

When using `cdk import` do not mix updates or deletions into the same diff. Model the resource exactly first and normalise it only after a successful import.

20 · REFERENCE

CloudFormation: validation and state

```
# Optional local tools
$ cfn-lint template.yaml
$ cfn-guard validate -r rules -d template.yaml

# AWS syntax validation
$ aws cloudformation validate-template
  --template-body file:///template.yaml

# Direct development deployment
$ aws cloudformation deploy
  --template-file template.yaml
  --stack-name NAME
  --capabilities CAPABILITY_NAMED_IAM
  --no-fail-on-empty-changeset
```

```
# Inventory and status
$ aws cloudformation list-stacks
$ aws cloudformation describe-stacks
  --stack-name NAME
$ aws cloudformation describe-stack-resources
  --stack-name NAME

# Drift
$ aws cloudformation detect-stack-drift
  --stack-name NAME
$ aws cloudformation describe-stack-resource-drifts
  --stack-name NAME
```

CREATE

The stack does not exist yet. A failure can leave it in state `ROLLBACK_COMPLETE` as its final status.

UPDATE

The template and parameters are compared with the managed state.

DELETE

Policies and dependencies determine which resources remain.

Validated baseline

For version 1.2, the AWS commands were validated using AWS CLI `2.36.19` as the reference. `cfn-lint` and `cfn-guard` remain optional tools that must be installed separately.

CloudFormation: change sets and recovery

explicit change setAWS CLI v2

```
aws cloudformation create-change-set \  
  --stack-name NAME \  
  --change-set-name review-001 \  
  --change-set-type UPDATE \  
  --template-body file://template.yaml \  
  --capabilities CAPABILITY_NAMED_IAM  
  
aws cloudformation describe-change-set \  
  --stack-name NAME \  
  --change-set-name review-001  
  
aws cloudformation execute-change-set \  
  --stack-name NAME \  
  --change-set-name review-001
```

Failures and recoveryread first

```
aws cloudformation describe-events \  
  --stack-name NAME \  
  --filters FailedEvents=true  
  
aws cloudformation cancel-update-stack \  
  --stack-name NAME  
  
aws cloudformation continue-update-rollback \  
  --stack-name NAME
```

Recover only after identifying the cause

Before running `continue-update-rollback` understand the blocking resource and its actual state.

Capabilities

<code>CAPABILITY_IAM</code>	IAM resources with generated names
<code>CAPABILITY_NAMED_IAM</code>	IAM resources with explicit names
<code>CAPABILITY_AUTO_EXPAND</code>	macros may expand the template before deployment

Review

Review replacements, IAM, networking, data storage and tags.

Execute

Execute only the change set that was reviewed.

Verify

Check stack status, the application, alarms and drift.

21 · PRACTICE

Four-week practice plan

WEEK 1

Stack foundations

- Build:** an encrypted, versioned S3 bucket with a KMS key, tags and outputs.
- Read the CloudFormation template after synthesis
 - Write template assertions
 - Be able to explain the diff
 - Verify deployment and retention

WEEK 3

Multiple stacks

- Build:** data, application and operations stacks in development and production configurations.
- Draw the dependencies
 - Separate stateful and stateless resources
 - Practise removing a cross-stack reference
 - Detect and correct drift

WEEK 2

Event-driven application

- Build:** S3 → SQS → Lambda with a DLQ, alarm and minimal grants.
- Explain the timeout and visibility timeout
 - Create a failure scenario
 - Check the DLQ and alarm
 - Inspect logs without exposing secrets

WEEK 4

Pipeline and recovery

- Build:** pull-request checks, a sandbox deployment, approval and a smoke test.
- Constrain the OIDC role
 - Intentionally break a policy gate
 - Identify a replacement in the diff
 - Demonstrate backup and restore

Questions for the technical review

Problem What specific problem does the stack solve, and for whom?	Boundary What is this version explicitly unable to do?	Evidence Which tests and metrics show that it works?
Risk Which change could cause data loss, downtime or high costs?	Recovery How is the system restored from a backup or previous state?	Operations Who sees failures, and who responds to them?

Capstone project

Document the architecture, IaC code, tests, threats, cost assumptions, deployment, rollback, restore process and known limitations in a public or private repository.

Compact glossary

ARN	global AWS resource identifier	Idempotency	repeated execution leads to the same target state
Asset	file, code bundle or container image used in a deployment	Logical ID	identity of a resource in the template
Blast Radius	maximum impact of a failure	Physical ID	actual name or identifier in AWS
Bootstrap	CDK support resources for each account and region	Replacement	an old resource is replaced by a new one
Change Set	CloudFormation preview of planned changes	Rollback	return to the last stable stack state
Construct	CDK building block made from one or more resources	Stateful	resource contains data worth retaining
Context	CDK values used for synthesis and lookups	Synthesis	CDK generates CloudFormation and assets
Drift	difference between expected and actual state	Token	deployment value that has not yet been resolved

Working rules

- One IaC owner per resource.
- No production deployment without a reviewed diff.
- Backups count only after a successful restore.
- Reference secrets; do not read them.
- Protect stateful resources separately.
- Manual changes are drift, not documentation.

Release check

- Account, role and region confirmed
- Tests and policy gates successful
- Replacement and IAM reviewed in the diff
- Rollback and restore understood
- Smoke test and alarms verified
- Drift and cost impact assessed

Understand systems. Reduce complexity. Build things that last.

Primary sources and release validation

AWS CDK v2 Developer Guide

docs.aws.amazon.com/cdk/v2/guide/home.html

AWS CDK CLI Reference

docs.aws.amazon.com/cdk/v2/guide/ref-cli-cmd.html

AWS CDK Python API Reference

docs.aws.amazon.com/cdk/api/v2/python/

AWS CloudFormation User Guide

docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html

CloudFormation Best Practices

docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html

CloudFormation CLI: describe-events

docs.aws.amazon.com/cli/latest/reference/cloudformation/describe-events.html

Intrinsic Function Reference

docs.aws.amazon.com/AWSCloudFormation/latest/TemplateReference/intrinsic-function-reference.html

Dynamic References

docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/dynamic-references.html

CloudFormation Guard

docs.aws.amazon.com/cfn-guard/latest/ug/what-is-guard.html

AWS SAM Developer Guide

docs.aws.amazon.com/serverless-application-model/latest/developerguide/what-is-sam.html

Choosing an IaC Tool

docs.aws.amazon.com/prescriptive-guidance/latest/choose-iac-tool/choose-tool.html

AWS Well-Architected Framework

docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html

Release validation

AWS CLI `2.36.19` and pinned AWS CDK CLI `2.1136.0`. The generated `validation-report.md` documents every validated command and expected option.

Version and responsibility

Version 1.2 was created on 15 August 2026. Before making production changes, review the current resource specification, API reference, permissions, quotas and generated change set.

Portable build

Relative font assets, the Inter licence, Python build, overflow check, PDF page check and release ZIP are part of the source package.