AWS Serverless: SAM – Serverless Applica…

1. AWS Serverless Application Reposity (AWS SAR) allow you to **share** your **Serverless applications packages** using SAM with other AWS accounts
2. SAM CLI + AWS Toolkits allows you to **debug your Lambda functions locally, inspect [检查] variables**, and **excute code line-by-line**

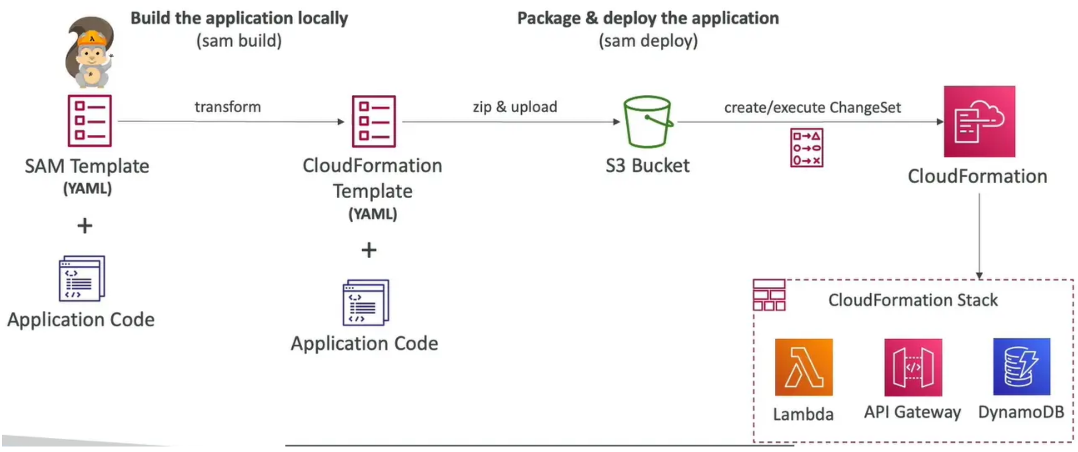
AWS SAM

- SAM = Serverless Application Model
- Framework for developing and deploying serverless applications
- all the configuraration is YAML code
- Generate complex CloudFormation from simple SAM YAML file
- Supports anything from CloudFormaiton : outputs,mappings, parameters,resources..
- SAM can use CodeDeploy to deploy Lambda functions
- SAM can help you to run Lambda, API Gateway,DynamoDB locally

Recipe

- transform header indicates [表明] it's SAM template:
  - Transform : 'AWS :: Serverless - 2016-10-31'
- Write Code
  - AWS :: Serverless :: Function
  - AWS :: Serverless :: Api
  - AWS :: Serverless :: SimpleTable
- Package & Deploy: sam deploy (optionally preceded by "sam package")
- quickly sync local changes to AWS Lambda (SAM Accelerate [加速]) : sam sync --watch

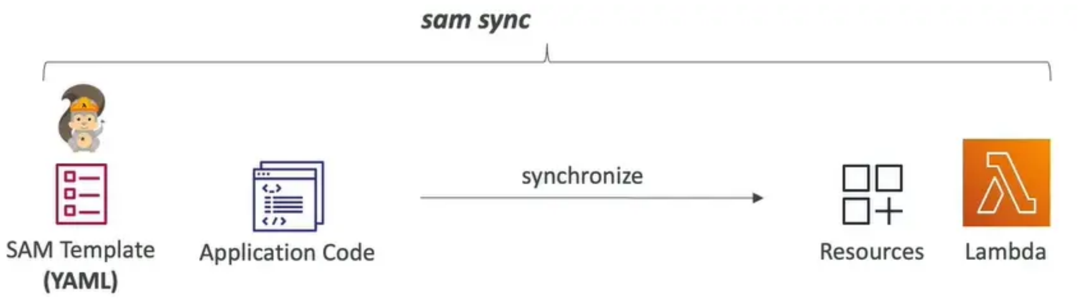
Deep Dive into [深入探讨] SAM Deployment



SAM Accelerate (sam sync)

- SAM Accelerate is a set of features to reduce latency while deploying resources to AWS

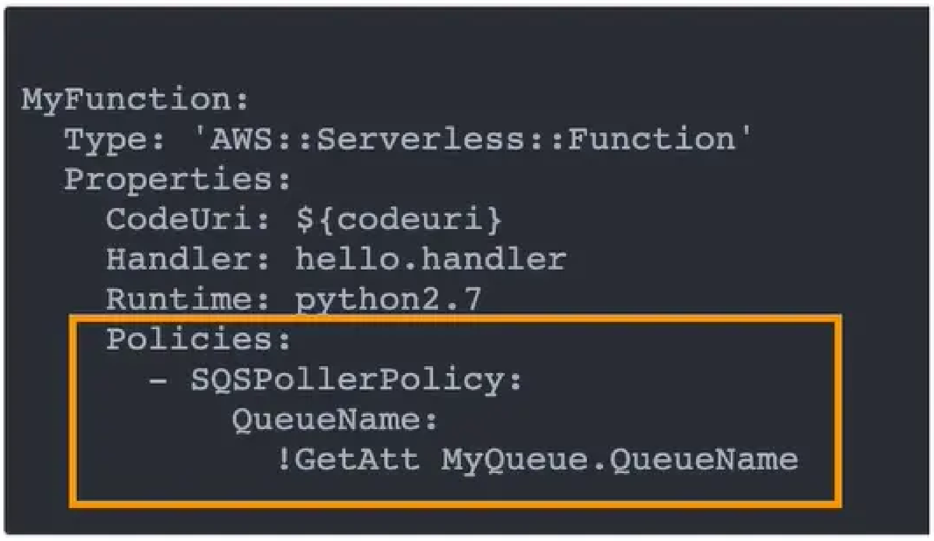
- sam sync
  - synchronizes your project declared [声明] in SAM templates to AWS
  - synchronizes code changes to AWS without updating infrastructure (uses servcie APIs & bypass CloudFormation)



SAM Accelerate

- sam sync (no optionss)
  - synchronize code and infrastructure
- sam sync --code
  - synchronize code changes without updating infrastructure (bypass Cloudfromation, upate in seconds)
- sam sync --code --resource AWS::Serverless::Function
  - synchronize only all lambda functions and their dependencies
- sam sync --code --resource-id HelloWorldLambdaFunction
  - synchronize only a specific resource by its ID
- sam sync --watch
  - monitor for file changes and automatically synchronize when changes are detected
  - if changes include configuration, it uses sam sync
  - if changes are code only, it uses sam sync --code

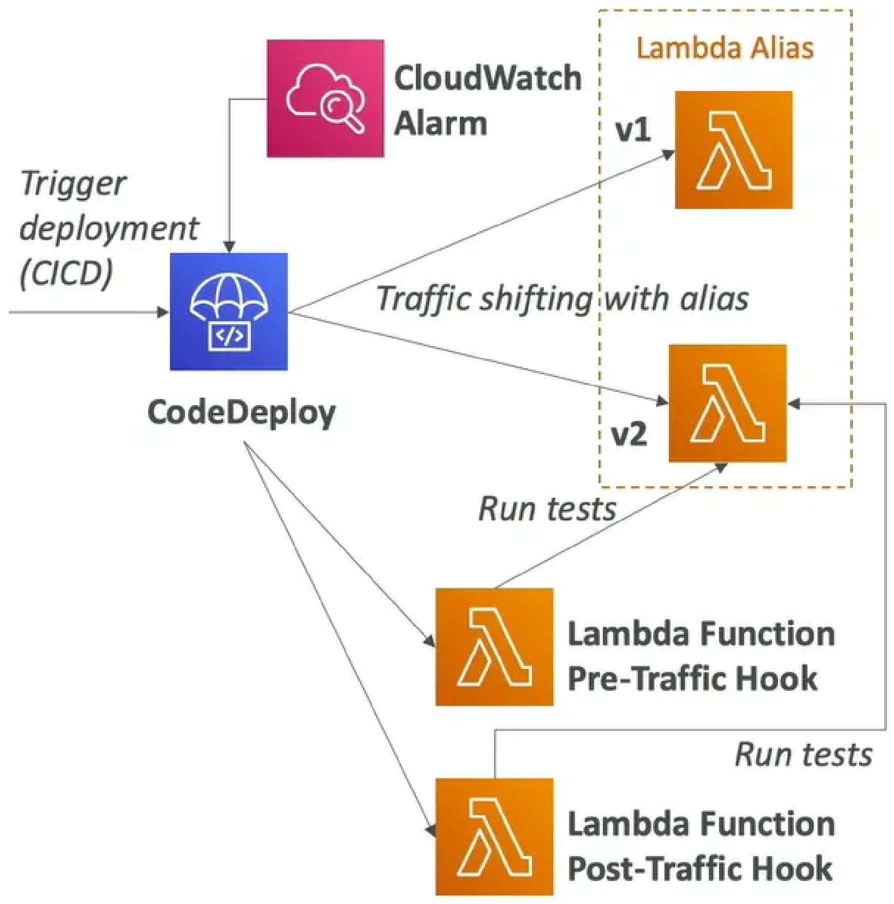
SAM Policy Template

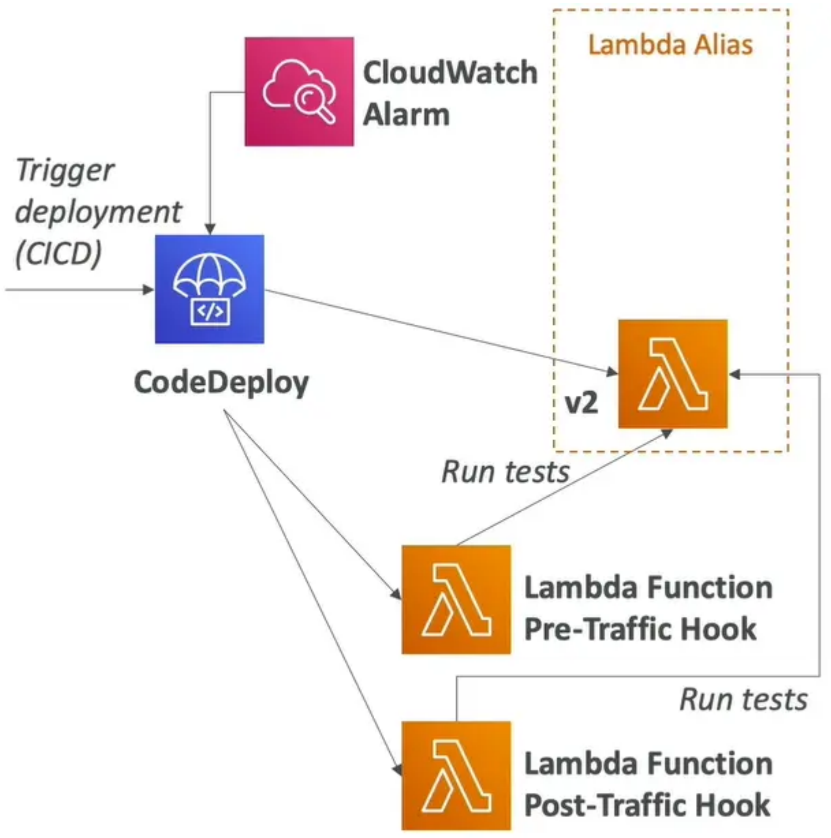


```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
    Policies:
      - SQSPollerPolicy:
          QueueName:
            !GetAtt MyQueue.QueueName
```

- list of templates to apply permissions to your lambda functions
- important examples:
  - S3ReadPolicy: gives read only permissions to objects in S3
  - SQSPollerPolicy: allows to poll an SQS queue
  - DynamoDBCrudPolicy : CRUD = create read update delete

SAM and CodeDeploy


CloudWatch Alarm
Lambda Alias
v1
Trigger deployment (CICD)
Traffic shifting with alias
CodeDeploy
v2
Run tests
Lambda Function Pre-Traffic Hook
Run tests
Lambda Function Post-Traffic Hook


CloudWatch Alarm
Lambda Alias
Trigger deployment (CICD)
CodeDeploy
v2
Run tests
Lambda Function Pre-Traffic Hook
Run tests
Lambda Function Post-Traffic Hook

- SAM framwork natively uses CodeDeploy to update Lambda functions
- traffic shifting feature

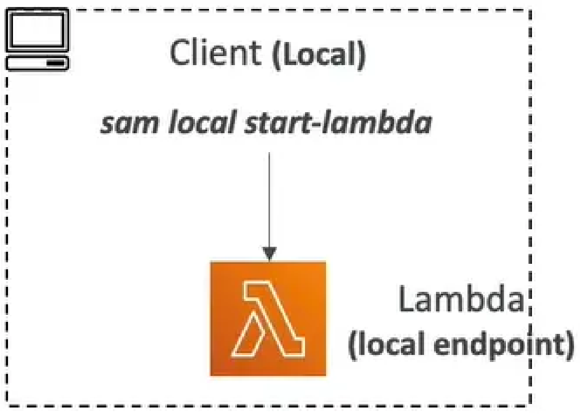
- pre and post traffic hooks features to validate deployment (before the traffic shift starts and after is ends)
- easy & automated rollback using CloudWatch Alarms

```
Resources:
MyLambdaFunction:
  Type: AWS::Serverless::Function
  Properties:
    Handler: index.handler
    Runtime: nodejs12.x
    CodeUri: s3://bucket/code.zip

    AutoPublishAlias: live

    DeploymentPreference:
      Type: Canary10Percent10Minutes
      Alarms:
        # A list of alarms that you want to monitor
        - !Ref AliasErrorMetricGreaterThanZeroAlarm
        - !Ref LatestVersionErrorMetricGreaterThanZeroAlarm
      Hooks:
        # Validation Lambda functions that are run before & after traffic shifting
        PreTraffic: !Ref PreTrafficLambdaFunction
        PostTraffic: !Ref PostTrafficLambdaFunction
```

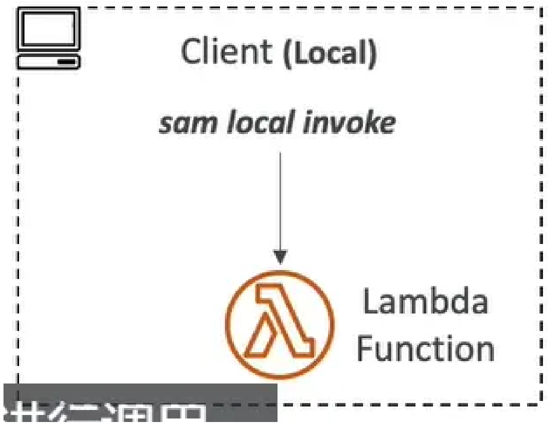
- AutoPublishAlias
  - detects when new code is being deployed
  - creates and publishes an updated version of that function with the latest code
  - points the alias to the updated version of the lambda function
- DeploymentPreference
  - Canary,Linear,AllAtOnce
- Alarms
  - alarms that can trigger a rollback
- Hooks
  - pre and post traffic shifting lambda funcitons to test your deployment

Local Capabilities [能力]

- locally start AWS Lambda

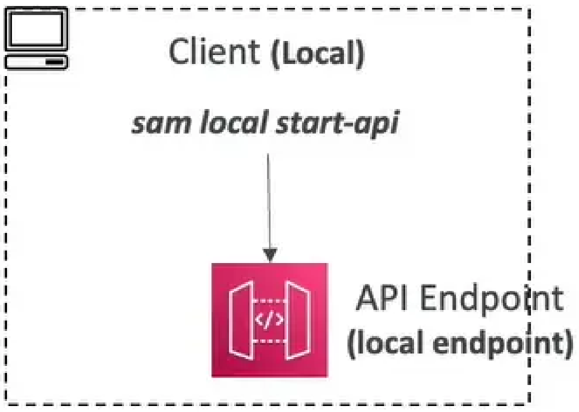


  - sam local start–lambda
  - starts a local endpoint that emulates [仿真] AWS Lambdas
  - can run automated tests against this local endpoint
- locally Invoke Lambda Function



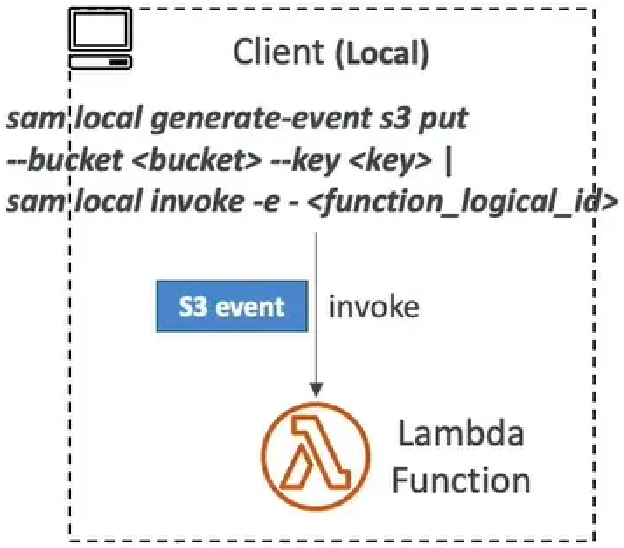
- sam local invoke
- invoke Lambda function with payload once and quit after invocation completes
- helpful for generating test cases
- if the function make API calls to AWS, make sure you are using the correct –– profile option

- Locally Start an API Gateway Endpoint



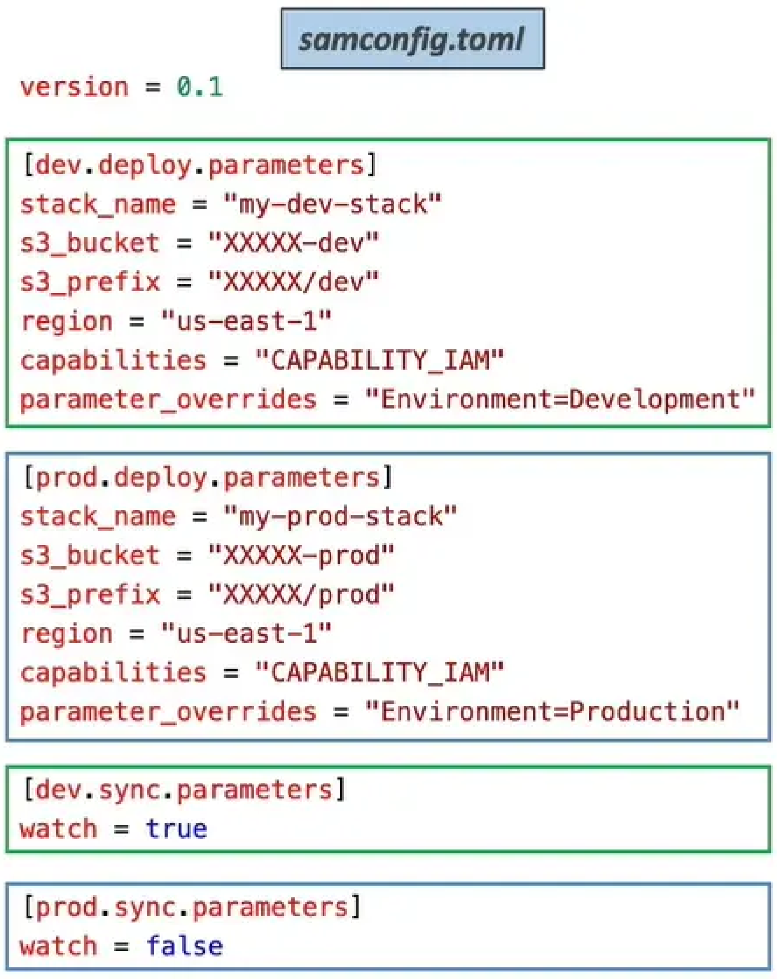
- sam local start–api
- starts a local HTTP server that hosts all you functions
- changes to functions are automatically reloaded

- Generate AWS Events for Lambda Functions


Client (Local)
sam local generate-event s3 put --bucket <bucket> --key <key> | sam local invoke -e - <function_logical_id>
S3 event
invoke
Lambda Function

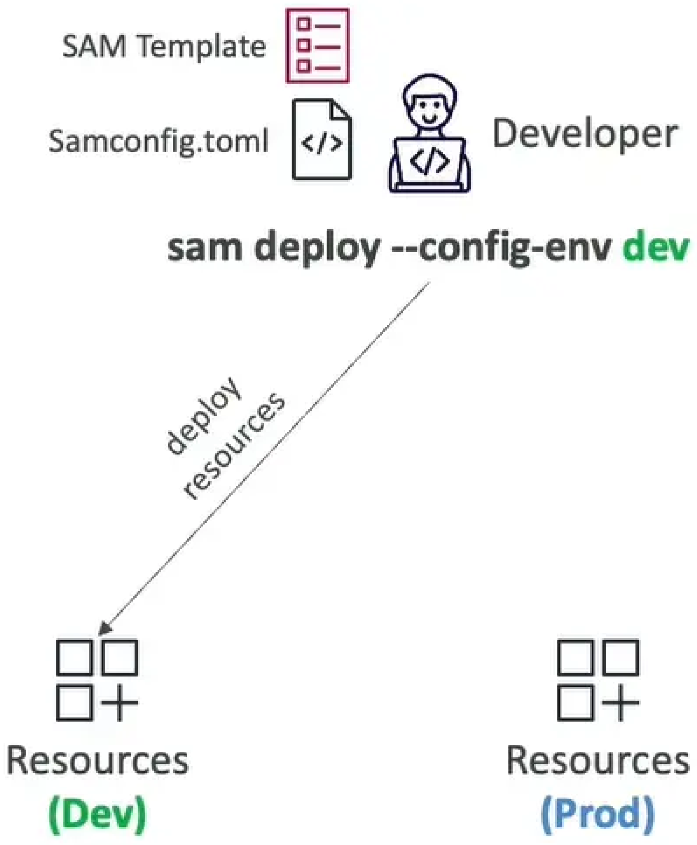
- sam local generate–event
- Generate sample payloads for event sources
- S3, API Gateway,SNS, Kinesis, DynamoDB

Multiple Environments



samconfig.toml

```
version = 0.1

[dev.deploy.parameters]
stack_name = "my-dev-stack"
s3_bucket = "XXXXX-dev"
s3_prefix = "XXXXX/dev"
region = "us-east-1"
capabilities = "CAPABILITY_IAM"
parameter_overrides = "Environment=Development"

[prod.deploy.parameters]
stack_name = "my-prod-stack"
s3_bucket = "XXXXX-prod"
s3_prefix = "XXXXX/prod"
region = "us-east-1"
capabilities = "CAPABILITY_IAM"
parameter_overrides = "Environment=Production"

[dev.sync.parameters]
watch = true

[prod.sync.parameters]
watch = false
```


SAM Template
Samconfig.toml
Developer
sam deploy --config-env dev
deploy resources
Resources (Dev)
Resources (Prod)