

API CloudFormation and CDK

CS516 – Cloud Computing

Computer Science Department

Maharishi International University

Maharishi International University - Fairfield, Iowa



All rights reserved. No part of this slide presentation may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying or recording, or by any information storage and retrieval system, without permission in writing from Maharishi International University.

Content

- CloudFormation
 - Intrinsic functions
 - Pseudo parameters
- CDK
- AWS Route 53
 - Hosted zones
 - NS , SOA, Alias, CNAME records
 - Routing policies
 - AWS Certificate Manager
- CloudWatch insights
- Well Architected

CloudFormation

AWS CloudFormation is an AWS service that uses template files (JSON, YAML) to automate deployment of AWS resources. Also known as Infrastructure-as-Code (IaC).

- Deployment speed.
- Consistency – you can apply precisely the same configuration repeatedly. In this way, CloudFormation ensures that your applications and services will be consistent and identical, no matter how many instances you create.
- Reduced human errors.
- Easy updates.
- Auditing and change management – Track how resources changed over time.

<https://docs.aws.amazon.com/codebuild/latest/userguide/cloudformation-vpc-template.html>

https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/CHAP_TemplateQuickRef.html

Template sections

Parameters: Values to pass to your template at runtime (when you create or update a stack). You can refer to parameters from the Resources and Outputs sections of the template.

Mappings: A mapping of keys and associated values. Use the `Fn::FindInMap` intrinsic function in the Resources and Outputs sections.

Conditions: Conditions that control whether certain resources are created or whether certain resource properties are assigned a value during stack creation or update. For example, you could conditionally create a resource that depends on whether the stack is for a production or test environment.

Resources: Stack resource and their properties.

Outputs: Exporting the resources then other resources can refer it.

Intrinsic functions

- Condition – Create resources or set properties based on condition.
- FindInMap – Finds value from 2-level map.
- GetAtt
- ImportValue and Export in the output
- Join
- Split and Select
- Sub
- Ref

Learn more: [Intrinsic function reference](#)

Pseudo parameters reference

You can read these values in your template directly:

- AccountId
- Region
- StackName
- NoValue – Removes the corresponding resource property when specified as a return value in the Fn::If intrinsic function.

Read more: <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/pseudo-parameter-reference.html>

CDK (Cloud Development Kit)

More flexible than CloudFormation templates. You can use programming languages to create resources in the cloud.

Code is synthesized into CloudFormation template.

Learn more: [Your first CDK app](#) and [Creating your first CDK project](#)

To get started:

- Configure AWS CLI. AWS CDK uses credentials in AWS CLI to create resources.
- Install NodeJS, AWS CDK, TypeScript.
- Bootstrap your account that creates assets that CDK requires when creating resources. `cdk bootstrap aws://ACCOUNT-NUMBER/REGION`
- `cdk init app --language typescript`
- `npm run build || cdk synth || cdk deploy`

Route 53

This is a highly available and scalable cloud Domain Name System (DNS) web service.

It is designed to give developers and businesses an extremely reliable and cost-effective way to route end users to applications by translating names, like `www.example.com`, into the numeric IP addresses, like `98.10.12.31`.

An AWS service that allows management of website domains and DNS records.

Hosted zones

it represents a collection of records that can be managed together, belonging to a single parent domain name. All resource record sets within a hosted zone must have the hosted zone's domain name as a suffix.

For example, the amazon.com hosted zone may contain records named **www.amazon.com**, and **www.aws.amazon.com**, but not a record named www.amazon.ca.

One hosted zone costs \$0.50 per month.

NS records

NS records point to the servers that help to translate domain names into the IP addresses that computers use to communicate with one another.

NS records are automatically created when you create a new hosted zone. Provide that to the domain name provider i.e., GoDaddy. Then you will have full control on your domain name in AWS and create the required records.

Record types

- **Alias** - A type of record that you can create with Amazon Route 53 to route traffic to AWS resources such as ALB, Amazon CloudFront distributions and Amazon S3 buckets.
- **CNAME** - It maps one domain name to another domain name. For example, RDS, ElastiCache.
- **SOA** - The record is created with hosted zone along with NS records. The SOA record stores important information about a domain when the domain was last updated, and how long the server should wait between refreshes.

Read more: [Supported record types in AWS](#)

Sub domain

A domain name that has one or more labels prepended to the registered domain name.

For example, The **example.com** domain can have sub domains:

- *accounting*.**example.com**
- *hr*.**example.com**
- *it*.**example.com** so on.

You can create a hosted zone for the sub domain and create, manage its sub domains of the subdomain. For example, **it.example.com**

- *team1*.**it.example.com**
- *team2*.**it.example.com**

Time To Live (TTL)

The amount of time, in seconds, that you want a DNS resolver to cache (store) the values for a record before submitting another request to Route 53 to get the current (new) values for that record.

If the DNS resolver receives another request for the same domain before the TTL expires, the resolver **returns the cached value**.

A longer TTL reduces your Route 53 charges, which are based in part on the number of DNS queries that Route 53 responds to.

Routing policies

- **Simple routing policy** – Route internet traffic to a single resource for your domain.
- **Geolocation routing policy** – Use when you want to route internet traffic to your resources based on the location of your users.
- **Latency routing policy** – Use when you have resources in multiple locations and you want to route traffic to the resource that provides the best latency.
- **Failover routing policy** – Failover routing lets you route traffic to a resource when the resource is healthy or to a different resource when the first resource is unhealthy.

Read more: [AWS Route 53 routing policies](#)

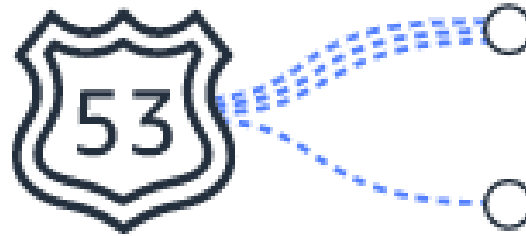
☒ Simple routing

Use if you want all of your clients to receive the same response(s).



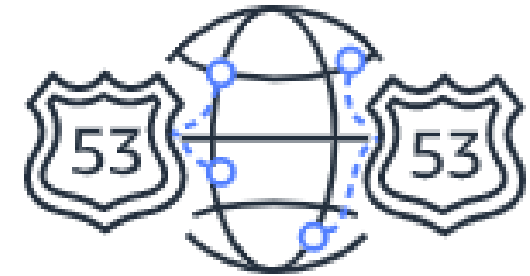
☐ Weighted

Use when you have multiple resources that do the same job, and you want to specify the proportion of traffic that goes to each resource. For example: two or more EC2 instances.



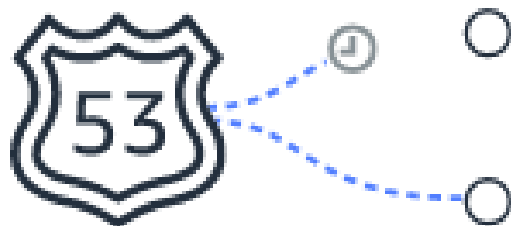
☐ Geolocation

Use when you want to route traffic based on the location of your users.



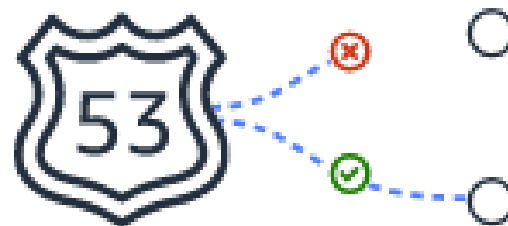
☐ Latency

Use when you have resources in multiple AWS Regions and you want to route traffic to the Region that provides the best latency.



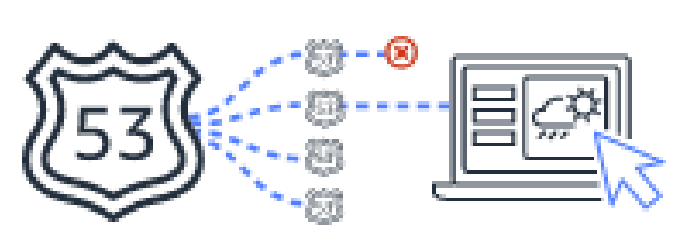
☐ Failover

Use to route traffic to a resource when the resource is healthy, or to a different resource when the first resource is unhealthy.



☐ Multivalue answer

Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.



Amazon Certificate Manager (ACM)

With ACM, you can

- Create and renew SSL/TLS X.509 certificates for free
- Import third-party certificates into the ACM and use it in your AWS resources by referring its ARN (Amazon Resource Name).

ACM certificates can secure wildcard domains. ACM wildcard certificates can protect an unlimited number of subdomains.

ACM generates a CNAME record that you need to add in the corresponding hosted zone.

CloudWatch insights

CloudWatch Logs Insights enables you to interactively search and analyze your log data in Amazon CloudWatch Logs. You can perform queries to help you more efficiently and effectively respond to operational issues. If an issue occurs, you can use CloudWatch Logs Insights to identify potential causes and validate deployed fixes.

Learn more: [Analyzing log data with CloudWatch Logs Insights](#)

General queries

Find the 25 most recently added log events.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
```

Get a list of the number of exceptions per hour.

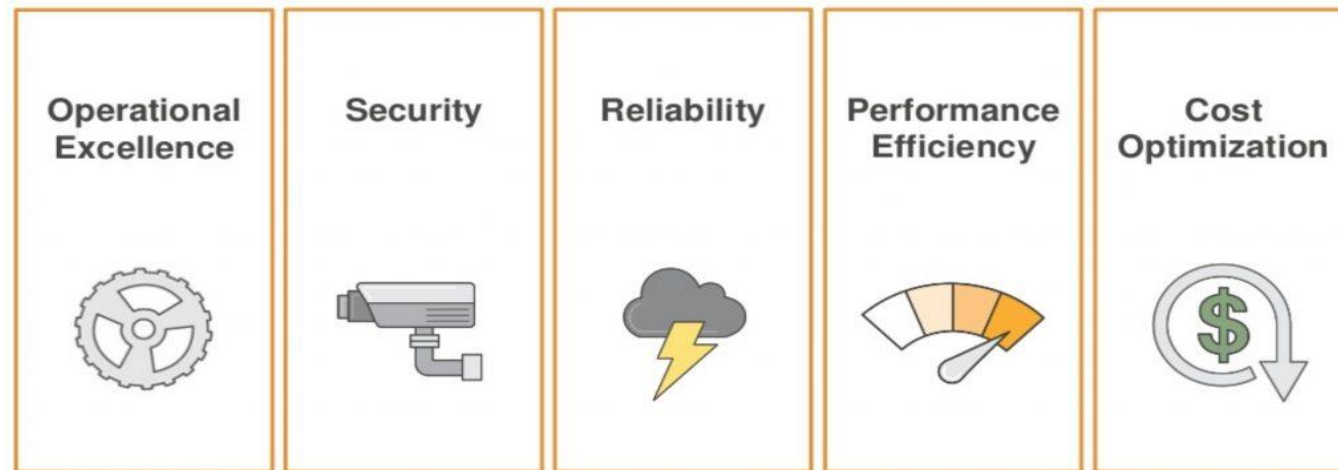
```
filter @message like /Exception/  
  | stats count(*) as exceptionCount by bin(1h)  
  | sort exceptionCount desc
```

Get a list of log events that aren't exceptions.

```
fields @message | filter @message not like /Exception/
```

AWS Well Architected – 6 pillars

AWS Well-Architected helps cloud architects build secure, high-performing, resilient, and efficient infrastructure for their applications and workloads. The sustainability pillar is recently added. That focuses on environmental impacts, especially energy consumption and efficiency.



Read more: AWS academy course and [AWS Well Architected](#)

Operational Excellence pillar - Deliver business value

- Focus - Run and monitor systems to deliver business value, and to continually improve supporting processes and procedures.
- Key topics
 - Automating changes
 - Responding to events
 - Defining standards to manage daily operations
- Principles
 - Perform operations as code
 - Make frequent, small, reversible changes
 - Refine operations procedures frequently
 - Anticipate failure
 - Learn from all operational events and failures

Security pillar - Protect and monitor systems

- Focus - Protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies.
- Key topics
 - Protecting confidentiality and integrity of data
 - Identifying and managing who can do what
 - Protecting systems
 - Establishing controls to detect security events
- Principles
 - Implement a strong identity foundation
 - Enable traceability
 - Apply security at all layers
 - Automate security best practices
 - Protect data in transit and at rest
 - Keep people away from data
 - Prepare for security events

Reliability pillar - Recover from failure and mitigate disruption.

- Focus - Ensure a workload performs its intended function correctly and consistently when it's expected to.
- Key topics
 - Designing distributed systems
 - Recovery planning
 - Handling change
- Principles
 - Automatically recover from failure
 - Test recovery procedures
 - Scale horizontally to increase aggregate workload availability
 - Stop guessing capacity
 - Manage change in automation

Performance Efficiency pillar - Use resources sparingly.

- Focus - Use IT and computing resources efficiently to meet system requirements and to maintain that efficiency as demand changes and technologies evolve.
- Key topics
 - Selecting the right resource types and sizes based on workload requirements
 - Monitoring performance
 - Making informed decisions to maintain efficiency as business needs evolve
- Principles
 - Democratize advanced technologies
 - Go global in minutes
 - Use serverless architectures
 - Experiment more often
 - Consider mechanical sympathy

Cost Optimization pillar - Eliminate unneeded expense.

- Focus - Avoid unnecessary costs.
- Key topics
 - Understanding and controlling where money is being spent
 - Selecting the most appropriate and right number of resource types
 - Analyzing spend over time
 - Scaling to meeting business needs without overspending
- Principles
 - Implement Cloud Financial Management
 - Adopt a consumption model
 - Measure overall efficiency
 - Stop spending money on undifferentiated heavy lifting
 - Analyze and attribute expenditure