# INFRASTRUCTURE AS CODE TOOLS COMPARISON ON AWS CLOUD ENVIRONMENT

Songwut Cotcharat[1]

Dr. Chaiyaporn Khemapatapan[2]

## ABSTRACT

In the agile world, characterized by an increased demand for speed and IT projects, teams are increasingly reliant on task automation. Consequently, Infrastructure engineers face the challenge of effectively managing their workloads. When it comes to cloud-based infrastructure, Infrastructure as Code (IaC) serves as an effective means of automating manual tasks, offering benefits such as scalability, speed, and transparency. Given its recent emergence, there are numerous IaC tools available for selection. The most commonly tools used in the industry are AWS CloudFormation and Terraform. This study aims to compare these tools and determine which one is more suitable. The research methodology involved conducting a survey to build a three-tier application infrastructure on AWS using IaC, while simultaneously examining and comparing the results obtained from these tools. The findings indicate that, in terms of superiority, there is no significant disparity between Terraform and CloudFormation; however, both tools may require substantial investments of time and resources due to their inherent complexity. The choice between these tools also depends on the specific requirements and preferences associated with building applications or infrastructures. Moreover, it is worth noting that the IaC community is rapidly expanding, and comprehensive support for advanced use cases can be easily found in official documentation. In summary, effectively evaluating and comparing these tools proves to be a challenging task. Nevertheless, this research provides valuable insights into their functionalities and how they can be effectively employed within various environments.

**Keyword:** DevOps, Cloud Engineering, Infrastructure as Code, Terraform, AWS CloudFormation, and IaC

## Introduction

[1]Master Student of College of Innovative Technology and Engineering, Dhurakij Pundit University

[2]A Research Supervisor

In the modern day, the ability to provision, configuration, and deployment of the infrastructure and application are very challenging and very competitive. The engineer must have a various knowledge in almost every technology field. The old ways to do this is that everything needs to be done manually and individually. The problem starts with on how you plan to buy a hardware, how you can configure all the essential tools to be compile with your application, and how you deploy your applications. This process could take months to complete before you can start a real work on that project. Not to include how difficult you have to maintenance and monitoring your own infrastructure. A lot of engineering time will spend on those topics. The Organizations that can provision, configuration their infrastructure, and deploy the application early and with high frequency will have the ability to compete in the market. A new approach called DevOps and Infrastructure as a code (IaC) promise to allow the organization to reach these goals. Many organizations seem interested in this new approach of organizing development and operations, as shown by the number of publications dealing with DevOps in popular press. DevOps has also become a topic of active scientific research (Campbell, 2020), as demonstrated by the increasing number of scientfic papers published on the topic. Infrastructure as a code (IaC) is the practice to automatically configured system dependencies and to provision local and remote instances. Practitioners consider IaC as a fundamental pillar to implement DevOps practices, which helps them to rapidly deliver software and services to end-users.

## Purpose/Objective

1.To measure the efficiency and the effectiveness of two IaC tools to use on AWS (Terraform and AWS CloudFormation).

2. To help individual or organization decide which tools they should select for their works.

3. To bring introduction to the new technology for personal or business use.

4. To prove that the traditional ways of how-to setup infrastructure and application should not be the best anymore.

## Scope of Research

1. Design a 3-tier web application that needs to be built on Amazon Web Service (AWS).

2. Write a terraform and AWS cloud formation script to build a same infrastructure on Amazon Web Service (AWS).

3. Start build the infrastructure on the centralized personal host machine.

4. Analyze the result.

**Expected Benefit**

1. Introduce an infrastructure as code tool and concept to wider audience.

2. Can help business in term of cost optimization for using infrastructure as code instead of the traditional way of provisioning.

3. Can help business improve daily productivity for an engineer by using more faster and reliable infrastructure as code tools

## Related Concept and Theory

1. Cloud Computing, Cloud computing is a relatively recent and currently very high-profile method of IT Deployment. Compute facilities such as virtualised server hosting or remote storage accessed via an API, are provided by a cloud provider to a cloud client. Often a client is a third party company, who will use the compute facilities to provide an external service to their users. For example, Amazon is a cloud provider supplying a storage API to Dropbox, who use the API to provide a file synchronisation service to domestic and commercial users. In many ways, cloud computing resembles a move away from conventional desktop computing that has been the hallmark of the previous decade (Labouardy, 2021), towards a centralised computing model. One stark difference to the mainframes of the timesharing past is that the new paradigm is supported by vast datacentres containing tens of thousands of servers working in unison. These datacentres are described as 'Warehouse-Sized Computers" (Artac, 2017), reflecting their view that the machines can collectively be regarded as a single entity The coordination of these machines is supported by specialist software layers, including virtualisation technologies.

2. DevOps, The word "DevOps" is a mashup of "development' and "operations" (Alexander, 2020) but it represents a set of ideas and practices much larger than those two terms alone, or together. DevOps includes security, collaborative ways of working, data analytics, and many other things. DevOps describes approaches to speeding up the processes by which an idea goes from development to deployment in a production environment where it can provide value to the user. These approaches require that development teams and operations teams communicate frequently and approach their work with empathy for their teammates.

Scalability and flexible provisioning are also necessary. With DevOps, those that need power the most, get it through self service and automation (Contino, 2022). Developers, usually coding in a standard development environment, work closely with IT operations to speed software builds, tests, and releases without sacrificing reliability.

3. Infrastructure as Code (IaC), Infrastructure as Code uses DevOps methodology and versioning with a descriptive model to define and deploy infrastructure, such as networks, virtual machines, load balancers, and connection topologies (Brikman, 2019). Just as the same source code always generates the same binary, an IaC model generates the same environment every time it deploys.

## Literature Review

Murphy (2022), did a research an adaptation of how the real world of infrastructure as a code. The conclusion was manual processes require more time and are error prone, but IaC provides developers with the ability to automate creation of infrastructure and support Cloud Data Ecosystems in an automated manner. Adoption of IaC has a potential to lower companies' expenses while improving time effieciency, consistency, and transparency of their cloud infrastructure. As mentioned by Gartner and HashiCorp, there is clearly a lack of professionals that are competent enough to automate more complex tasks and environments. Without these professionals, successful implementation of IaC is greatly reduced and the potential benefits to industry would be negligible. To get more IaC competent and fluent developers, industry wide adoption will be necessary. A big part of adoption will include successfully learning IaC and opening discussion for its development. From both the literature review and the results of the survey the best strategies for learning IaC are to partition the subject into smaller, easily absorbed pieces and convey full teams to follow said practices. However, this learning will require time and repetition, along with commitment from management, customers and developers in companies.

Rahman (2021), at 41st International Conference on Software Engineering completed a systematic mapping study of IaC and concluded that the trend of Infrastructure as Code is growing so fast. However, it is still work in progress steps. Meaning that a lot of things can be improve in term of the provider module and community support. The researcher also mentioned that the direction of the research should emphasize a lack of defect analysis, error pattern, security, and knowledge and training. While IaC is consider a one part of DevOps methodology, but the researcher disagreed that DevOps is just a tool that everyone can learn and practice and should not be consider it as a tool that should be practice to only the one that use DevOps methodology. As the researcher sees DevOps is more of a culture that a tool or technologies. As various tools are emerging quickly in IaC, the researcher recommend the company to invest in training for only specific IaC tool that are widely use because of the numbers of knowledge based and community support online are much bigger compared to the unpopular one.

## Research Methodology

### System Development and Implementation

The process will start by using IaC tools building AWS infrastructure 3 tier web application with high availability and high scalability. The 3-tier web application will include VPC, Internet Gateway, Security group, Application Load Balancer, Private and Public Subnets, EC2 with Auto scaling group, and Database with high availability.

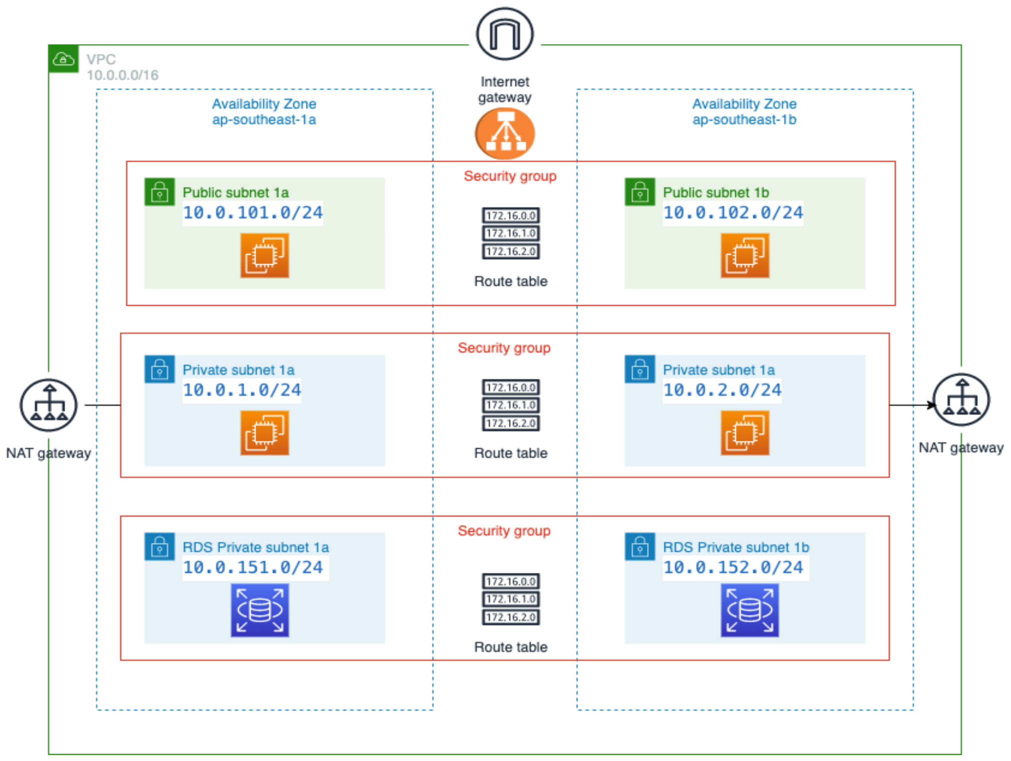

**Figure 1** Overall System Design

<u>Terraform Implementation</u>

in this step terraform script will be created to build a 3tier web application which consist with all the infrastructure module script in it with the following steps.

1. Prerequisites, to have terraform to communicate to AWS. The must do steps are having AWS account, AWS CLI Installed, and AWS credential configured locally.

2. Create Terraform VPC Module file, this script will be accommodating the creation of VPC environment on cloud that will include VPC itself, Public and Private Subnet, Nat Gateway, Internet Gateway, and routing destination for each subnet.

3. Create Terraform EC2 Module file, this script will create two EC2 instances, one in each availability zone. They will host the web server and both machines will have an access in and out of the internet through internet gateway.

4. Create Terraform Load Balancer Module file, this script Load balancer Module will be used for directing the traffic between two EC2 public instance in two availability zones.

5. Create Terraform RDS Module file, The Amazon RDS id the database service on AWS. In this research we will be using RDS with MySQL database.

6. Create Terraform Security Group Module file, this module will create the security group for EC2, Load Balancer and RDS. The security group are group of the rules that will be determine on who which IP address can be access to each resource on the environment.

CloudFormation stack implementation

In this step CloudFormation stack scripts will be created using yml format file, to build a 3tier web application which consist with all the infrastructure module script in it with the following steps.

1. Create VPC CloudFormation Template, this script will be accommodating the creation of VPC environment on cloud that will include VPC itself, 4 EC2 Instances (2 for public and 2 for private), Public and Private Subnet, Internet Gateway, security groups for the VPC and and routing destination for each subnet. In this file, to create EC2, the researcher chooses to use the autoscaling group to create EC2 instead of defining the EC2 stack.

2. Create Amazon RDS CloudFormation Template**,** this template will create RDS with MySQL database version 8.0.

3. Create Nat Gate CloudFormation Template, this template will build the Nat Gateway that will allow the 2 private EC2 instances that host the application to access the internet as they might need to connect with APIs, but it will not allow any one from outside access these instances directly.

## System Testing and Result

In this chapter, we will discuss about the system testing and result on the comparison between Terraform and CloudFormation stack. The comparation criteria will be Capability, Efficiency, Error Handling, and Rollback Ability

1. Capability Test Result, After completed the creation of the infrastructure on AWS using Terraform and Cloud Formation stack. The result will be gathered to see if all necessary components that needed for 3 tier web application were built successfully using both tools. All the resources type that need to be provisioned on AWS can be provisioned using both tools, so we cannot conclude on which tools is better in terms of the capabilities test.

2. Efficiency Test Result, We can conclude that in terms of time efficiency, CloudFormation stack has the edge as it ran 33% faster. However, for the Resource Used efficiencyy we cannot conclude as we cannot measure CloudFormation resource used as previously mentioned. Anyways, for terraform, we recommended that the spec of 4Cores CPU and 4 GB of memory of a basic computer should be able to run Terraform without any issue.

3. Error Handling Test Result, The result in terms of Error Handling Test, Terraform is a little bit better in the error message department as it showed much clearer message even tell you which line of the code is having a problem. While the CloudFormation Stack does not give you that kind of details. On the other criteria, both online instruction can be found very easily.

4. Rollback Test Result, The result in terms of Rollback Ability Test, Both tools have a store state file. The rollback method is a bit better on CloudFormation since it has the auto rollback. Both are easy to rollback and have a robust online instruction.

## Result Summary and Suggestions

As we have seen, both CloudFormation and Terraform offer powerful IaC capabilities, but it is important to consider your workload, team composition, and infrastructure needs when selecting IaC platform.

CloudFormation is a better option if your entire infrastructure is on AWS and there are no plans to go multi-cloud. If you are new to AWS services, native support would be beneficial. It is built by AWS and has faster AWS-related updates. It also uses JSON and YAML, so there is no learning curve as opposed to HCL. Terraform is the best option if you are using or planning to use multi-cloud resources. The modular approach allows you to create reusable templates, which speed up the configuration process. The bottom line is this research still cannot be undetermined on what is best for you, or your organization is depends on the requirements. The researcher recommends selecting the IaC tools after evaluating your application's infrastructure strategy.

**Suggestions**

1. Try to create more sophisticated infrastructure than 3 tier web application and see how both tools response.

2. Use both tools to complement each other's when building the infrastructure and see how they fit in with each other's.

3. Not just use both tools to build or provisioning the infrastructure but try to use in the configuration management also. For example, try to deploy application using the tools and see how they are response.

## REFERENCES

Artac, M. (2017). DevOps: Introducing Infrastructure-as-Code. In 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), (pp. 497-498). IEEE.

Brikman, Y. (2019). Terraform: Up & Running: Writing Infrastructure as Code. O'Reilly Media, Inc.

Campbell, B. (2020). The Definitive Guide to AWS Infrastructure Automation: Craft Infrastructure-as- Code Solutions. Apress.

Labouardy, M. (2021). Pipeline as Code: Continuous Delivery with Jenkins, Kubernetes, and Terraform. Manning Publications.

Alexander, S. (2019). https://www.techtarget.com/searchioperations/definition/Terraform.

Contino (2022). https://www.contino.io/insights/aws-cloudformation.

Murphy, O. (2022). Adoption of Infrastructure as Code (IaC) in Real World; Lessons and practices from industry. JAMK University of Applied Sciences.

Rahman, A. (2021). Systematic mapping study of IaC. 41st International Conference on Software Engineering.