การประเมินประสิทธิภาพของโครงสร้างพื้นฐาน

ในรูปแบบบริการของผู้ให้บริการระบบคลาวด์

PERFORMANCE EVALUATION OF INFRASTRUCTURE AS A SERVICE

ACROSS CLOUD SERVICE PROVIDERS

ศรัณย์ สิทธิโยภาสกุล

SARAN SITHIYOPASAKUL

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2566

KMITL-2023-EN-M-027-047

PERFORMANCE EVALUATION OF INFRASTRUCTURE AS A SERVICE ACROSS CLOUD SERVICE PROVIDERS

SARAN SITHIYOPASAKUL

A THESIS SUMMITED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRICAL AND COMPUTER ENGINEERING

SCHOOL OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2023

KMITL-2023-EN-M-027-047

หัวข้อวิทยานิพนธ์	การประเมินประสิทธิภาพของโครงสร้างพื้นฐาน ในรูปแบบบริการของผู้ให้บริการระบบคลาวด์
นักศึกษา	นาย ศรัณย์ สิทธิโยภาสกุล
รหัสประจำตัว	65016090
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
พ.ศ.	2566
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ.ดร.ชวลิต เบญจางคประเสริฐ

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์ เพื่อตรวจสอบ วิเคราะห์ และเปรียบเทียบประสิทธิภาพของโครงสร้างพื้นฐานในรูปแบบบริการ (IaaS) ระหว่างผู้ให้บริการระบบคลาวด์ที่เลือก เพื่อให้มั่นใจว่าผู้ให้บริการคลาวด์รายใดมีเสถียรภาพ ความน่าเชื่อถือ และความสามารถในการปรับขนาดได้มากกว่ากัน วิทยานิพนธ์ฉบับนี้มุ่งเน้นไปที่การทดสอบประสิทธิภาพตามเว็บเซิร์ฟเวอร์ที่ใช้งานในสภาพแวดล้อมคลาวด์ คุณสมบัติหลักของคลาวด์คอมพิวติ้งคือความสามารถในการปรับขนาด ดังนั้นผู้ให้บริการคลาวด์ IaaS (CSP) ทั่วไปส่วนใหญ่จึงมีคุณสมบัติปรับขนาดอัตโนมัติสำหรับเครื่องเสมือน (Virtual machine) วิทยานิพนธ์ฉบับนี้ไม่เพียงแต่ให้ผลการทดลองของการทดสอบความสามารถในการขยายขนาดเท่านั้น แต่ยังให้ผลการทดสอบการกู้คืนเพื่อตรวจสอบว่าเว็บเซิร์ฟเวอร์สามารถกู้คืนจากความล้มเหลวและการทดสอบโหลดที่จำลองคำขอทราฟิกใช้เวลามากหรือน้อยอย่างไร การทดสอบจะดำเนินการในระบบคลาวด์สาธารณะหลัก ๆ ของ Google Cloud Platform (GCP), Microsoft Azure และ Amazon Web Services (AWS) จากผลการทดสอบความสามารถในการปรับขนาด Azure ทำงานได้ดี โดยภาพรวมทั้งในการปรับขนาดออกและขยายขนาดที่ 540 วินาที และ 570 วินาที ผลการทดสอบการกู้คืนจากสถานการณ์ความล้มเหลวของเซิร์ฟเวอร์ AWS มีเวลาการกู้คืนที่เร็วที่สุด โดยใช้เวลาเพียง 89 วินาที และในการทดสอบโหลดได้ผลลัพธ์ดังนี้ Azure ทำงานได้เร็วที่สุดด้วยเวลาตอบสนองเฉลี่ยต่ำสุดที่ 186 มิลลิวินาที แม้ว่า GCP จะมีเวลาตอบสนองขั้นต่ำที่ 129 มิลลิวินาที สำหรับแค่บางกลุ่มตัวอย่าง แต่ก็ยังมีเวลาตอบสนองเฉลี่ยที่ 322 มิลลิวินาที ซึ่งยังช้ากว่า Azure และ AWS มีการเชื่อมต่อเซิร์ฟเวอร์ล้มเหลวที่น้อยสุด โดยพิจารณาจากเปอร์เซ็นต์ข้อผิดพลาดต่ำสุดที่ 0.23%

Thesis	Performance Evaluation of Infrastructure as a Service across Cloud Service Providers
Student	Mr. Saran Sithiyopasakul
Student ID.	65016090
Degree	Master of Engineering
Program	Electrical and Computer Engineering
Year	2023
Thesis Advisor	Assoc.Prof.Dr.Chawalit Benjangkaprasert

ABSTRACT

The purpose of this research aims to monitor, analyze, and compare the performance of infrastructure as a service (IaaS) between the selective cloud providers. To assure which cloud provider has more stability, reliability, and scalability. This thesis focuses on performance testing based on a deployed web server in the cloud environment. The main feature of cloud computing is scalability thus most common IaaS cloud service providers (CSPs) have Auto Scaling features for instances or virtual machines. Not only does this thesis gives the experimental results of the scaling scalability testing, but it also provides the results of recovery testing to inspect how long a web server is able to recover from failures and load testing which simulated traffic requests. Testing was conducted in the major public clouds of Google Cloud Platform (GCP), Microsoft Azure (Azure), and Amazon Web Services (AWS). In the scalability result, Azure performs well on overall both scale-out and scale-in of 540 and 570 seconds. In the recovery testing result, AWS demonstrated the fastest recovery time, taking only 89 seconds to recover from the failure scenario. In the load testing result, Azure performs the fastest with the lowest average response time of 186 ms. Although GCP has the minimum time requests taken of 129 ms for some requests, it has an average response time of 322 ms which is still slower than Azure. AWS becomes the least failed requests based on the lowest percentage error of 0.23%.

ACKNOWLEDGEMENT

I would like to appreciate and extend my sincere gratitude to my supervisor, Assoc. Prof. Dr. Chawalit Benjangkaprasert, for the educational possibilities that my committee has provided. I thank him a huge amount of gratitude for helping me fulfill my project. Without any assist, I would not have been successful in completing my thesis, which gives me a lot of motivation to conclude my work. All great works in this research would not have been possible without them. I truly appreciate and gladly acknowledge their contributions. Last but not least, I want to express my gratitude to my family for supporting me in completing my education. My family has hugely affected me and widened my perspective.

Saran Sithiyopasakul

TABLE OF CONTENTS

Page

บทคัดย่อ..... ..I

Abstract.....II

Acknowledgment.....III

Table of contents..... IV

List of tables.....VII

List of figures.....VIII

CHAPTER 1 – Introduction.....1

1.1 Problems and background.....1

1.2 Cloud Computing Overview 2

1.3 Infrastructure as a Service (Iaas).....5

1.4 Performance Metrics and Evaluation 6

CHAPTER 2 – Related Theory.....7

2.1 Hypertext Transfer Protocol.....8

2.2 Secure Shell.....10

2.3 Operating System.....12

2.4 Virtual Machine 13

2.5 Autoscaling 14

2.6 Load Balancer 15

CHAPTER 3 – Cloud Service Providers Overview.....17

3.1 Google Cloud Platform....17
3.2 Microsoft Azure....18
3.3 Amazon Web Services....19
3.4 Comparison of Cloud Service Provider....20
CHAPTER 4 – Research Methodology....24
4.1 Research Design and Approach....24
4.2 Experimental Setup....26
4.3 Performance Testing Tools and Scripts....30
CHAPTER 5 – Performance Evaluation....36
5.1 Performance Metrics Selection....36
5.2 Google Compute Engine Implementation....37
5.3 Azure Virtual Machines Implementation....45
5.4 AWS EC2 Implementation....52
5.5 Scalability Testing....58
5.6 Recovery Testing....62
5.7 Load Testing....64
CHAPTER 6 – Discussion....75
6.1 Findings and Implications....75
6.2 Limitations and Future Work....76
CHAPTER 7 – Conclusion....79
7.1 Summary of the Study....79
7.2 Contribution and Significance of the Study....81
7.3 Recommendations for Future Research....82

References.....84

Appendix.....86

Published research articles.....87

Author biography.....88

LIST OF TABLES

Table

Page

1 Experimental VM Configuration....26

2 CSPS Auto Scaling Configuration....29

3 Summary Report....72

4 Functionality Score....80

LIST OF FIGURES

Figure

Page

1.1 Cloud Model....3

2.1 Hypertext transfer protocol....8

2.2 HTTP and HTTPS connection....9

2.3 Secure shell encrypted connection....10

2.4 Virtual machine contained operating system....13

3.1 Products comparison....21

4.1 Auto Scaling Structure Overall....24

4.2 Comparing the geographical coverage of CSPs....27

4.3 Installing Nginx through SSH for CSPs....28

4.4 Nginx Web Interface.....29

4.5 Flow chart of scalability testing....31

4.6 Flow chart of recovery testing....32

4.7 An instance status while unhealthy....33

4.8 Load testing by using the Apache JMeter tool....33

4.9 JMeter Listeners Formats....35

5.1 Create instance group on GCP....38

5.2 Setup autoscaling metrics on GCP....39

5.3 Configure load balancer on GCP....40

5.4 Load balancer details on GCP....40

5.5 Review instance groups on GCP 41

5.6 Nginx web server interface of GCP 41

5.7 Connect to instance via SSH of GCP 42

5.8 Install stress via SSH of GCP 43

5.9 Execute stress command via SSH of GCP 43

5.11 Instance group members on GCP 44

5.12 Stop nginx web server via SSH of GCP result.....44

5.13 Instance heath of GCP 45

5.14 Create an image on Azure 46

5.15 Create virtual machine scale set on Azure 47

5.16 Configure virtual machine settings on Azure 48

5.17 Review instance details on Azure 49

5.18 Nginx web server interface of Azure 49

5.19 Connect to instance via SSH of Azure 50

5.20 Install stress via SSH of Azure 51

5.21 Execute stress command via SSH of Azure 51

5.22 Review instances in azure-vmss 51

5.23 Stop nginx web server via SSH of Azure 52

5.24 Recheck the instance health on Azure 52

5.25 Configure instance on AWS 53

5.26 Create Auto scaling group on AWS 54

5.27 Review the Auto Scaling groups on AWS 54

5.28 View instances on AWS 55

5.29 Nginx web server interface of AWS 55

5.30 Connect to instance on AWS browser 56

5.31 Connect to instance via SSH of AWS 56

5.32 Install stress via SSH of AWS 57

5.33 Review instances on AWS 57

5.34 Stop nginx web server via SSH of AWS 58

5.35 Recheck instance health on AWS 58

5.36 CPU Utilization and Auto Scaling Monitoring Graphs of GCE 59

5.37 CPU Utilization and Auto Scaling Monitoring Graphs of Azure VMs 60

5.38 CPU Utilization and Auto Scaling Monitoring Graphs of AWS EC2 60

5.39 Auto Scaling Average Time 61

5.40 Auto Recovery Instance Time 62

5.41 Aggregate graph of 500 requests 65

5.42 Aggregate graph of 1000 requests 66

5.43 Aggregate graph of 3000 requests 67

5.44 Aggregate graph of 5000 requests 68

5.45 Aggregate graph of 10000 requests 69

5.46 Response time graph of 500 requests 70

5.47 Response time graph of 1000 requests 70

5.48 Response time graph of 3000 requests 71

5.49 Response time graph of 5000 requests 71

5.50 Response time graph of 10000 requests 71

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

Cloud computing has become a ubiquitous concept in recent years and has revolutionized the way organizations approach their IT infrastructure. It provides the flexibility, scalability, and cost-effectiveness that businesses need to meet their ever-changing demands. With the emergence of cloud computing, Infrastructure as a Service (IaaS) has become a popular model for delivering cloud services, allowing businesses to outsource their IT infrastructure requirements and concentrate on their core business operations.

Cloud computing or cloud service has become the main factor for businesses that should not be overlooked especially for businesses that need lots of flexibility and availability of very economical and reliable on-demand computing resources [1]. In today's fast-paced and highly competitive market, organizations are looking to deploy cloud services that provide high availability, scalability, and reliability. With the availability of multiple cloud service providers, businesses can choose from a variety of options to meet their specific requirements. However, selecting the right cloud service provider can be a daunting task, given the complexity of cloud infrastructures and the varying performance capabilities of different providers.

This study aims to provide a performance evaluation of the three most popular cloud service providers, namely Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS), and to provide a comparative analysis of their performance in terms of scalability, recovery, and load testing. The study also aims to identify the strengths and weaknesses of each cloud service provider and provide recommendations for businesses looking to adopt cloud services.

The purpose of this section is to provide an overview of the background and motivation behind the study, and to outline the main research questions that the study seeks to answer.

The rest of the paper is structured Followed by a literature review on cloud computing, IaaS, and performance metrics and evaluation, an overview of the three cloud service providers, details the research methodology used in the study, the results of the performance evaluation, discusses the findings and implications of the study, as well as the limitations and future work, and finally concludes the study, summarizing the main findings and contributions of the research.

1.2 CLOUD COMPUTING OVERVIEW

Cloud computing has revolutionized the way in which computing resources are accessed, managed, and delivered. It enables the on-demand availability of scalable computing resources that can be accessed through the internet. Cloud computing is a model that enables convenient, on-demand network access to a shared pool of configurable computing resources, such as servers, storage, applications, and services. The main cloud computing deployment models are public, private, and hybrid clouds and these offer different services [3].

The public cloud is a type of cloud computing that is offered by third-party cloud service providers, and it's available to the general public over the internet. Examples of public clouds include Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform. The private cloud is a type of cloud computing that is dedicated to a single organization or user group, and it's typically hosted on-premises or in a data center. The organization has complete control over the infrastructure, and it can be managed by the organization's IT department or a third-party provider. The hybrid cloud is a type of cloud computing that combines elements of both public and private clouds, allowing organizations to leverage the benefits of both deployment models. For example, an organization might use a public cloud for some workloads and a private cloud for others, and they might use a cloud management platform to manage both environments.

This research focuses on the public cloud with IaaS which is the fundamental cloud computing model. Cloud service providers propose the IaaS where a set of virtual machine computing resources, e.g. CPU, storage, and network elements are provided to the user [1][3].

Consumers can simply deploy and run their web applications using these instances. Due to the fact that there are numerous cloud service providers, service selection and assessment is a primary goal for the critical business. The model includes three main service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) as Fig. 1.1 below.

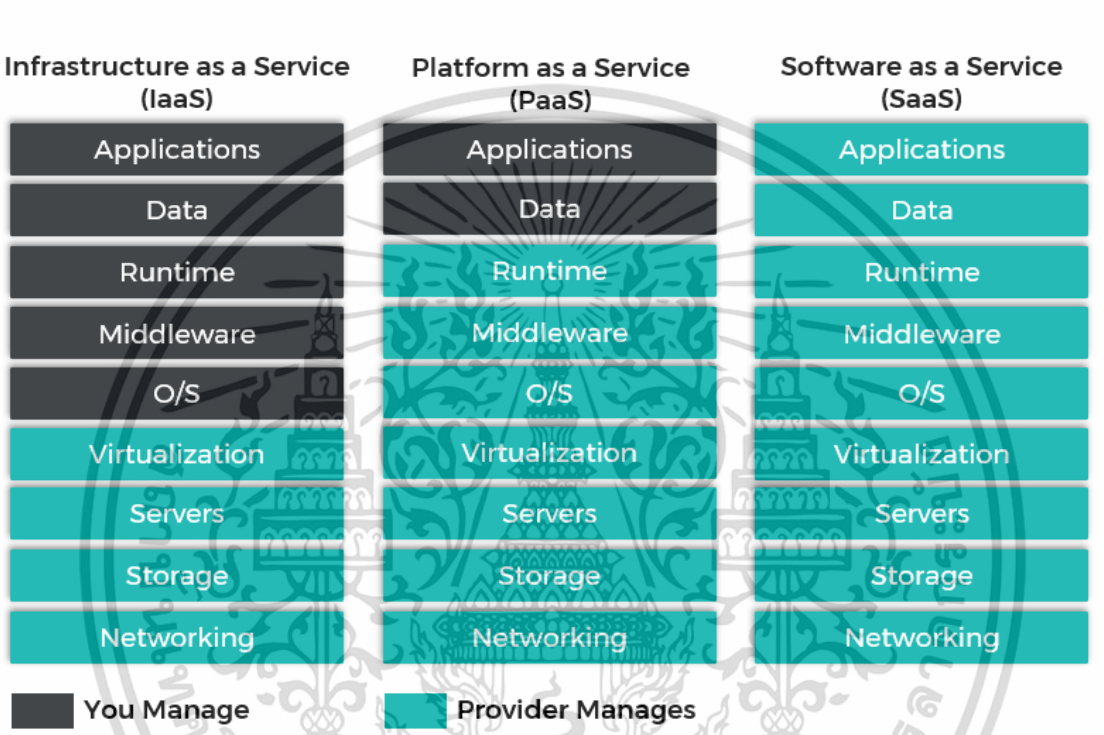


Fig. 1.1 Cloud Model

IaaS is the most fundamental and basic cloud computing service model, providing users with access to virtualized computing resources, such as servers, storage, networking, and other computing infrastructure, over the internet. IaaS enables users to rent and deploy computing infrastructure on-demand, without the need to invest in and manage physical hardware and infrastructure. IaaS is suitable for organizations that require the flexibility and scalability of cloud computing, but still need to maintain control over their infrastructure, applications, and data.

PaaS is a cloud computing service model that provides users with a complete development and deployment environment in the cloud. PaaS services provide a platform for developers to build, test, and deploy applications, without the need to manage the underlying infrastructure. PaaS services typically include a set of development tools, programming languages, databases, middleware, and other application services that can be used to develop and deploy custom applications. PaaS is suitable for organizations that need to develop and deploy custom applications quickly and efficiently, without the need to manage the underlying infrastructure.

SaaS is a cloud computing service model that provides users with access to software applications over the internet, without the need to install or maintain the applications on their own devices or infrastructure. SaaS services are typically provided as a subscription-based service, with users paying a monthly or annual fee for access to the software. SaaS is suitable for organizations that need to access and use software applications without the need to invest in and manage their own infrastructure and software.

Cloud has become a significant business solution [2] in order to reduce business costs and support future business expansion. Also suitable for the rapid change of the modern world which can simply control to access all the resources that are available through the website of cloud providers. Moreover, the cloud starts with a low price because in most cases service providers charge based on usage. Infrastructure costs e.g. server installation, electricity, hardware, and maintenance no need to pay these expenses because the cloud provider has handled it all [3]. In terms of reliability, it is quite reliable as it can systematically back up data and perform recovery in case of severe problems

One of the main benefits of cloud computing is its ability to provide scalability and elasticity to applications. Scalability refers to the ability to increase or decrease computing resources in response to changing workloads. Elasticity refers to the ability to automatically provision and de-provision computing resources based on the current workload. Cloud computing also offers a number of other benefits, such as reduced capital expenditure, increased efficiency, and improved security.

However, there are also some challenges associated with cloud computing. One of the main challenges is security, as customers need to trust their data and applications to a third-party service provider. Other challenges include data privacy, vendor lock-in, and regulatory compliance. In addition, there are also technical challenges, such as network latency, application performance, and availability.

Despite these challenges, cloud computing has become an essential part of modern computing. The cloud computing market is expected to continue to grow, with increasing adoption by businesses of all sizes. As a result, it is important for organizations to understand the benefits and challenges of cloud computing, and to have a solid understanding of the different service models and deployment models available. In the following sections, we will focus on Infrastructure as a Service (IaaS) and evaluate the performance of different cloud service providers in this category.

1.3 INFRASTRUCTURE AS A SERVICE (IAAS)

Infrastructure as a Service (IaaS) is a cloud computing model that provides virtualized computing resources such as servers, storage, and networking infrastructure to users over the internet. IaaS provides a flexible and scalable solution that enables users to create and manage their own IT infrastructure in the cloud, without the need to purchase and maintain physical hardware. In this section, we will discuss the main characteristics and benefits of IaaS, as well as the key players in the IaaS market.

One of the main characteristics of IaaS is its on-demand self-service model. Users can quickly and easily provision computing resources as needed, without the need to go through a lengthy procurement process. This allows organizations to be more agile and responsive to changing business needs. IaaS also provides a high degree of flexibility, allowing users to scale resources up or down as needed, based on changes in workload or demand.

Another key feature of IaaS is its resource pooling capability. This means that multiple users can share the same physical infrastructure, while still maintaining their own isolated virtual environment. This approach allows for greater efficiency and utilization of resources,

as well as reducing costs for the user. In addition, IaaS provides a high degree of automation and standardization, enabling users to rapidly provision and configure resources without the need for manual intervention.

Other main benefits of IaaS is its cost-effectiveness. By leveraging shared infrastructure and automation, IaaS providers can offer computing resources at a lower cost than traditional on-premises infrastructure. This can be particularly attractive for organizations that have variable or unpredictable workloads, as they can avoid the expense of maintaining infrastructure that may be underutilized.

IaaS is a cloud computing model that provides virtualized computing resources to users over the internet. IaaS provides a flexible, scalable, and cost-effective solution that allows organizations to create and manage their own IT infrastructure in the cloud. With the rapid growth of the IaaS market, organizations have more options than ever before for their IT infrastructure needs.

1.4 PERFORMANCE METRICS AND EVALUATION

Performance evaluation is a critical aspect of infrastructure as a service (IaaS) providers. As users rely on IaaS to host their applications, ensuring high-performance levels are critical to their success. Therefore, it is important to identify and understand the various performance metrics and evaluation techniques used to assess IaaS providers.

The performance of IaaS can be evaluated using different performance metrics. These metrics may include availability, reliability, scalability, responsiveness, and throughput. Availability refers to the percentage of time the infrastructure is accessible and usable by the user. Reliability refers to the ability of the infrastructure to deliver services without failure. Scalability refers to the capability of the infrastructure to handle varying levels of user demand without negatively impacting its performance. Responsiveness refers to the time taken for the infrastructure to respond to a user's request. Throughput refers to the number of transactions that can be processed within a given time frame.

Various performance evaluation techniques are used to assess IaaS providers. These techniques may include synthetic transactions, load testing, stress testing, and reliability testing. Synthetic transactions involve simulating a user's interaction with the IaaS infrastructure to assess its performance. Load testing involves testing the infrastructure under high user load to evaluate its performance. Stress testing is used to evaluate the infrastructure's ability to handle high load and simulate scenarios that may cause failures. Reliability testing is used to evaluate the infrastructure's ability to deliver services without failure.

Performance metrics and evaluation techniques can be combined to provide a more comprehensive performance evaluation of IaaS providers. For instance, load testing can be used to evaluate scalability and throughput, while reliability testing can be used to evaluate reliability and availability. By using a combination of metrics and evaluation techniques, a more accurate and comprehensive evaluation of IaaS providers can be achieved.

Evaluating the performance of IaaS providers is critical for ensuring the success of user applications. Different performance metrics and evaluation techniques can be used to assess IaaS performance. Combining multiple metrics and evaluation techniques can provide a more comprehensive evaluation of IaaS providers. As technology continues to evolve, performance evaluation techniques must also evolve to ensure high-performance levels are maintained.

CHAPTER 2

RELATED THEORY

2.1 HYPERTEXT TRANSFER PROTOCOL

HTTP (Hypertext Transfer Protocol) is a communication protocol port 80 used for transferring data between web servers and web clients, such as web browsers. It operates on top of the TCP/IP network layer and allows clients to request resources from servers by sending HTTP requests, and servers to respond with HTTP responses containing the requested data as shown in Fig. 2.1.

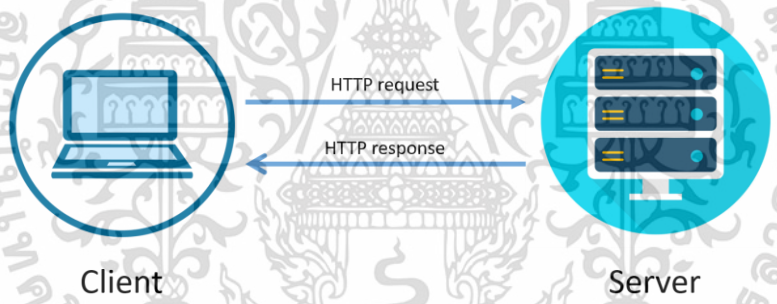


Fig. 2.1 Hypertext transfer protocol request and response

HTTP (Hypertext Transfer Protocol) and HTTPS (Hypertext Transfer Protocol Secure) are both communication protocols used for transferring data between web servers and clients. However, there are important differences between the two protocols. HTTP does not provide any encryption or security mechanisms, making it vulnerable to interception and tampering by attackers.

HTTPS, on the other hand, is a secure version of the HTTP protocol. It uses encryption to protect data in transit as shown in Fig. 2.2, making it more difficult for attackers to intercept and view sensitive information. HTTPS works by adding an SSL/TLS layer on top of the HTTP

protocol, which uses cryptographic techniques to establish a secure connection between the client and server and to encrypt data transmitted over the connection. One of the main benefits of HTTPS is improved security. By encrypting data in transit, HTTPS prevents attackers from intercepting sensitive information such as passwords, credit card numbers, and personal information. HTTPS is particularly important for protecting the privacy and security of online transactions, such as online banking and e-commerce.

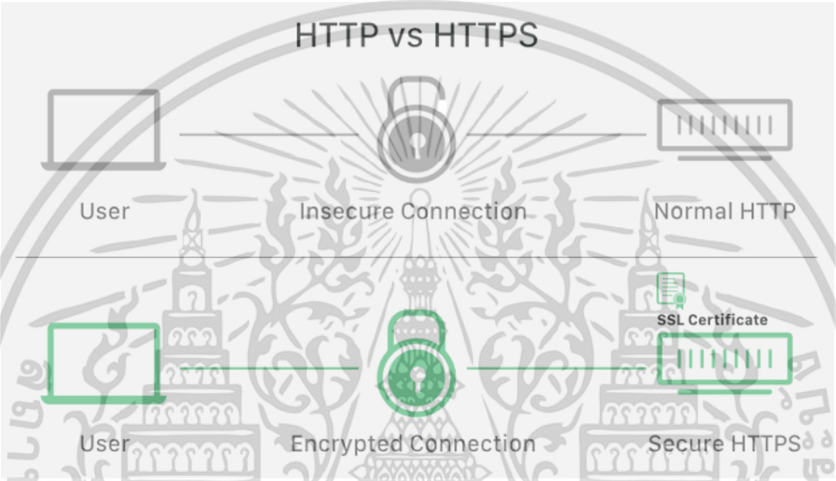


Fig. 2.2 HTTP and HTTPS connection

HTTP requests consist of several components, including a method, a URI (Uniform Resource Identifier), and headers. The method specifies the type of request, such as GET to retrieve data or POST to submit data to a server. The URI identifies the resource being requested, such as a web page or a file. Headers provide additional information about the request, such as the type of content being sent or accepted. HTTP responses also consist of several components, including a status code, headers, and a message body. The status code indicates the outcome of the request, such as 200 for success or 404 for not found. Headers provide additional information about the response, such as the type of content being sent or caching instructions. The message body contains the requested data, such as the HTML content of a web page.

HTTP also supports other methods besides GET and POST, such as PUT, DELETE, and HEAD. PUT is used to update an existing resource, while DELETE is used to remove a resource.

HEAD is used to retrieve only the headers of a resource, without the message body. These methods allow clients to perform more advanced operations on resources. HTTP also supports cookies, which are small pieces of data sent from a server to a client and stored in the client's browser. Cookies can be used to track user sessions or preferences, and are often used for authentication purposes. However, cookies can also be used for tracking user behavior and privacy concerns have been raised about their use.

Both HTTP and HTTPS are communication protocols used for transferring data between web servers and clients. While HTTP is the basic protocol used for communication, HTTPS provides added security and encryption to protect sensitive information in transit. However, implementing HTTPS can be more complex and resource-intensive than HTTP.

2.2 SECURE SHELL

SSH (Secure Shell) is an encryption protocol port 22 that isused for managing most Unix/Linux servers, Internet routers, and much of cloud computing infrastructure [4]. It is commonly used for remote access to servers and other systems, allowing users to securely log in to a remote system and execute commands as if they were using the local console.

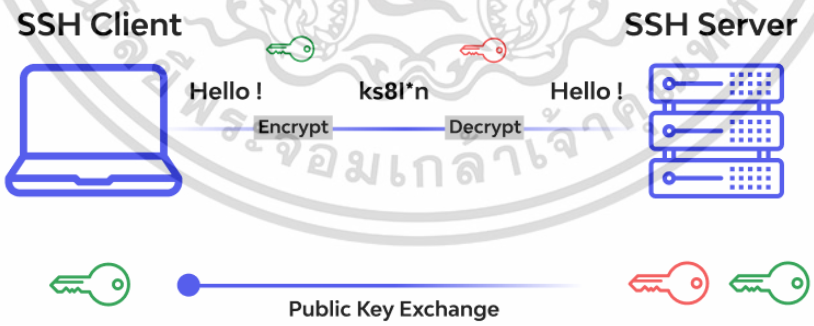


Fig. 2.3 Secure shell encrypted connection

SSH works by establishing an encrypted connection between the client and the server as shown in Fig. 2.3, which provides protection against eavesdropping, tampering, and other

forms of network-based attacks. It uses public keys for authenticating servers and users [4] ensuring that only authorized users are able to access the system.

One of the main benefits of SSH is improved security. By encrypting all data transmitted between the client and server, SSH prevents attackers from intercepting sensitive information such as passwords, usernames, and other credentials. This makes it much more difficult for attackers to gain unauthorized access to a system. Another benefit of SSH is its flexibility. It can be used for a wide range of tasks, including remote access, file transfers, and tunneling of other protocols. This makes it a versatile tool for system administrators and developers, who can use it to manage and secure a wide range of systems and applications.

SSH also supports a wide range of authentication methods, including password-based authentication, public-key authentication, and multi-factor authentication. This allows users to choose the most appropriate authentication method for their needs, depending on the level of security required and the complexity of the system being accessed.

However, there are also some drawbacks to using SSH. For example, SSH can be more complex to set up and configure than other remote access protocols. Additionally, SSH sessions can consume a significant amount of network bandwidth, particularly when transferring large files or using graphical applications over the remote connection.

SSH is a secure network protocol used for remote access to systems and other applications. It provides encryption and authentication to protect sensitive data transmitted between the client and server, and supports a wide range of authentication methods for added security. While SSH can be more complex to set up and configure than other remote access protocols, it provides a versatile and secure solution for managing and securing a wide range of systems and applications.

2.3 OPERATING SYSTEM

Operating system (OS) is the software that manages all the resources of a computer and provides a platform for other software applications to run. It is responsible for allocating resources such as memory, processor time, and input/output devices, and for providing a user interface that allows users to interact with the computer. One of the main functions of an operating system is memory management. The operating system is responsible for allocating memory to different programs and ensuring that they do not interfere with each other. It also manages virtual memory, which allows programs to use more memory than is physically available.

Another important function of an operating system is process management. The operating system is responsible for creating and managing processes, which are individual instances of programs running on the computer. It is also responsible for scheduling processes and allocating processor time to them. An operating system also provides a user interface that allows users to interact with the computer. This can be a graphical user interface (GUI), which allows users to interact with the computer using a mouse and keyboard, or a command-line interface (CLI), which allows users to enter commands directly.

The operating system also provides a platform for other software applications to run. This includes everything from productivity software such as word processors and spreadsheets to entertainment software such as video games and media players. Without an operating system, these applications would not be able to run on the computer. Another important function of an operating system is device management. The operating system is responsible for managing input/output devices such as keyboards, mice, printers, and network adapters. It provides device drivers that allow software applications to communicate with these devices and provides a unified interface for accessing them.

Operating system is the software that manages all the resources of a computer and provides a platform for other software applications to run. It is responsible for memory management, process management, providing a user interface, providing a platform for other software applications, and device management. Without an operating system, a computer would not be able to function.

2.4 VIRTUAL MACHINE

Virtual machine (VM) is a software emulation of a physical computer system that allows multiple operating systems (OS) to run on a single physical machine as shown in Fig. 2.4. Each virtual machine is isolated from the others, with its own virtual hardware and operating system, allowing multiple applications and services to run concurrently on the same physical hardware.

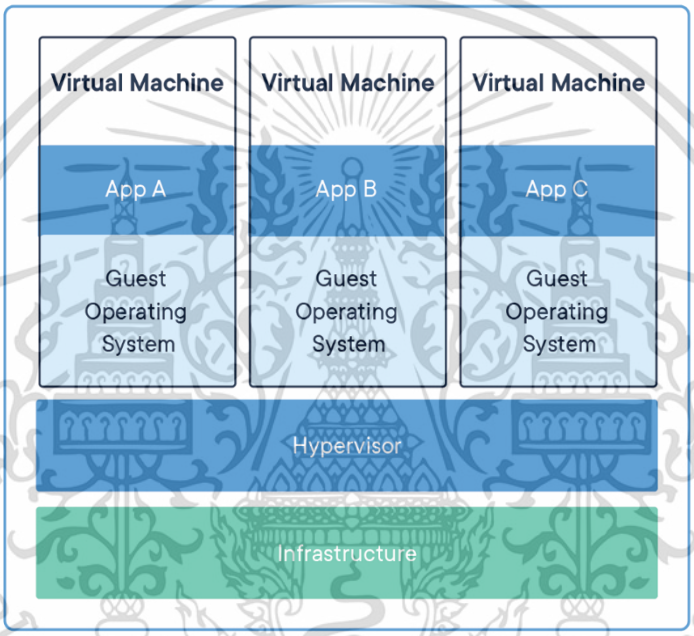


Fig. 2.4 Virtual machine contained operating system

One of the main benefits of using a virtual machine is improved efficiency and resource utilization. By allowing multiple operating systems and applications to run on a single physical machine, virtual machines can help organizations make better use of their hardware resources, reducing the need for additional hardware and associated costs. Another benefit of virtual machines is improved security. By isolating each virtual machine from the others, virtual machines can help prevent the spread of malware and other security threats between different systems. In addition, virtual machines can be easily backed up and restored, allowing organizations to quickly recover from security incidents or other disruptions.

Virtual machines can also provide a high degree of flexibility and scalability. Because virtual machines are essentially software-based, they can be easily created, cloned, and migrated to other physical hardware as needed, allowing organizations to quickly provision new systems and scale up their infrastructure as their needs change.

However, there are also some drawbacks to using virtual machines. For example, virtual machines can consume significant amounts of resources, particularly memory and CPU cycles, which can impact overall system performance. Additionally, managing a large number of virtual machines can be complex and require specialized skills and expertise.

Another potential issue with virtual machines is compatibility. Some applications may not be fully compatible with virtual machines, or may require additional configuration or tuning to run effectively. In addition, some applications may require direct access to hardware resources, which may not be available in a virtualized environment.

Virtual machines provide a powerful tool for organizations looking to improve efficiency, scalability, and security in their IT infrastructure. While virtual machines can be complex to manage and may have compatibility issues with some applications, they offer a flexible and cost-effective way to make better use of hardware resources and improve overall system performance.

2.5 AUTOSCALING

Autoscaling is a technique used in cloud computing to dynamically adjust the amount of computing resources allocated to an application based on demand. With autoscaling, an application can automatically increase or decrease the number of virtual machines, containers, or other computing resources used to run the application, based on the current workload. One of the main benefits of autoscaling is improved performance and availability. By automatically adding computing resources when demand is high, an application can continue to provide fast response times even during periods of peak demand. Conversely, when demand decreases, autoscaling can reduce the number of resources used, reducing costs and improving efficiency.

Autoscaling can also help ensure that an application remains available even in the event of hardware failures. By using multiple virtual machines or containers, autoscaling can provide redundancy and failover capabilities that can help prevent downtime or data loss. Another benefit of autoscaling is improved cost efficiency. By using only the computing resources that are needed at any given time, autoscaling can help organizations avoid overprovisioning their infrastructure and reduce costs associated with unused resources.

Autoscaling can also improve the agility and flexibility of an organization's IT infrastructure. By allowing applications to quickly adjust to changing demand, autoscaling can help organizations rapidly deploy new applications or services and respond to changing business requirements. However, there are also some challenges associated with autoscaling. For example, properly configuring autoscaling algorithms can be complex, and there is a risk of overprovisioning or under provisioning resources if the algorithms are not optimized correctly. In addition, autoscaling can increase the complexity of an organization's IT infrastructure, which can make it more difficult to manage and maintain.

Another potential issue with autoscaling is the cost of virtual machines or containers. While autoscaling can help reduce costs by eliminating the need for overprovisioning, the cost of virtual machines or containers can still be significant, particularly for organizations with large-scale applications or high levels of demand. Autoscaling is a powerful technique that can help organizations improve performance, availability, cost efficiency, and flexibility in their IT infrastructure. However, it also presents challenges and requires careful planning and management to ensure that it is optimized for the specific needs of the organization.

2.6 LOAD BALANCER

Load balancer is It is a technique of reassigning the entire load to the distinct nodes of the collaborative system to make resource utilization efficacious and to upgrade the response time of the job, concurrently eliminating a condition in which some of the nodes are over loaded [5]. This load contemplate can be in terms of CPU load, quantity of memory utilized, dawdle or Network load [5].

One of the primary benefits of load balancers is improved performance. By distributing incoming network traffic across multiple servers or other resources, load balancers can help prevent any one server from becoming overloaded and slowing down performance. This can help ensure that users have fast response times and can access resources quickly. Load balancers can also help improve resource utilization by distributing traffic to the most appropriate server based on factors such as capacity, latency, and health. This can help prevent underutilized servers from wasting resources and ensure that all resources are being used efficiently.

Another benefit of load balancers is increased availability. By distributing traffic across multiple servers or other resources, load balancers can help ensure that even if one server fails or goes offline, users can still access the application or service through other servers. This can help prevent downtime and ensure that critical services remain available. Load balancers can also provide additional security benefits by providing a single point of entry into an organization's network. By directing all incoming traffic through the load balancer, organizations can monitor and filter traffic more effectively, identify potential security threats, and provide an additional layer of protection against attacks.

However, there are also some potential drawbacks to using load balancers. For example, load balancers can introduce additional complexity into an organization's network architecture, and they require careful configuration and management to ensure that they are working effectively.

Another potential issue with load balancers is the cost. Hardware-based load balancers can be expensive to purchase and maintain, and cloud-based load balancers can incur ongoing usage fees that can add up over time. Load balancers are a powerful tool for improving performance, resource utilization, availability, and security in an organization's network. However, they also present challenges and require careful management to ensure that they are optimized for the specific needs of the organization.

CHAPTER 3

CLOUD SERVICE PROVIDERS OVERVIEW

3.1 GOOGLE CLOUD PLATFORM

Google Cloud Platform (GCP) is a cloud computing platform provided by Google that offers a variety of infrastructure and platform services, including data storage, virtual machines, and application development tools. GCP is designed to provide high performance, security, and scalability, making it a popular choice for businesses of all sizes.

One of the key features of GCP is its global network infrastructure, which consists of dozens of data centers located around the world. This allows businesses to deploy their applications and services in the location that is closest to their customers, reducing latency and improving performance. GCP also offers a variety of tools for managing and securing data, such as Cloud Storage and Cloud SQL, making it easy to store and manage data in a secure and compliant manner.

Another important feature of GCP is its emphasis on open-source software and development tools. GCP supports a variety of popular open-source frameworks and programming languages, such as Python, Java, and Node.js, making it easy for developers to build and deploy applications using their preferred tools. GCP also offers a variety of pre-built machine learning models and tools, such as AutoML, making it easy for businesses to build and deploy machine learning applications without needing extensive expertise in the field.

GCP also offers a variety of pricing models to meet the needs of different businesses. For example, GCP offers pay-as-you-go pricing for many of its services, allowing businesses to only pay for the resources they actually use. GCP also offers discounts for long-term commitments and pre-emptible instances, which can help businesses save money on their cloud computing costs.

GCP is a robust and flexible cloud computing platform that offers a variety of infrastructure and platform services. Its global network infrastructure, support for open-source

software and development tools, and flexible pricing models make it a popular choice for businesses looking to build and deploy applications in the cloud.

3.2 MICROSOFT AZURE

Microsoft Azure is a cloud computing platform provided by Microsoft that offers a variety of infrastructure and platform services, including data storage, virtual machines, and application development tools. Azure is designed to provide a reliable and scalable cloud computing environment for businesses of all sizes, from small startups to large enterprises.

One of the key features of Azure is its ability to integrate with Microsoft's other services, such as Office 365 and Dynamics 365. This allows businesses to build and deploy applications that are tightly integrated with their existing IT infrastructure, making it easy to manage data and applications across different systems. Azure also offers a variety of tools for managing and securing data, such as Azure Key Vault and Azure Information Protection, making it easy to store and manage data in a secure and compliant manner.

Another important feature of Azure is its support for hybrid cloud environments. Azure offers a variety of tools and services that make it easy to build and deploy applications that span both on-premises data centers and the cloud. This allows businesses to take advantage of the scalability and flexibility of the cloud while still maintaining control over their data and applications.

Azure also offers a variety of pricing models to meet the needs of different businesses. For example, Azure offers pay-as-you-go pricing for many of its services, allowing businesses to only pay for the resources they actually use. Azure also offers discounts for long-term commitments and reserved instances, which can help businesses save money on their cloud computing costs.

Azure is a powerful and versatile cloud computing platform that offers a variety of infrastructure and platform services. Its ability to integrate with Microsoft's other services, support for hybrid cloud environments, and flexible pricing models make it a popular choice for businesses looking to build and deploy applications in the cloud.

3.3 AMAZON WEB SERVICES

Amazon Web Services (AWS) is a popular cloud service provider that offers a wide range of services, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). AWS has grown to become one of the leading cloud service providers in the market. The platform provides a range of services, including computing, storage, and networking, that enable customers to run their applications and workloads on the cloud.

One of the primary advantages of AWS is its flexibility, which allows customers to use only the services they need and scale them up or down as required. The platform also offers a pay-as-you-go pricing model, which means that customers are only charged for the services they use. This makes it easy for businesses of all sizes to use AWS, from startups to large enterprises.

Another advantage of AWS is its security features. The platform provides a range of security tools and services that are designed to protect customer data and infrastructure. These include network firewalls, identity and access management tools, encryption, and compliance certifications. AWS also complies with a range of global data protection regulations, such as GDPR and HIPAA, which makes it a popular choice for businesses in regulated industries.

AWS also provides a range of tools and services for managing and monitoring applications on the platform. These include AWS CloudFormation, which automates the deployment and management of infrastructure; AWS Elastic Beanstalk, which simplifies the deployment and management of web applications; and AWS CloudWatch, which provides real-time monitoring and alerting for AWS resources.

One of the key services offered by AWS is Amazon Elastic Compute Cloud (EC2), which provides scalable computing capacity in the cloud. Customers can use EC2 to launch virtual machines on the cloud, which can be customized with different configurations of CPU, memory, storage, and networking. EC2 also provides a range of preconfigured machine images

that can be used to launch virtual machines for different use cases, such as web servers, databases, and analytics.

AWS also provides a range of storage services, including Amazon Simple Storage Service (S3), Amazon Elastic Block Store (EBS), and Amazon Glacier. S3 is a highly scalable and durable object storage service that is designed for storing and retrieving large amounts of data. EBS is a block storage service that provides high-performance storage for EC2 instances, while Glacier is an archival storage service that is designed for long-term storage of data.

AWS offers a wide range of services and tools that make it easy for customers to run their applications and workloads on the cloud. The platform's flexibility, security features, and pay-as-you-go pricing model make it a popular choice for businesses of all sizes, while its range of management and monitoring tools make it easy to manage applications on the platform. With its extensive network of data centers and global reach, AWS is a top cloud service provider that is well-suited for a variety of use cases.

3.4 COMPARISON OF CLOUD SERVICE PROVIDER

Cloud computing is a rapidly evolving field that has transformed the way organizations manage their IT infrastructure. With the increase in demand for cloud computing, there are many cloud service providers offering different types of services. Each of these cloud service providers has its own strengths and weaknesses, which can make it challenging for organizations to choose the right provider for their needs. In this section, we will compare the three major cloud service providers, Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS), and discuss their similarities and differences.

PRODUCT	aws	Microsoft Azure	Google Cloud
Virtual Servers	Instances	IVMs	VM Instances
Platform-as-a-Service	Elastic Beanstalk	Cloud Services	App Engine
Serverless Computing	Lambda	Azure Functions	Cloud Functions
Docker Management	ECS	Container Service	Container Engine
Kubernetes Management	EKS	Kuberneter Service	Kubernetes Engine
Object Storage	S3	Block Blob	Cloud Storage
Archive Storage	Glacier	Archive Storage	Coldline
File Storage	EFS	Azure Files	ZFS / Avere
Global Content Delivery	CloudFront	Delivery Network	Cloud CDN
Managed Data Warehouse	Redshift	SQL Warehouse	Big Query

Fig. 3.1 Products comparison

Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS) are three of the most popular cloud service providers in the market. Each of these providers offers unique features and services, making it difficult to compare them directly. Fig. 3.1 shows the products of these three providers in general.

Pricing is a significant factor in choosing a cloud provider, and all three providers offer flexible pricing plans to suit the varying needs of their customers. AWS has a reputation for being the most affordable of the three, with a pay-as-you-go pricing model that offers a range of services at different prices. GCP offers similar pricing plans with discounts for sustained usage, while Azure has a complex pricing structure that can be challenging to navigate, but also offers discounts for long-term usage. It's worth noting that pricing can vary depending on the region, service, and usage, so it's essential to consider these factors when comparing the costs of each provider.

Each of the providers has a wide range of services that cater to different needs, including infrastructure as a service (IaaS), platform as a service (PaaS), and software as a

service (SaaS). AWS is known for its vast array of services, with over 200 offerings in compute, storage, networking, databases, analytics, machine learning, and more. GCP offers similar services with an emphasis on big data analytics and machine learning, while Azure has a strong focus on PaaS offerings for application development and deployment. It's important to choose a provider that offers the services we need for our specific use case.

Security is a crucial consideration when choosing a cloud provider, and all three providers have robust security measures in place. AWS is known for its security and compliance features, including encryption, identity and access management, and compliance certifications. GCP also offers a range of security features, including encryption, access controls, and monitoring tools. Azure has a similar set of security features with additional compliance certifications, making it a popular choice for organizations that require strict regulatory compliance.

All three providers offer different levels of scalability, allowing users to scale their resources up or down as needed. AWS is known for its scalability, with the ability to add or remove resources on-demand. GCP also offers flexible scaling options, with automatic scaling for compute engine instances and autoscaling for Kubernetes Engine. Azure has a similar level of scalability, with the ability to scale up or down depending on the application's needs.

Finally, each of the providers has excellent customer support, with documentation, forums, and tutorials available to help users get started. it's important to consider the level of support and documentation provided by each cloud service provider. AWS has the largest community of users, which means that there are many resources available for getting help and finding documentation. However, GCP and Azure also offer extensive documentation and support, and they are rapidly growing their user communities.

Cloud services need performance testing to ensure which one performs the most suitable for business requirements such as stability, reliability, and scalability. Although there are many cloud service providers globally, the majors are Google Cloud Platform, Microsoft Azure, and Amazon Web Services. As we had mentioned, cloud service providers offer the IaaS which is similar but different providers also have different infrastructure names. Google, Microsoft, and Amazon namely Google Compute Engine (GCE), Azure Virtual Machines (VMs),

and AWS EC2 respectively. All these well-known public clouds will be compared and tested for IaaS including scalability testing, recovery testing, and load testing with results that can be found in the experimental results.

Choosing the right cloud provider can be challenging, but AWS, GCP, and Azure are all excellent options. Each provider offers unique services and features, making it important to consider the specific needs of our use case when comparing them. Pricing, services, security, scalability, and customer support are all crucial factors to consider, and it's worth taking the time to evaluate each provider carefully before making a final decision.

CHAPTER 4

RESEARCH METHODOLOGY

4.1 RESEARCH DESIGN AND APPROACH

The research design and approach used in this study aimed to evaluate the performance of three cloud service providers, namely Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS). The study utilized an experimental approach that involved designing and executing performance tests on the selected cloud service providers to evaluate their infrastructure as a service (IaaS) offerings. The objective of the research was to determine the most suitable cloud service provider based on their performance and capabilities.

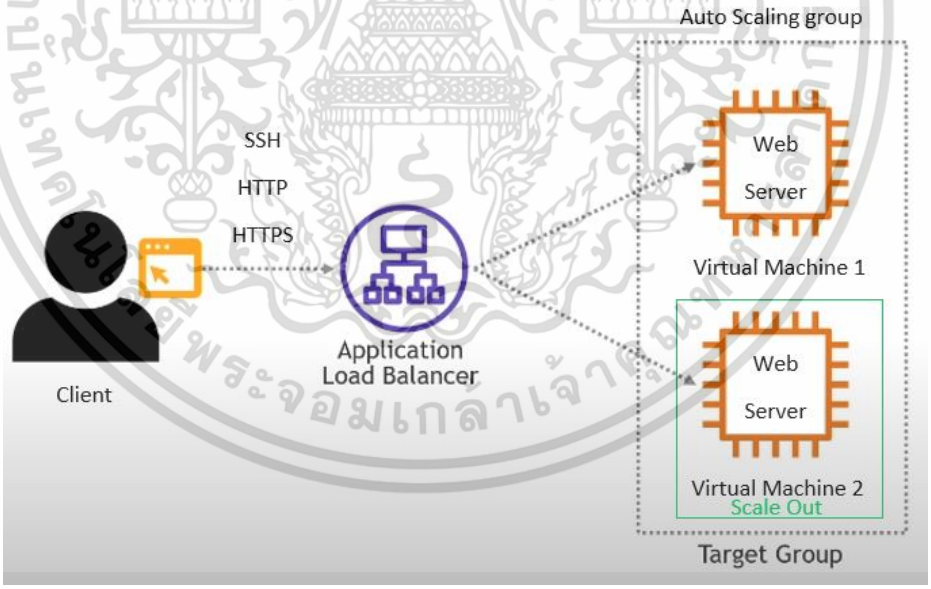


Fig. 4.1 Auto Scaling Structure Overall

Fig. 4.1 illustrates an overview of the system which simulated user requests to a web server through Hypertext Transfer Protocol (HTTP), Hypertext Transfer Protocol Secure (HTTPS), or Secure Shell (SSH) with an environment of Auto Scaling and load balancing that CSP

provided these features for us [6][7]. Auto Scaling helps maintain application availability and allows us to automatically add or reduce virtual machines or instances based on the conditions we specify. We can use cluster management features to maintain the health and availability of our clusters. Auto Scaling can predict dynamic scaling to scale-out or scale-in instances. Load balancing is a core network that distributes traffic across multiple servers. Load Balancer as shown in Fig. 4.1 can handle high-volume workloads due to the process of multiple server machines working together on the same task to distribute the workload to each machine and also can optimize response time. It has the flexibility to design a system to suit every situation.

In this experiment, it can be seen that we designed the infrastructure with an Auto Scaling group starting with just 1 virtual machine containing a web server but if it meets the conditions that we specify, it will automatically increase by 1 to 2 instances. The Auto Scaling conditions of GCE, Azure VMs, and AWS EC2 will be examined and explained in the following section.

The research approach utilized in this study was quantitative research. The approach involved the use of numerical data to evaluate the performance of the cloud service providers. The performance data obtained from the tests were analyzed using statistical techniques to identify significant differences in the cloud providers' performance. The quantitative approach provided objective and reliable data that could be used to make informed decisions.

The research design and approach used in this study had several advantages. First, the experimental approach enabled the research team to control and manipulate the variables in the study to ensure the validity and reliability of the data obtained. Second, the use of quantitative research provided objective and reliable data that could be used to make informed decisions. Third, the performance testing approach enabled the research team to evaluate the cloud providers' performance under realistic conditions, providing insights into their capabilities and limitations.

However, the research design and approach used in this study also had some limitations. First, the study only evaluated the performance of the three cloud service

providers and did not consider other factors such as cost and security. Second, the performance tests carried out in the study only simulated a limited range of workloads and did not consider the real-world complexity of cloud computing environments. Finally, the research approach used in this study may not be suitable for all research questions and may not provide sufficient insights into the research problem.

The research design and approach used in this study aimed to evaluate the performance of three cloud service providers, namely Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS). The research utilized a quantitative approach that involved the use of numerical data and statistical techniques to evaluate the performance of the cloud providers. While the approach had several advantages, it also had some limitations that should be considered when interpreting the results.

4.2 EXPERIMENTAL SETUP

The experimental setup consisted of creating identical virtual machines (VMs) on each of the three cloud platforms: Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS).

Table 1. Experimental VM configuration

CSP	Instance Type	Memory	vCPUs	OS Image	Region
GCP	e2 custom	1 GiB	1 vCPU	Ubuntu Server 22.04	Mumbai
Azure	Standard_B1s	1 GiB	1 vCPU	Ubuntu Server 22.04	Central India
AWS	t2.micro	1 GiB	1 vCPU	Ubuntu Server 22.04	Mumbai

According to Table 1, we had set the environment baseline starting point used for comparisons is going to be as same as possible in order to avoid the impact from other areas that we have not interested in. Fig. 4.2 shows the map of geographical coverage and there are

many data centers available in the selected regions provided by CSPs. We set the whole system environment including the computer system, load balancer, and server all in the same region in the Asia Pacific where GCE and AWS EC2 are in Mumbai, and Azure VMs is in Central India. The reason why we chose South Asia instead of South East Asia like Singapore which has the nearest distance compared to Thailand is Singapore has unavailable in Microsoft Azure. Therefore, Central India is the best option that we had in this experiment compared to Mumbai.

Fig. 4.2 Comparing the geographical coverage of CSPs

The instances chosen for this experiment of GCE, Azure VMs, and AWS EC2 were e2 custom, Standard_B1s, and t2.micro respectively selected as they had similar CPU capabilities which have 1 gibibyte memory and 1 virtual centralized processing unit (vCPU). GCE e2 custom can define and modify the number of memory and vCPUs but others are unable to custom their own CPU. The operating system image that we chose is Ubuntu Server 22.04 the latest version right now because it has ease of use for the developer as well as the common user, fast installation, and also provides a better option for security and privacy hence Ubuntu operating system is a perfect fit for this experimental testing as shown in Table 1. Ubuntu is an open-source operating system based on Linux.

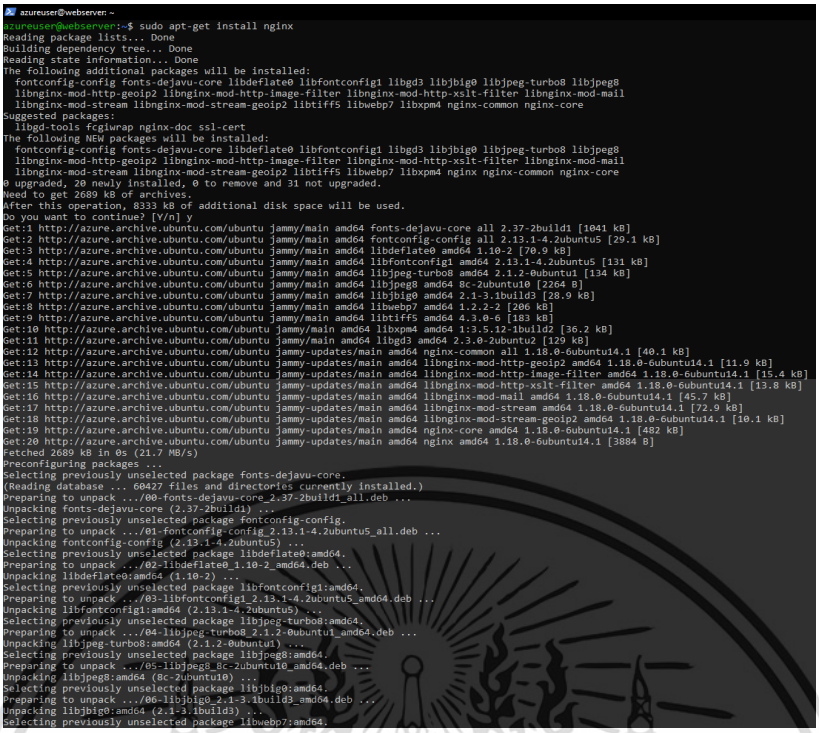


Fig. 4.3 Installing Nginx through SSH for CSPs

For the web server, we installed Nginx on the Ubuntu Server through SSH for client requests as following scripts:

sudo apt-get update

sudo apt-get install nginx

nginx -v

sudo systemctl status nginx

Fig. 4.4 Nginx Web Interface

We applied this installation as shown in Fig. 4.3 to all public clouds which we had selected. If the server gets HTTP/S requested by users, the server will respond with a web server based on Nginx as shown in Fig. 4.4 Nginx performs much faster than Apache by 2.5 times and consumes less memory as well. Even though Apache has more functionality and features, Nginx is more suitable for performance testing. By that, Nginx is the base web server in this experiment. Moving to the process of installation of the Nginx web server, start with getting the latest update of system packages, next step is installing the Nginx web server, followed by checking the version of Nginx, and finally, identifying the status of Nginx whether it is running or not.

Table 2. CSPS Auto Scaling configuration

Min. Instances	Max. Instances	Scaling Metric	Threshold
1	2	CPU Utilization	60%

In Auto Scaling conditions as shown in Table 2, we specified the scaling policies by using the scaling metric provided by Cloud Service Providers. Determining CPU Utilization with a threshold of 60% which scale-out or add one virtual machine when the maximum CPU utilization exceeds 60% and if the maximum CPU utilization is lower than 60%, the instance will be removed by one or scale-in. In this experiment, we set the maximum virtual machines to 2 so that while scaling up, there will be no more than 2 instances, and when scaling down, there will be just only 1 instance. Although the last instance gets terminated or failed, the Auto Scaling group will automatically create a new virtual machine according to the Auto Scaling policies.

The experimental setup was designed to be as similar as possible across all three cloud platforms, allowing for a fair comparison of their performance, auto-scaling, and recovery capabilities. The performance metrics were collected for each cloud platform, and the results were analyzed to evaluate their relative strengths and weaknesses.

4.3 PERFORMANCE TESTING TOOLS AND SCRIPTS

Performance testing is the process of measuring the quality of the overall system [8] that has been developed to analyze how the system handles a particular workload. If there are many user requests, the system should still respond smoothly and well. It is a test of the system before actual use. Because if we don't do a performance test before actual use, there may be lots of drawbacks that can harm our system. Generally, a web server should be tested before deploying to production to ensure that our web server is stable and reduce the chance of crashing. In this experiment, we simulated a web server in the cloud environment in order to monitor how well CSPs were dealing with the workload and managing resources.

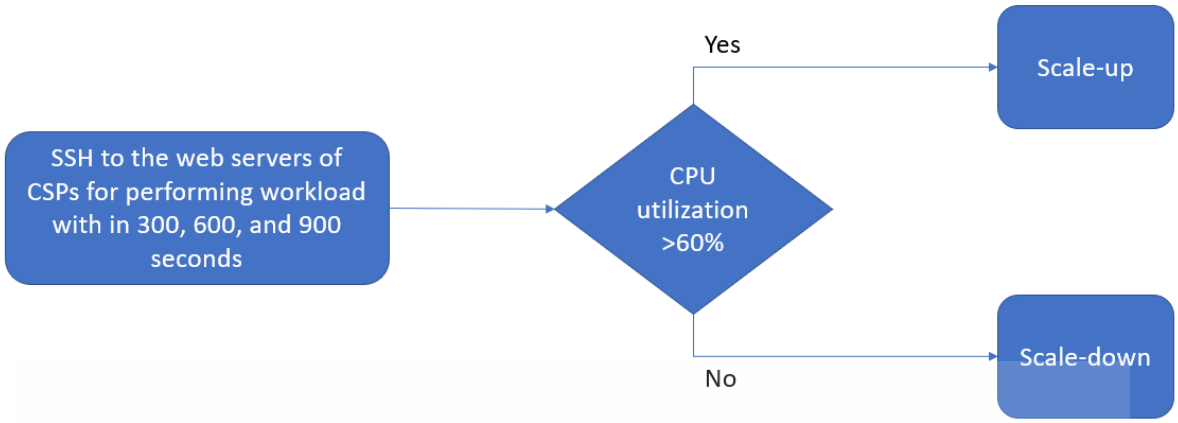


Fig. 4.5 Flow chart of scalability testing

To begin with scalability testing, Auto Scaling is the main feature of IaaS cloud service [6] so we tested its scalability by stress based on the CPU utilization metric that we had set [7]. Stress testing is an abnormal load test for the system which is more load than the reality that the system can really support. To focus on the scalability of CSPs, we imposed an artificially high CPU workload to spawn 9 workers spinning with a timeout of 300, 600, and 900 seconds to the virtual machine, so it will reach the scaling metric limitation as following scripts:

```
sudo apt-get -y install stress
stress --cpu 9 --timeout 300
stress --cpu 9 --timeout 600
stress --cpu 9 --timeout 900
```

Despite the aforementioned section, the instance will scale out or add by one when the maximum CPU utilization exceeds the threshold of 60%. If the maximum CPU utilization is lower than the threshold, the instance will scale in or be deleted by one as shown in Fig. 4.5, we collected the scaling data from the CSPs monitoring tool and then calculated the average time of these.

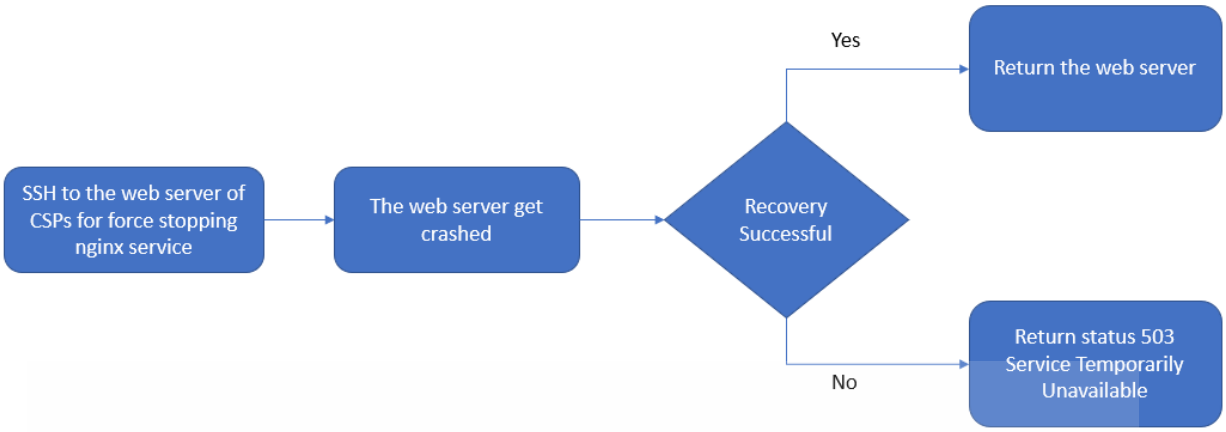


Fig. 4.6 Flow chart of recovery testing

Moving to the recovery testing, evaluating how long a web server is able to recover from failures. The web server is normally running if it does not get errored or crashed. For this simulation, we connected SSH to the server and forced stop the service as shown in Fig. 4.6 to test how well CSPs handled this issue as following scripts:

```
sudo systemctl stop nginx
sudo systemctl status nginx
```

To ensure that the service is stopped, we try to connect the web server by HTTP request and the result is 503 Service Unavailable. While monitoring the Health Check that CSPs provided, the virtual machine status was unhealthy as shown in Fig. 4.7. To compare time, we collected the exact time when the web server is crashing and the moment when a new web server is created provided by CSPs. All of these recovery results are illustrated in the bar chart to indicate which CSP recovery instance service is faster.

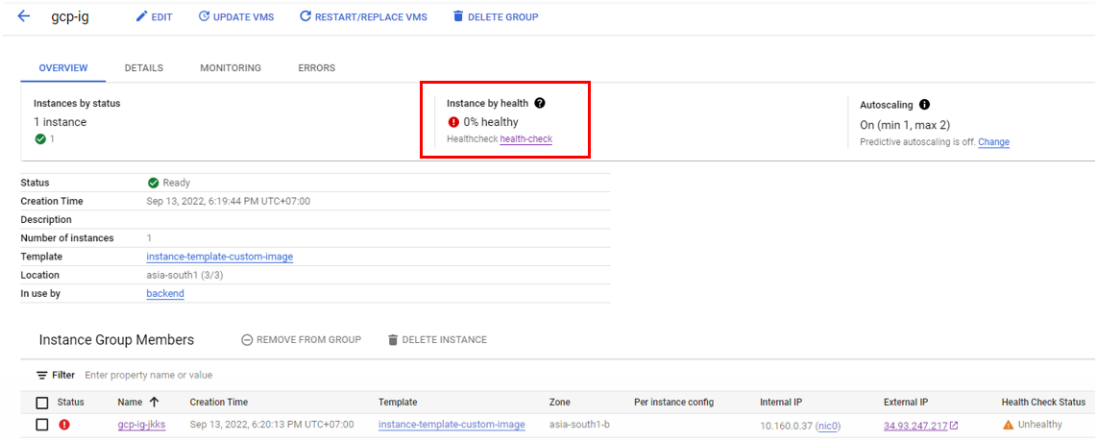


Fig. 4.7 An instance status while unhealthy

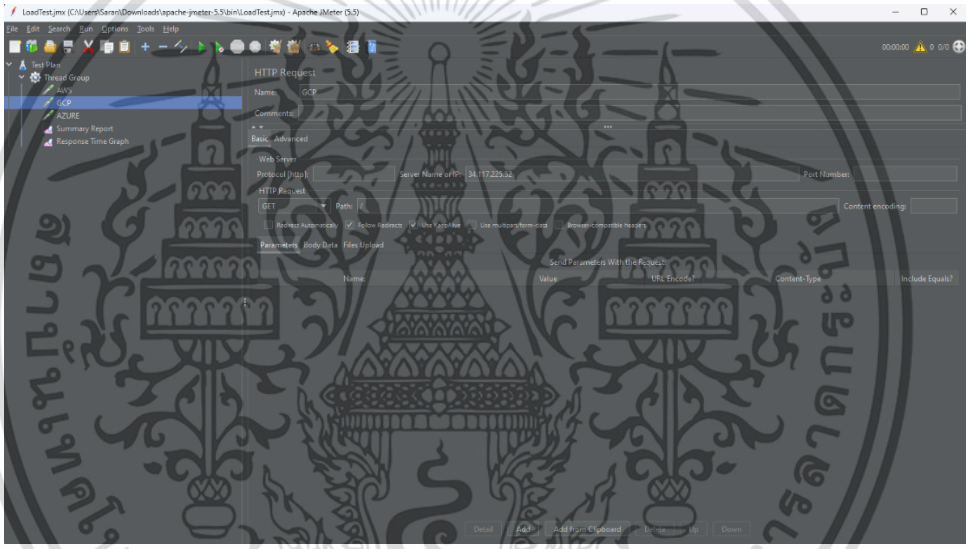


Fig. 4.8 Load testing by using the Apache JMeter tool

Followed by load testing which is the process of putting threads to measure the response times, throughput, capacity, or error rates of a system [8]. The evaluation tool that we used in this experiment is Apache JMeter as shown in Fig. 4.8. It is an open-source software tool developed by Java used for load testing of user behavior and system performance. Before using this tool, we need to install java required a completely compliant JDK 8 or higher.

Apache JMeter is an open source which provides a platform to carry load of many number of applications as well as users [9]. It was developed by the Apache Software Foundation and released under the Apache License 2.0. JMeter is written in Java and can be

used to simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load types. JMeter also works for protocols like- Web (HTTP and HTTPS sites using Ajax, flex, and flex-ws-amf, Web services- XML-RPC/SOAP, Directory- LDAP, Database- JDBC drivers, JMS, Services- POP3, FTP, SMTP and IMAP [9]. JMeter can simulate the behavior of multiple users accessing the application simultaneously, and can measure response times, throughput, error rates, and other performance metrics.

JMeter is a highly configurable tool and supports a wide range of plugins and extensions, including custom plugins for specialized testing requirements. It also includes a scripting language that allows testers to create complex test scenarios that simulate real-world usage patterns. One of the primary benefits of JMeter is that it is a free and open-source tool, making it accessible to a wide range of users and organizations. It also has a large community of users and contributors who provide support and contribute to the development of the tool.

Another benefit of JMeter is its ease of use. The tool has a user-friendly interface that makes it easy to set up and run tests, even for users with limited technical expertise. JMeter also includes a range of pre-built templates and examples that can help users get started quickly. JMeter is also highly flexible and can be used in a wide range of testing scenarios, from basic functional testing to complex load testing and performance tuning. It can be integrated with other testing tools and frameworks and can be used in both manual and automated testing workflows.

Performance testing entails analyzing numerous server answers and delivering them to the client. As a sampling component of JMeter runs, listeners give a visual representation of the information obtained about those test cases by JMeter. JMeter provides many listeners and contains various formats. It makes it easier for the user to view sampler results in some log files as tables, graphs, trees, or plain text as shown in Fig. 4.9. In this research, we will focus on the listener graph and the listener table. Listener generally applicable for post process data. When we save data into the file and get results with a chart at that time Jmeter process many configuration options to add a data processing model and extra visualization [10].

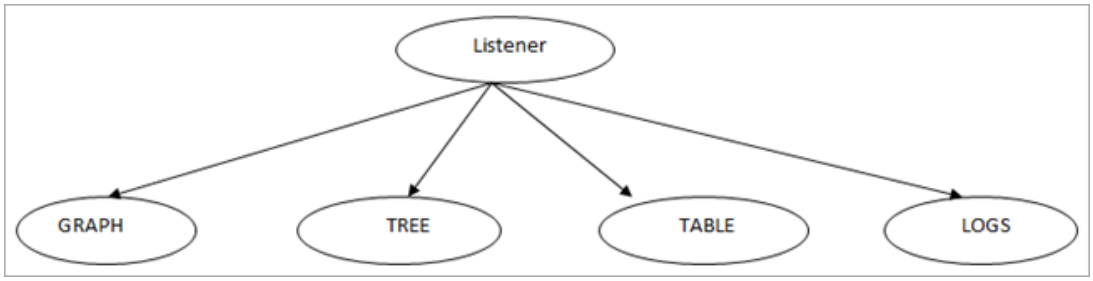


Fig. 4.9 JMeter Listeners Formats

Apache JMeter can load-test functional behavior and measure performance. The tool supports various protocols, including HTTP or HTTPS and provides a wide range of performance metrics. Apache JMeter was used to simulate HTTP requests to a web server deployed on each of the VMs. The web server was configured with the NGINX web server software. The number of samples that we tested with HTTP requests in this experiment is 500, 1000, 3000, 5000, and 10000 users respectively because of continuously increasing the load to measure how the server can handle it until overloaded or an error occurs. However, It is one of the important testing criteria in this testing where we can observe the system response, according to different traffic scenario such as low, medium and high traffic [10]. The comparison of load testing results between CSPs is described by Summary Report, Response Time graph, and Aggregate graph which are the most significant listeners in JMeter because they provide all necessary data.

CHAPTER 5

PERFORMANCE EVALUATION

5.1 PERFORMANCE METRICS SELECTION

In performance testing, it's essential to select appropriate metrics to measure the performance of the cloud services accurately. These performance metrics are used to evaluate different aspects of the cloud infrastructure, such as the response time, throughput, and resource utilization. In this section, we discuss the performance metrics selection for our research study, focusing on the critical performance metrics that we will be measuring for our chosen cloud service providers, namely Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS).

In our research study, we will be measuring the auto-scaling capabilities of each cloud service provider. Auto-scaling is a critical feature of cloud services that allows the infrastructure to automatically adjust its capacity based on demand. We will be measuring the auto-scaling capabilities of each cloud service provider to determine how well they can handle sudden spikes in user requests.

However, we will also be measuring the recovery time of each cloud service provider. Recovery time measures the time it takes for the infrastructure to recover from a failure. Recovery time is a critical performance metric because it directly affects the availability of the cloud service. A shorter recovery time means less downtime and a more reliable service.

One of the most important metrics for measuring cloud performance is response time. Response time measures the time it takes for a request to be processed and returned to the user. In cloud computing, response time is a critical performance metric because it directly affects the user experience. Longer response times can lead to frustrated users and potentially lost revenue. To measure the response time, we will be using Apache JMeter, which is a popular open-source load testing tool.

Another important performance metric is throughput, which measures the number of requests that can be processed in a given amount of time. Throughput is a vital performance metric for cloud services because it determines the capacity of the cloud infrastructure. In our research study, we will be measuring the throughput of the cloud services to determine their scalability and capacity to handle a large number of user requests.

In addition to the above metrics, we will also be measuring the error rate of the cloud services. Error rate measures the number of errors that occur during the testing period. This performance metric is critical because it determines the stability of the cloud infrastructure. Higher error rates can indicate instability in the cloud infrastructure, which can lead to potential downtime and lost revenue.

Selecting the right performance metrics is essential for accurately measuring the performance of cloud services. By measuring the auto-scaling capabilities, and recovery time, and response time, throughput, error rate we can gain a comprehensive understanding of the performance of each cloud service provider. These performance metrics will help us compare the cloud service providers and determine the best cloud service for our application.

5.2 GOOGLE COMPUTE ENGINE IMPLEMENTATION

Performance evaluation operational process of Google Compute Engine, starting with creating an instance group in Google Cloud is a process that can be completed in just a few steps. First, log in to our Google Cloud Console and navigate to the "Compute Engine" section. From there, select "Instance groups" and click on the "Create instance group" button. We will be prompted to choose the type of instance group we want to create which is "Managed instance group. Next, we will need to provide a name which is “gcp-ig” and specify the machine type and boot disk which is the baseline experimental virtual machine configuration that had the installation of Ubuntu OS and Nginx web server which saved to the custom image

namely “instance-template-custom-image” then we set the region for our instance group which is “asia-south1 (Mumbai)” as shown in Fig. 5.1.

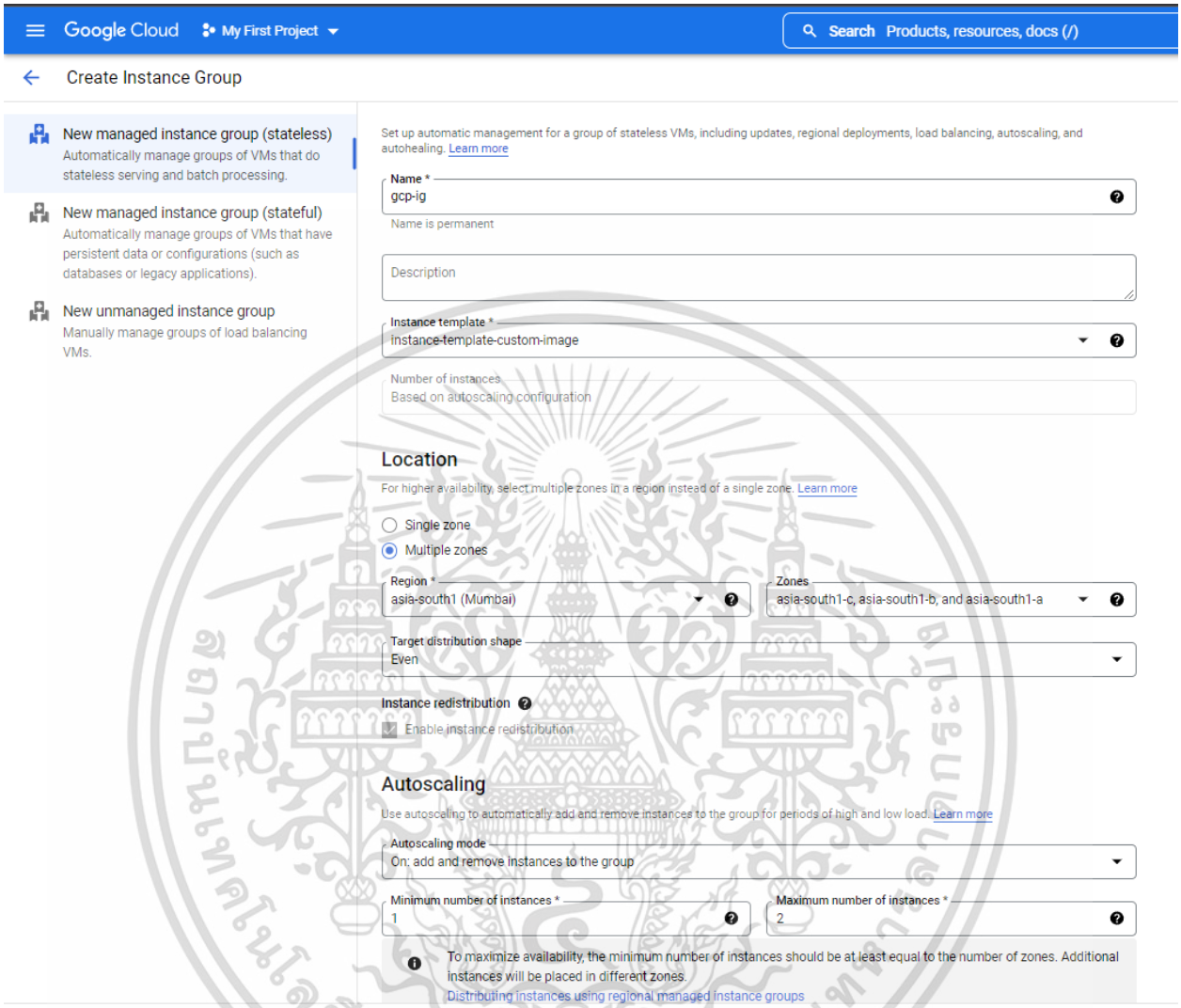


Fig. 5.1 Create instance group on GCP

Following this, specify the number of instances we want to create which the minimum number of instances is 1 and the maximum number of instances is 2 along with the option for “Autoscaling” to enable automatic scaling of our instance group based on demand, and configure the scaling policies as per our requirements. In the Autoscaling metrics section, leave almost the setting to default such as Autoscaling schedules and Autohealing for this cloud default behavior except for the scaling policies that we focusing which we add the CPU utilization and set to 60% as shown in Fig. 5.2. Finally, review our settings and click “Create”

to create our instance group. Once created, we can also configure additional settings like load balancing if desired.

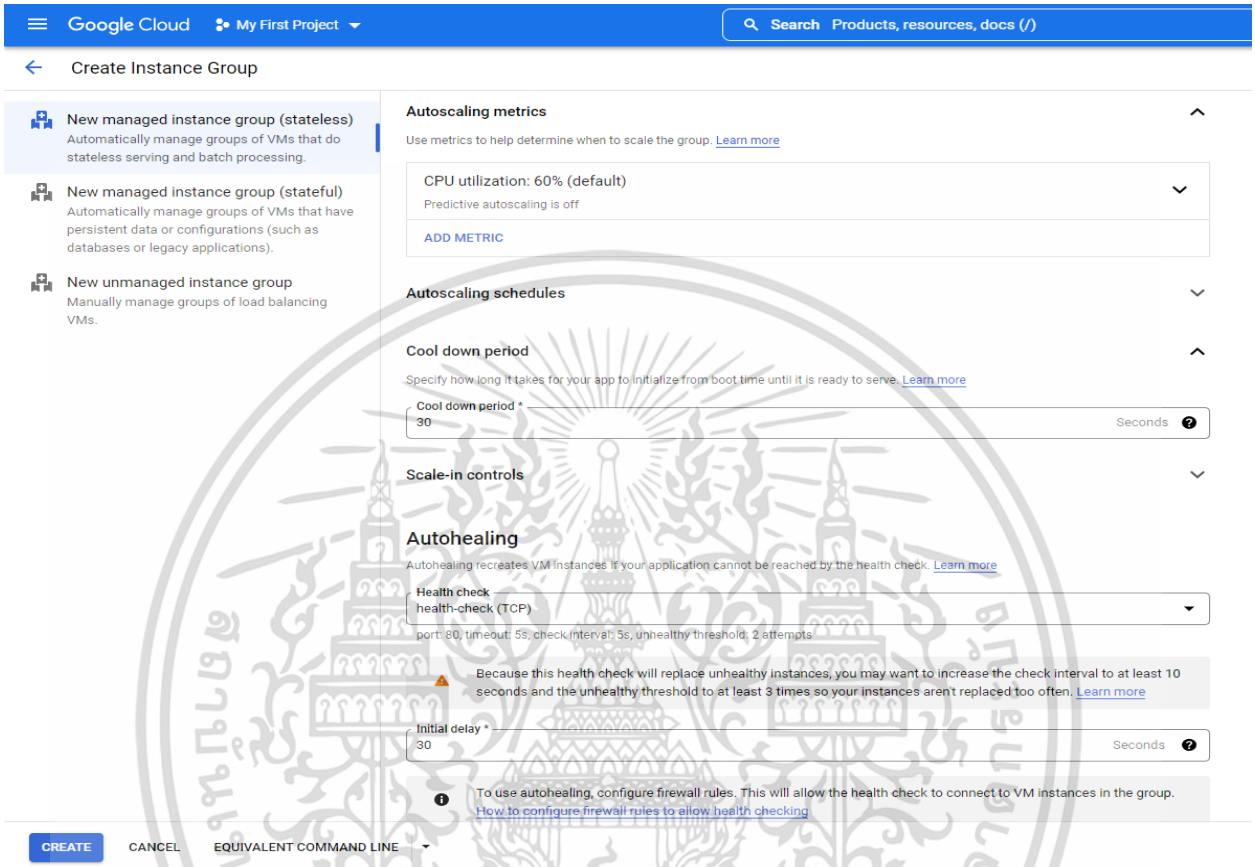


Fig. 5.2 Setup autoscaling metrics on GCP

Adding load balancing to our Google Cloud environment can help distribute traffic evenly across our instances, improving performance and ensuring high availability. To add load balancing in Google Cloud, start by selecting the appropriate type of load balancer for our application, such as HTTP or TCP. Then, configure the load balancer's settings, including the backends, frontend IP address, and protocols. Once our load balancer is configured, we can associate it with our instance group by specifying the instance group as the backend service for the load balancer. We can also configure the load balancer to use a health check to monitor the instances in our instance group and remove any instances that are not responding. Finally, review and finalize then select create load balancer as show in Fig. 5.3 to ensure that

it is distributing traffic evenly across our instances. After that, we can review our load balancing details and able to edit the setting again as show in Fig. 5.4.

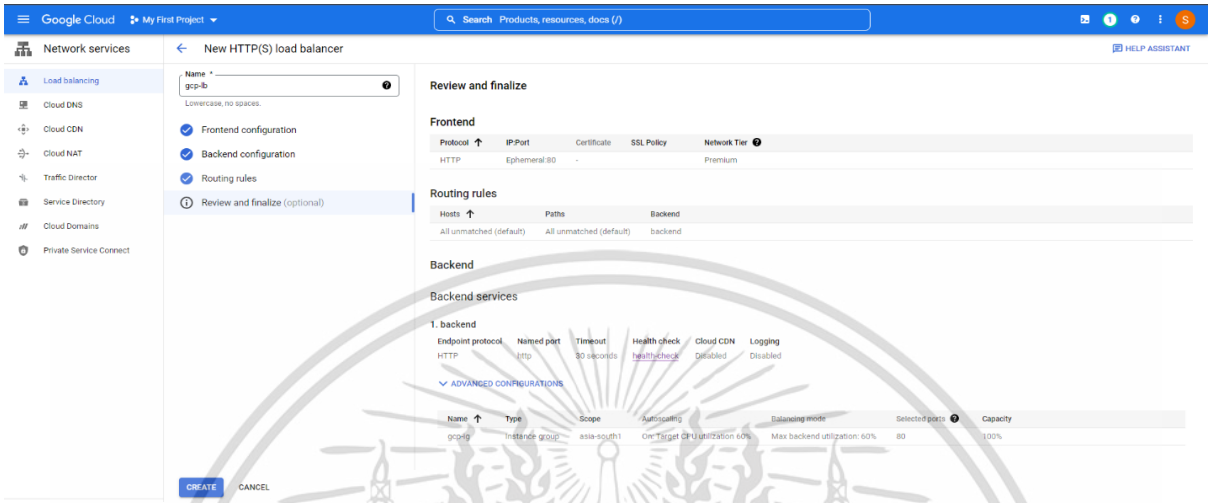


Fig. 5.3 Configure load balancer on GCP

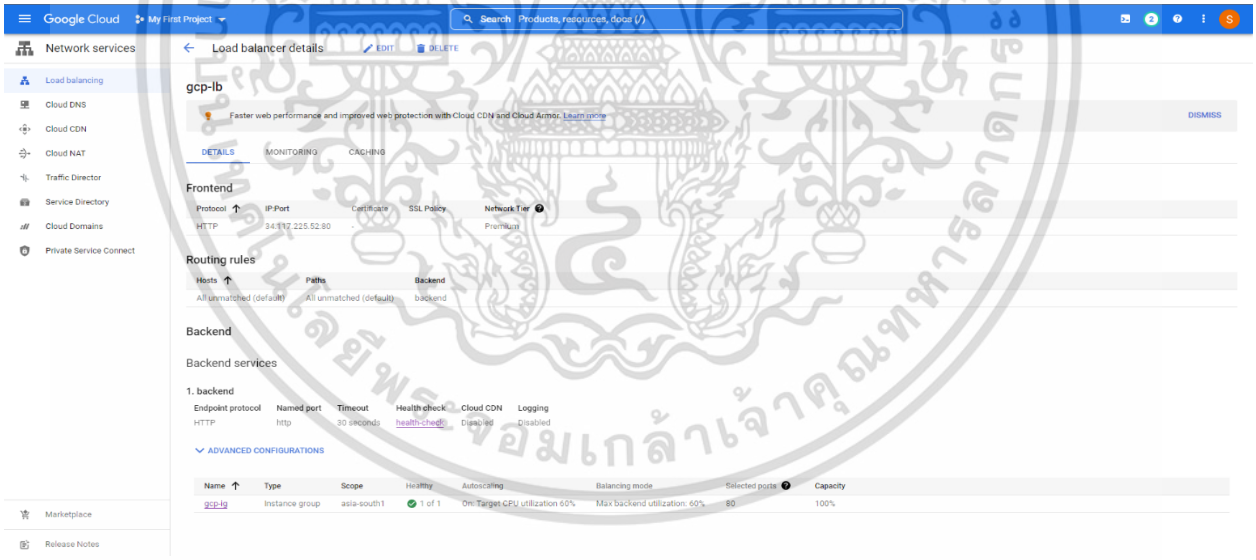


Fig. 5.4 Load balancer details on GCP

As shown in Fig. 5.5, we can review the instances that we created earlier in the Instance groups of Compute Engine. It can be seen that the policy that we defined is working as starting with the minimum of one instance created and provided information such as Status, Creation Time, Autoscaling, and IP address which is “34.93.225.107”. However, we will try to test that

deployed web server is usable by connecting to the External IP with the HTTP request method on a browser as shown in Fig. 5.6.

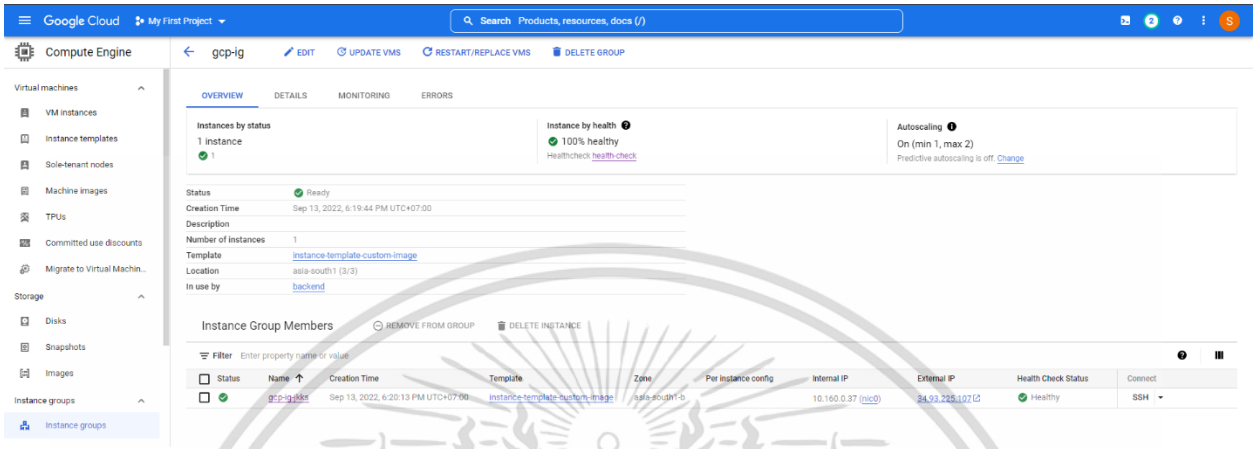


Fig. 5.5 Review instance groups on GCP

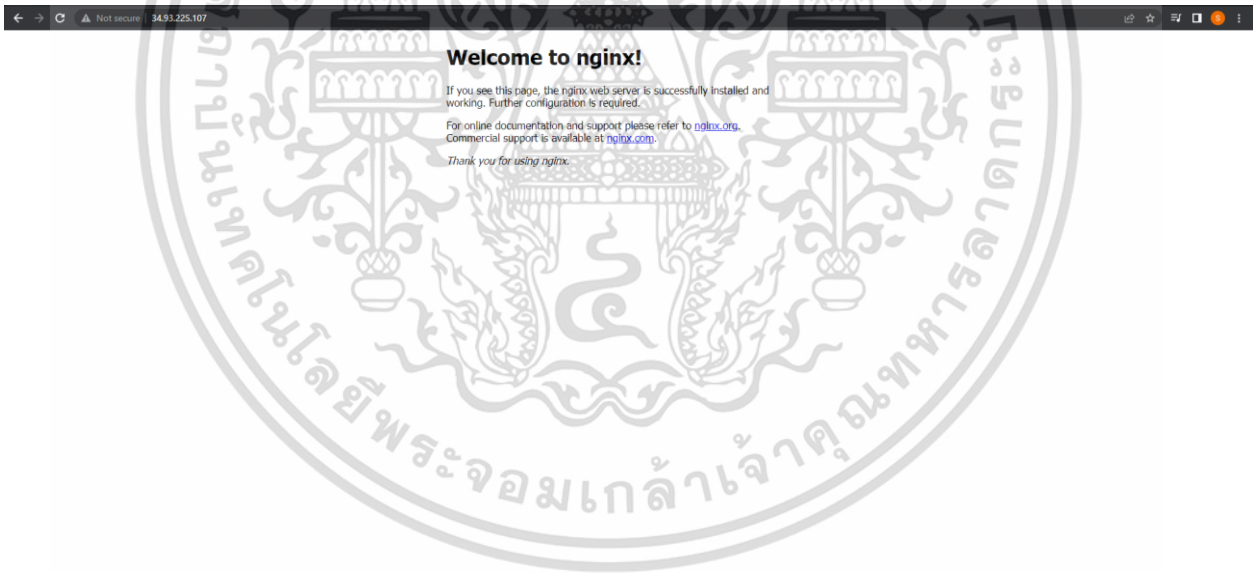


Fig. 5.6 Nginx web server interface of GCP

Google Compute Engine allows us to connect to the virtual machine by providing Google SSH-in-browser rather than connecting in the traditional way of opening the local terminal which makes us more convenient and faster. Therefore, we use this feature to SSH

requests to our virtual machine by just clicking connect and the terminal will pop up on the browser view as shown in Fig. 5.7.

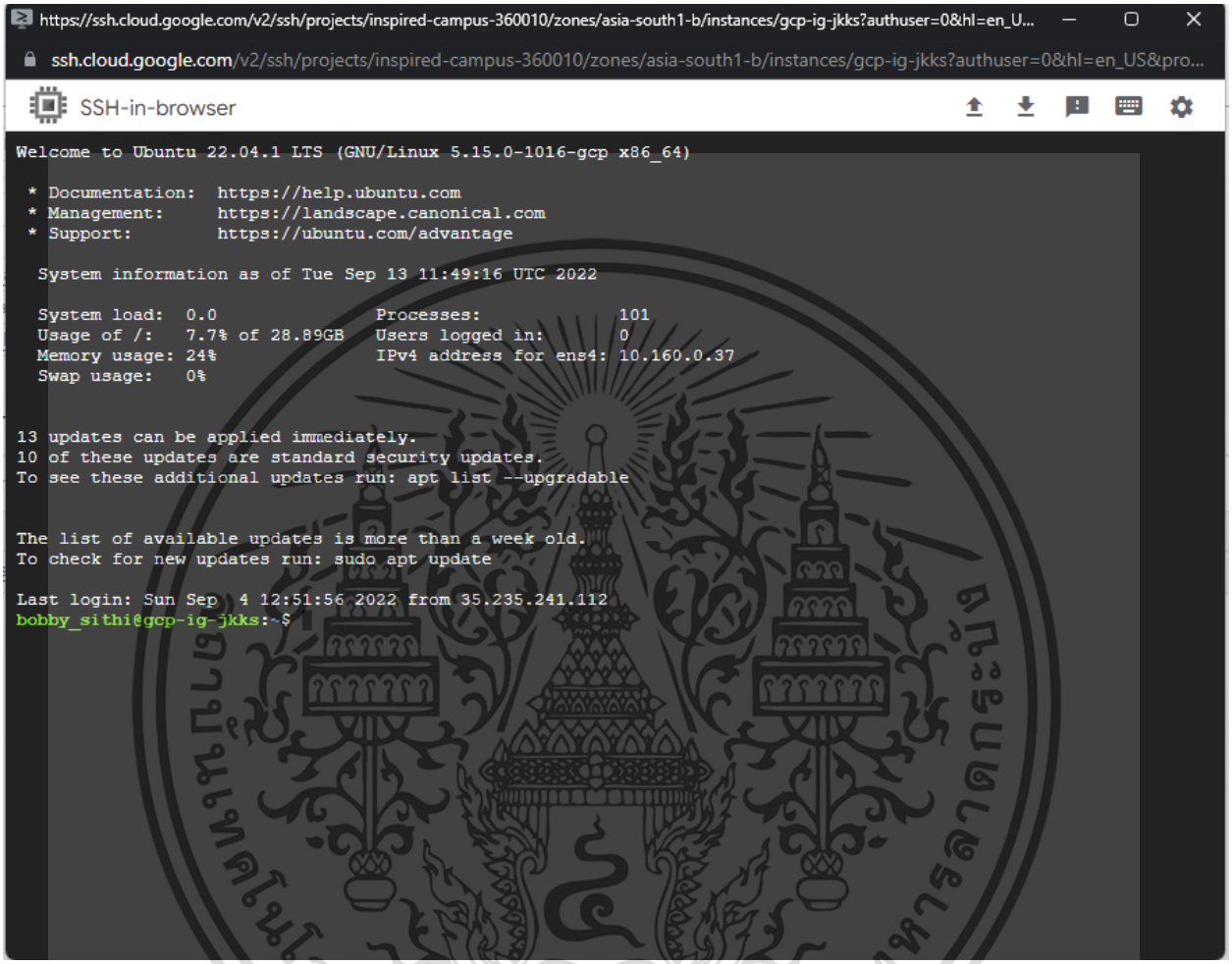


Fig. 5.7 Connect to instance via SSH of GCP

After we connected to our instance, we are going to test the scalability of the Autoscaling of Google Compute Engine. We will generate artificial loads by starting to type “sudo apt-get -y install stress” as shown in Fig. 5.8 then “stress –cpu 9 –timeout 300”, “stress –cpu 9 –timeout 600”, and “stress –cpu 9 –timeout 900” respectively as shown in Fig. 5.9. This will make the CPU utilization exceed 60% and let’s look back on the Google Compute Engine interface to monitor its behavior.

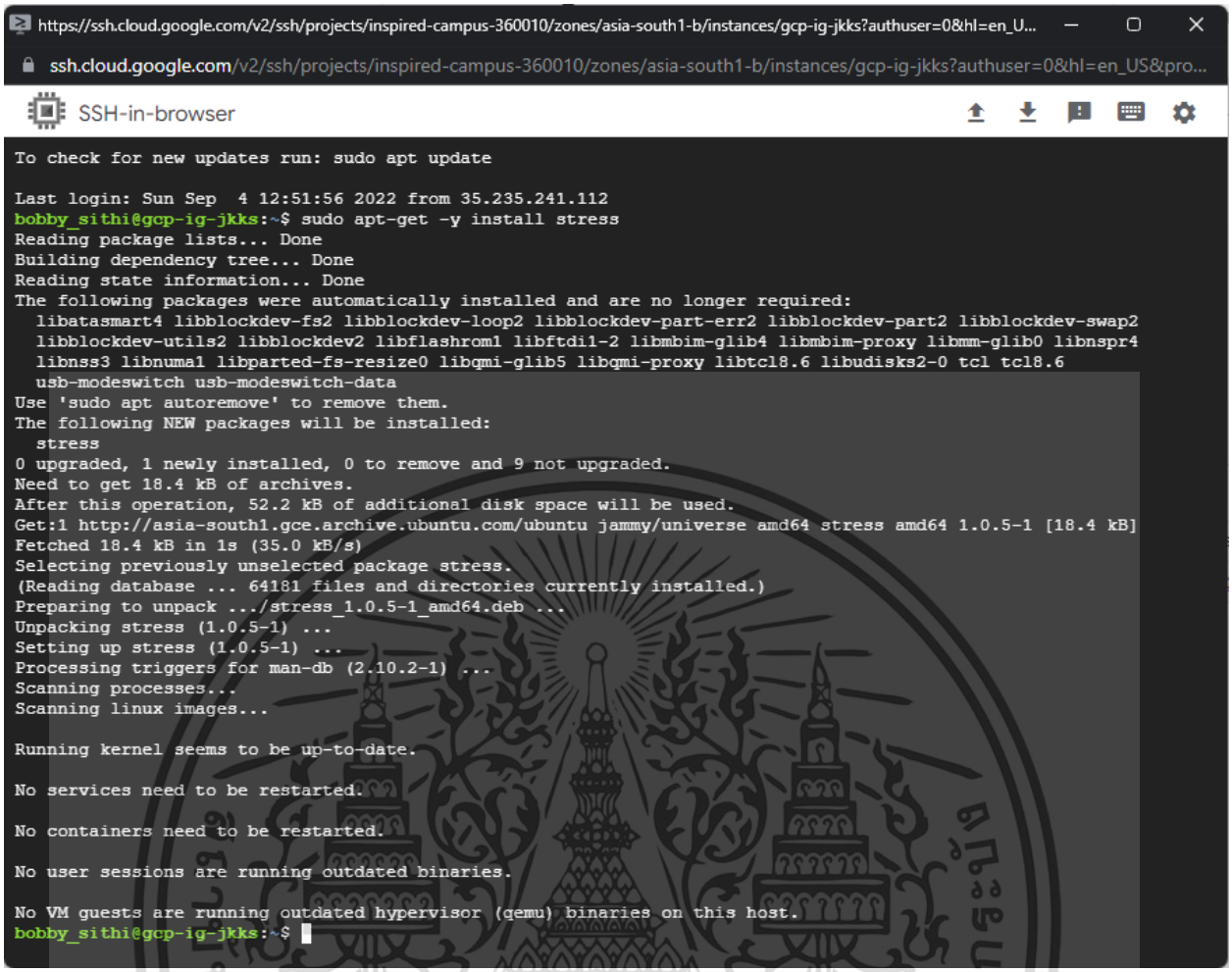


Fig. 5.8 Install stress via SSH of GCP

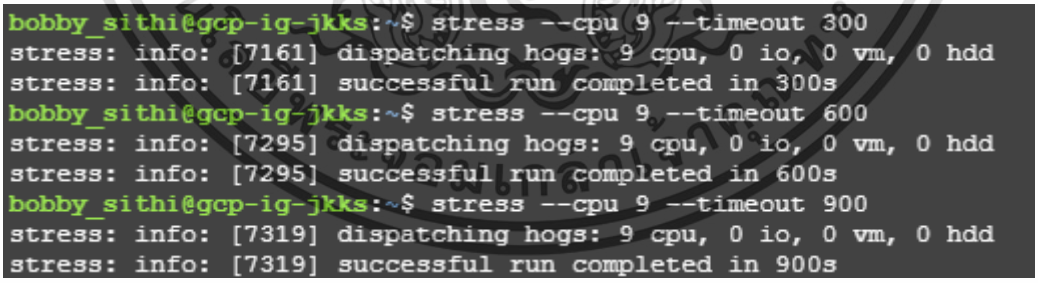


Fig. 5.9 Execute stress command via SSH of GCP

As we can see in Fig. 5.11, the instance group members are updating one new instance IP “34.93.247.217” so this is accurate as the Autoscaling policy that we defined. We will provide and analyze the monitoring result in the scalability testing section to ensure that the stress loads actually exceed CPU utilization of 60%.

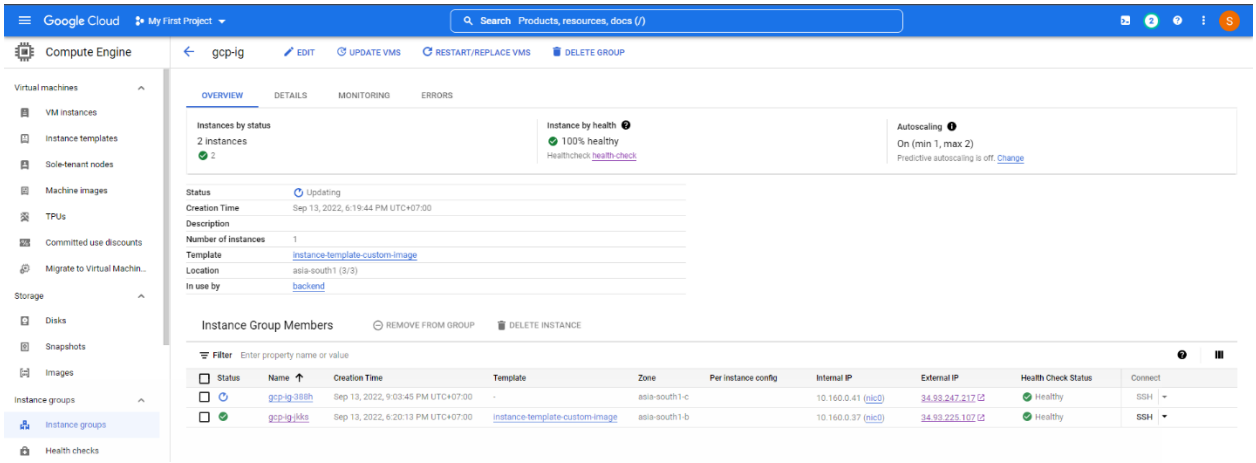


Fig. 5.11 Instance group members on GCP

Next, we will simulate server crashing by stopping the web server service which type the command “sudo systemctl stop nginx” then we will type the command “sudo systemctl status nginx” to collect down time and ensure that the web server is inactive as shown in Fig 5.12.

```
bobby_sithi@gcp-ig-jkks:~$ sudo systemctl stop nginx
bobby_sithi@gcp-ig-jkks:~$ sudo systemctl status nginx
o nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: inactive (dead) since Tue 2022-09-13 22:04:29 UTC; 8s ago
       Docs: man:nginx(8)
    Process: 8283 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.pid (cod>
   Main PID: 571 (code=exited, status=0/SUCCESS)
        CPU: 2.777s

Sep 13 11:20:47 gcp-ig-jkks systemd[1]: Starting A high performance web server and a reverse proxy server...
Sep 13 11:20:48 gcp-ig-jkks systemd[1]: Started A high performance web server and a reverse proxy server.
Sep 13 22:04:29 gcp-ig-jkks systemd[1]: Stopping A high performance web server and a reverse proxy server...
Sep 13 22:04:29 gcp-ig-jkks systemd[1]: nginx.service: Deactivated successfully.
Sep 13 22:04:29 gcp-ig-jkks systemd[1]: Stopped A high performance web server and a reverse proxy server.
Sep 13 22:04:29 gcp-ig-jkks systemd[1]: nginx.service: Consumed 2.777s CPU time.
lines 1-14/14 (END)
```

Fig. 5.12 Stop nginx web server via SSH of GCP

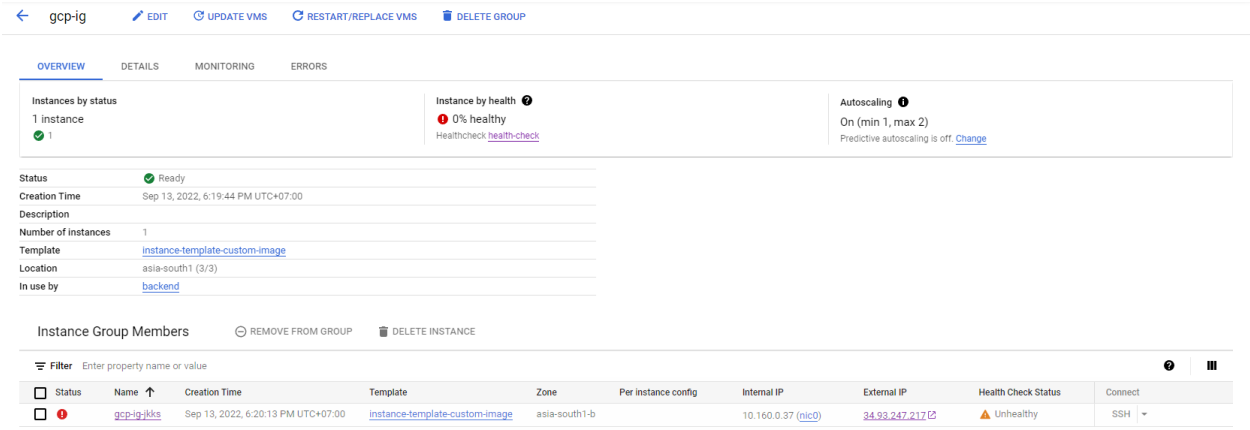


Fig. 5.13 Instance heath of GCP

Finally, recheck the instance by health as shown in Fig. 5.13 which is 0% healthy means the virtual machine status is unhealthy so we will wait until the Google Compute Engine handles this issue by terminating the unhealthy virtual machine and recreating another based on the policy that we defined whether a minimum of one instance with the same default custom image template and collect the creation time to compare both of instance down time and the new instance creation time for getting recovery time.

5.3 AZURE VIRTUAL MACHINES IMPLEMENTATION

Performance evaluation operational process of Azure for creating virtual machines in Microsoft Azure is a process that can be done in several steps. First navigate to the Azure portal and select the "Virtual Machines" section. From there, select the virtual machine that we want to use as the basis for our image. We set the region for our virtual machine which is “Central India” and install web server of “Nginx”. Once we have selected the virtual machine, click on the "Capture" button to begin the image creation process. This will open a new window where we can provide a name and description for our image, as well as specify the storage account and container where the image will be stored. We can also configure any additional settings, such as whether to generalize the image or capture it with the virtual machine running. Once we have configured our settings, click on the "Create" button to create our image as shown in Fig. 5.14. The image will be stored in the specified storage account

which is “Azure Subscription 1” in our experiment and can be used to deploy and scale new virtual machines quickly and easily.

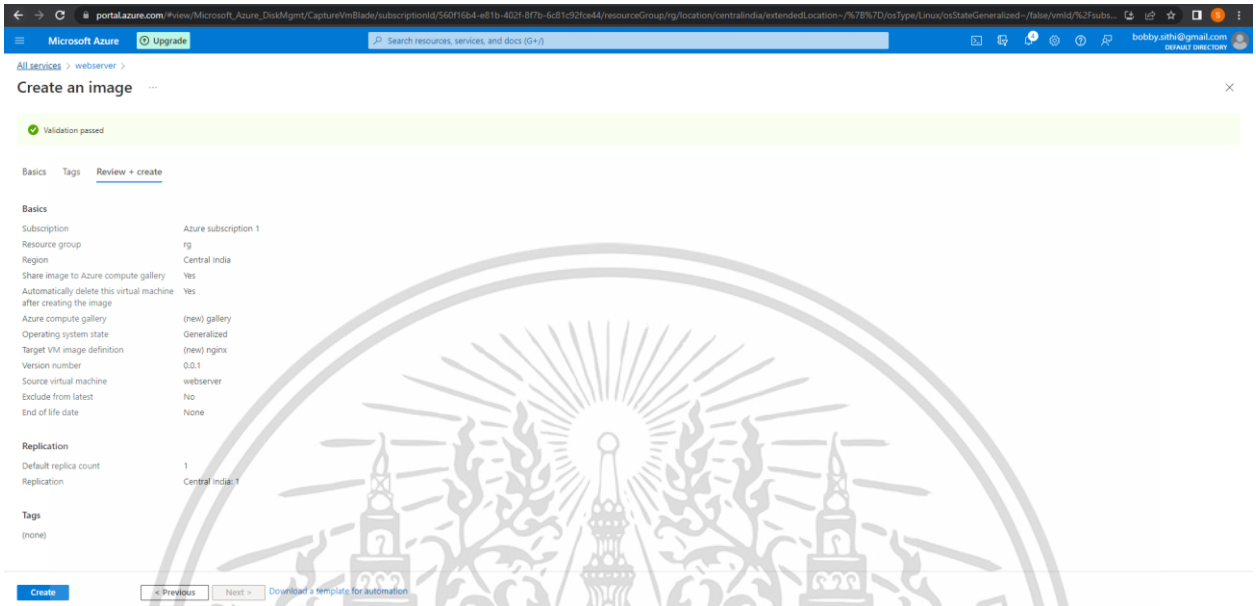


Fig. 5.14 Create an image on Azure

A virtual machine scale set in Microsoft Azure is a group of identical virtual machines that can automatically scale up or down based on demand. To create a virtual machine scale set, we start by navigating to the Azure portal and selecting the "Virtual Machine Scale Sets" section. From there, we click on the "Add" button to begin creating our scale set. This will open a new window where we can specify the basic settings for our scale set, such as the region, and operating system as shown in Fig. 5.15. Next, we can configure the virtual machine settings, such as the size, image settings. We use the custom image that we created earlier for the baseline. We can also configure the scaling settings, such as the minimum and maximum number of instances, as well as the scale-out and scale-in rules as shown in Fig. 5.16. We set the minimum number of instances is 1 and the maximum number of instances is 2. In the scaling section, we select scaling CPU threshold between 60% which scale out is 61% and scale in is 59%. Once we have configured our settings, we click on the "Review + create" button to review our virtual machine scale set settings and create our scale set. Azure will then create the virtual machines in the scale set, and we can manage and monitor the scale

set from the Azure portal, making it easy to scale and configure our virtual machines to meet our changing needs.

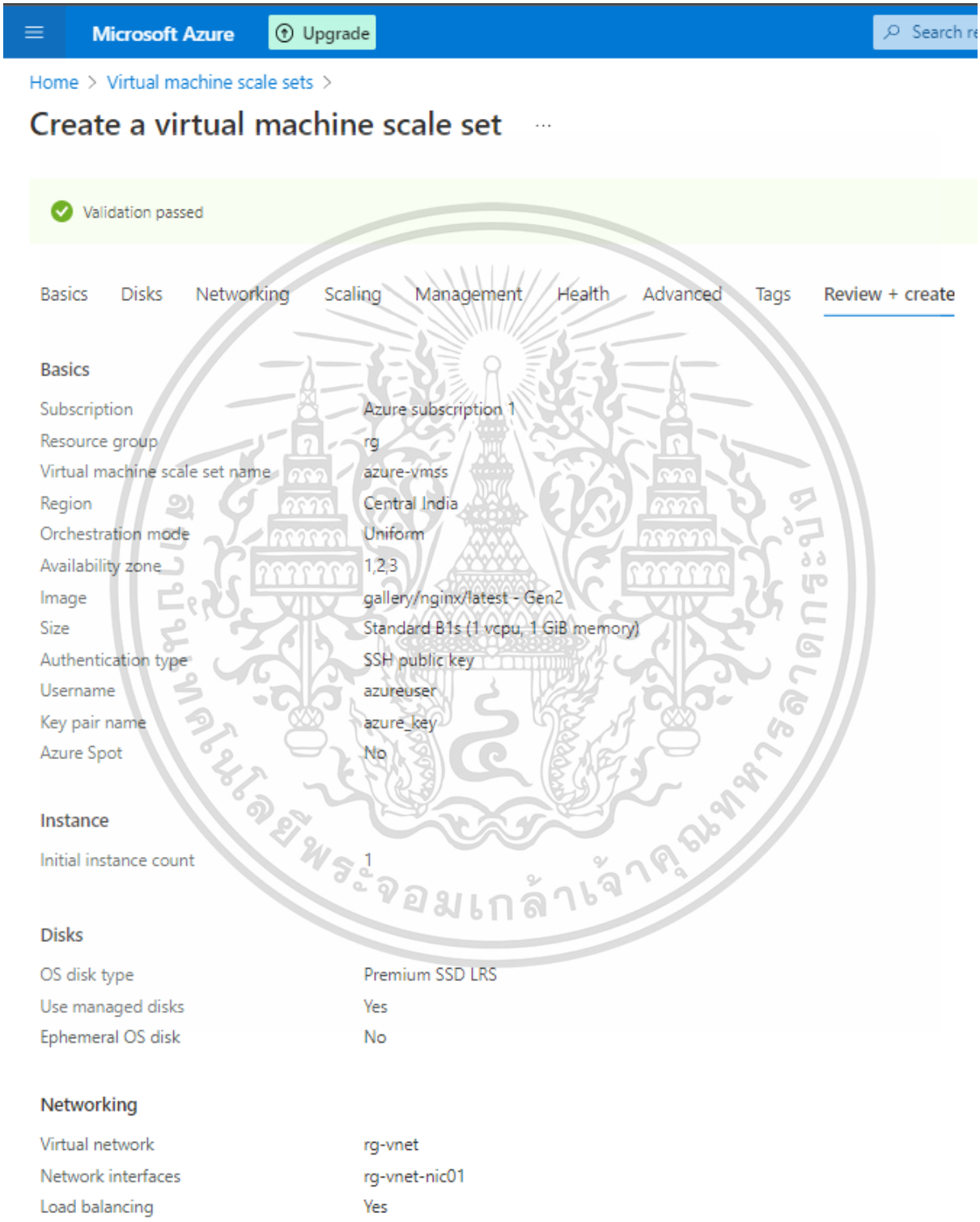


Fig. 5.15 Create virtual machine scale set on Azure

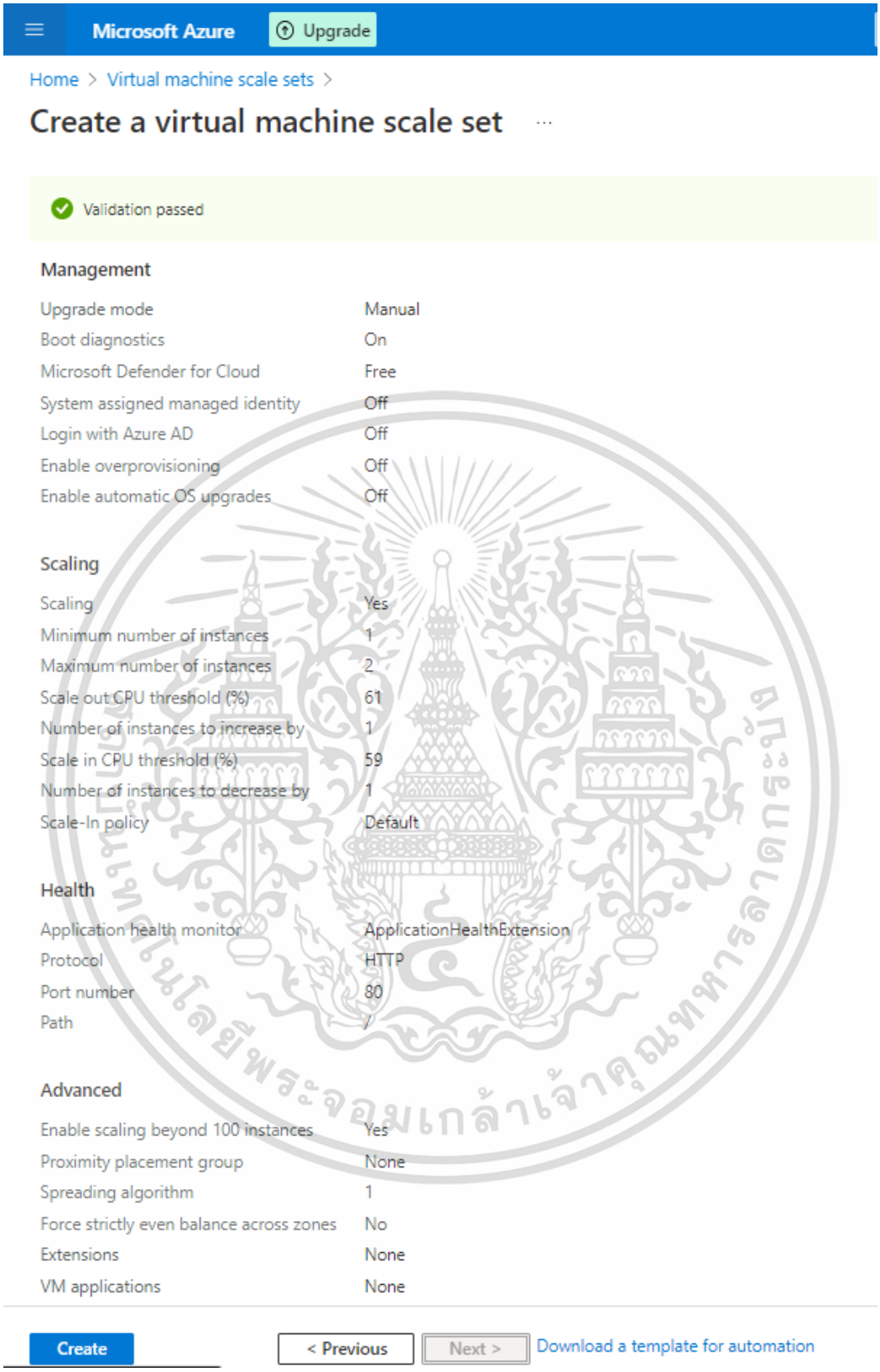


Fig. 5.16 Configure virtual machine settings on Azure

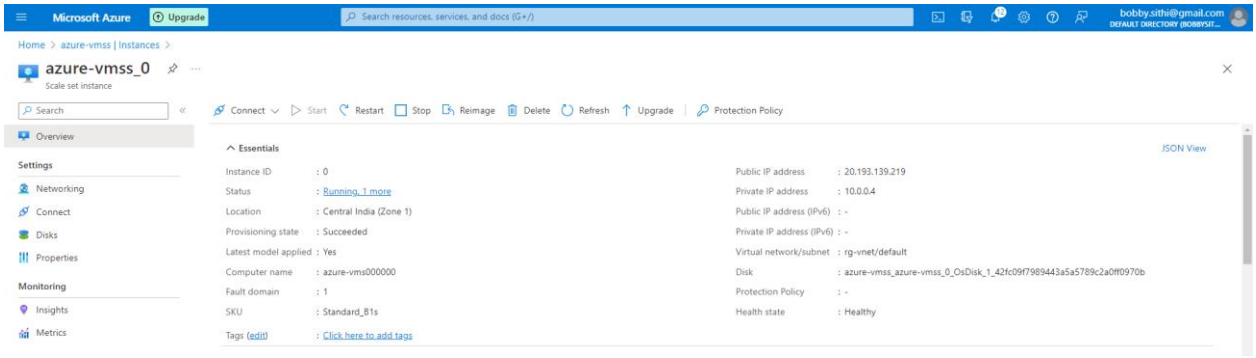


Fig. 5.17 Review instance details on Azure

As shown in Fig. 5.17, we can review the instances that we created in the virtual machine scale set of Azure. It can be seen that the policy that we defined is working as starting with the minimum of one instance created and provided information which name and public IP address are “azure-vmss_0” and “20.193.139.219” respectively. However, we will try to test that deployed web server is usable by connecting to the Public IP with the HTTP request method on a browser as shown in Fig. 5.18.

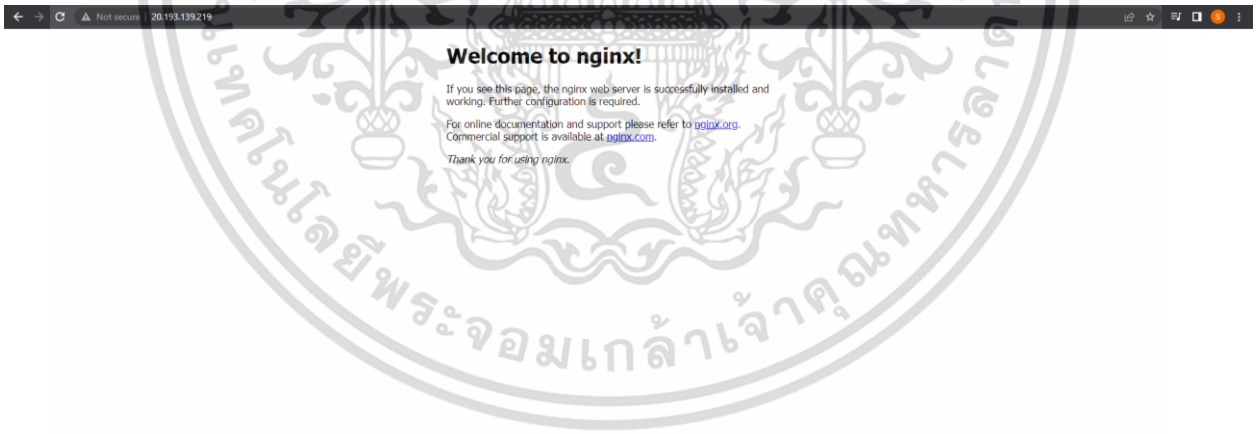


Fig. 5.18. Nginx web server interface of Azure

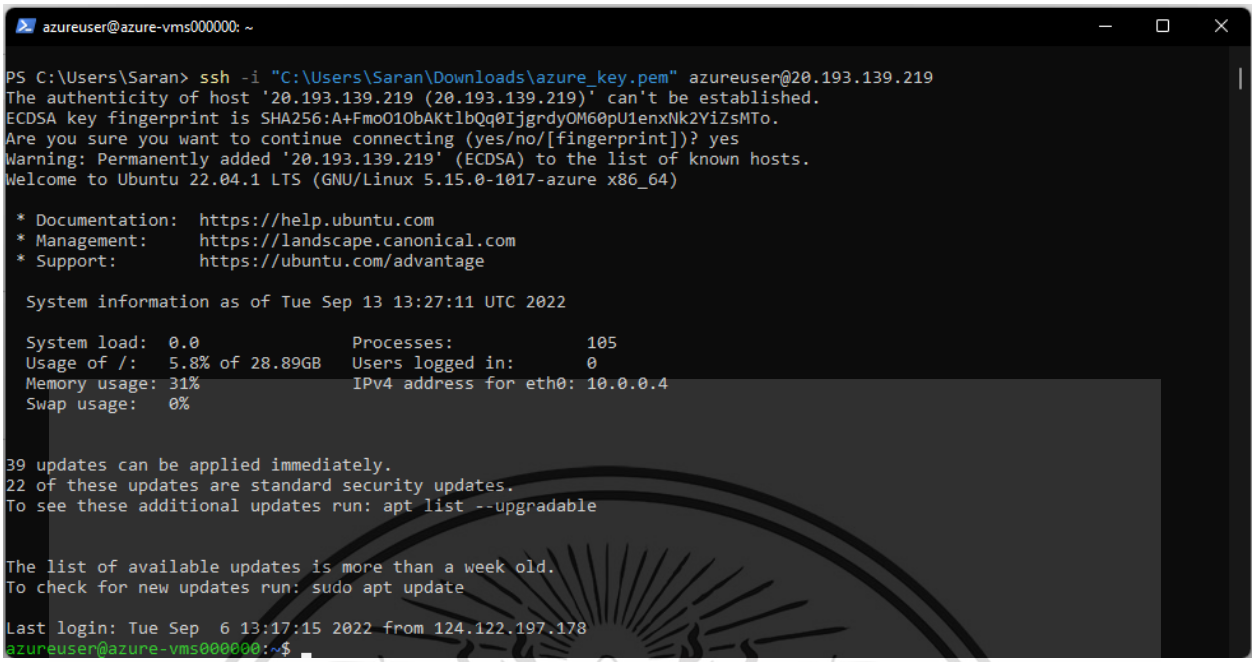


Fig. 5.19 Connect to instance via SSH of Azure

Once we connected via SSH to our instance in power shell terminal as shown in Fig. 5.19, we are going to test the scalability of the Autoscaling of Azure. We will generate artificial loads as shown in Fig. 5.20 and Fig. 5.21 which same as when we install stress and generate loads on Google instance. This will make the CPU utilization exceed 60% and let's look back on the azure-vmss interface to monitor its behavior.

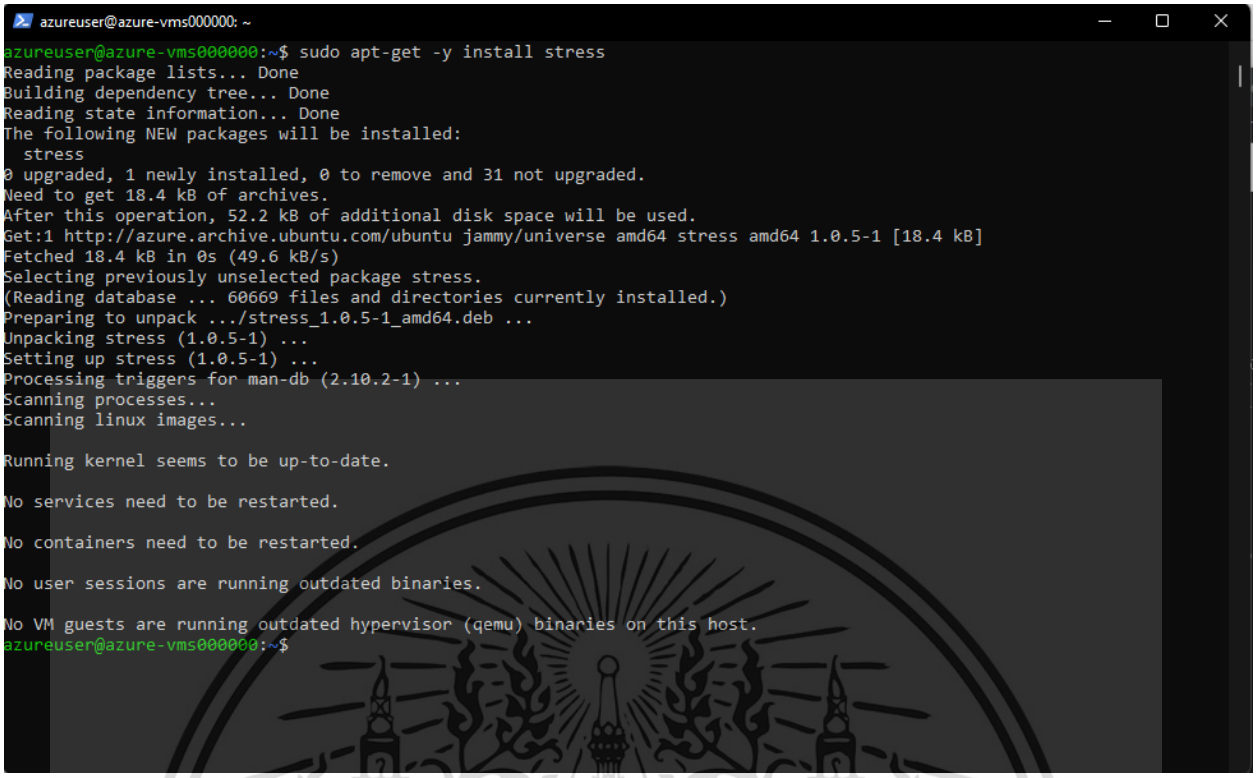


Fig. 5.20 Install stress via SSH of Azure

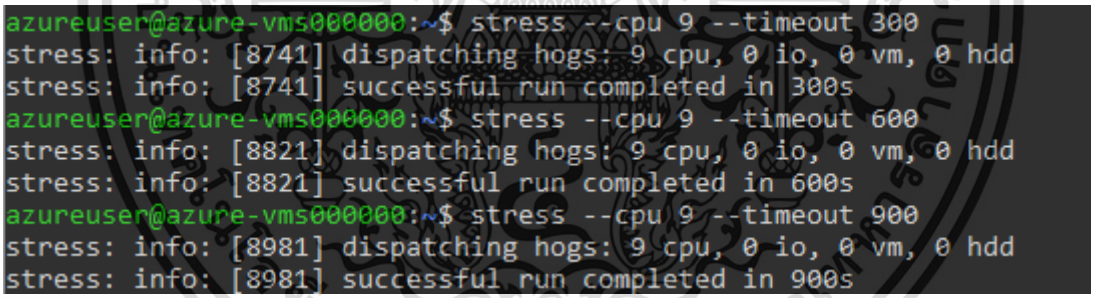


Fig. 5.21 Execute stress command via SSH of Azure

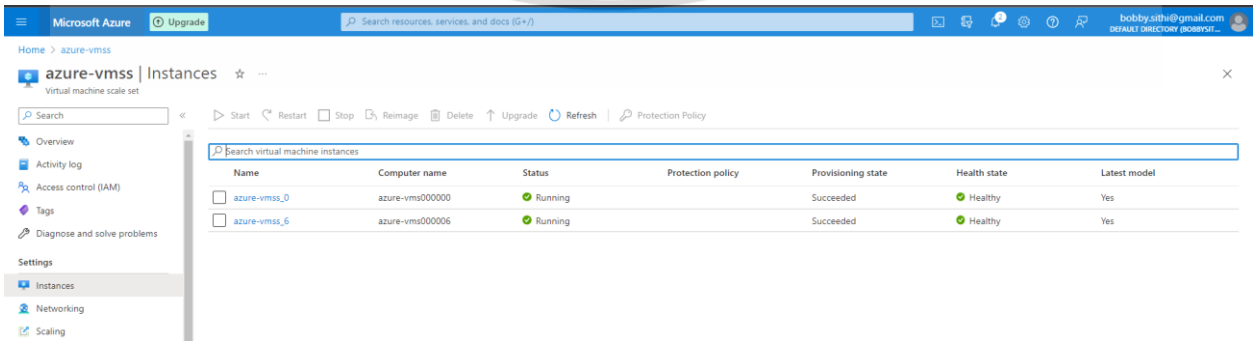


Fig. 5.22 Review instances in azure-vmss

As we can see in Fig. 5.22, the azure-vmss are updating one new instance “azure-vmss_6” so this is accurate as the Autoscaling policy that we defined. We will provide and analyze the monitoring result in the scalability testing section to ensure that the stress loads actually exceed CPU utilization of 60%.

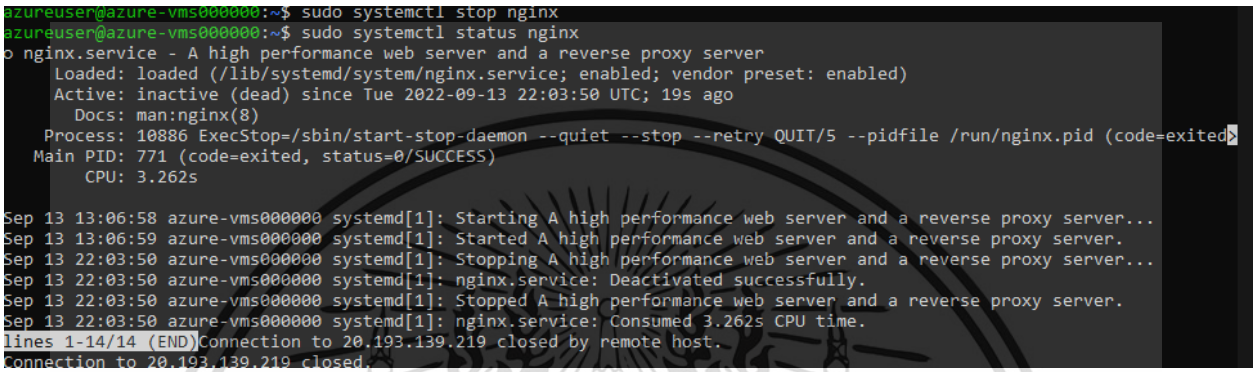


Fig. 5.23 Stop nginx web server via SSH of Azure

Fig. 5.24 Recheck the instance health on Azure

Moving to simulate server crashing process, stopping the web server service which type the command “sudo systemctl stop nginx” then we will type the command “sudo systemctl status nginx” to collect down time and ensure that the web server is inactive as shown in Fig 5.23. Finally, recheck the instance by health as shown in Fig. 5.24 which is 0% healthy means the virtual machine status is unhealthy

5.4 AWS EC2 IMPLEMENTATION

Performance evaluation operational process of AWS EC2 for creating an auto scaling group in AWS EC2 is a process that we can do in just a few steps. First, we navigate to the EC2 console and select the "Auto Scaling Groups" section. From there, we click on the "Create

Auto Scaling Group" button to begin creating our group. This will open a new window as shown in Fig. 5.25 and Fig. 5.26 where we can select the launch template or launch configuration for our instances which is our baseline custom image with Ubuntu OS and Nginx web server, as well as we can specify the size and configuration of our auto scaling group which is the minimum of 1 and maximum of 2 and enable load balancing. We can also configure scaling policies, which determine how our group scales in and out based on CPU Utilization metric of 60%. Once we've configured our settings, we click on the "Create Auto Scaling Group" button to review our auto scaling group settings and create our group. AWS will then create the instances in the group and automatically scale them up or down based on demand. We can also manage and monitor our auto scaling group from the EC2 console, making it easy to adjust our scaling policies and configure our instances to meet our changing needs.

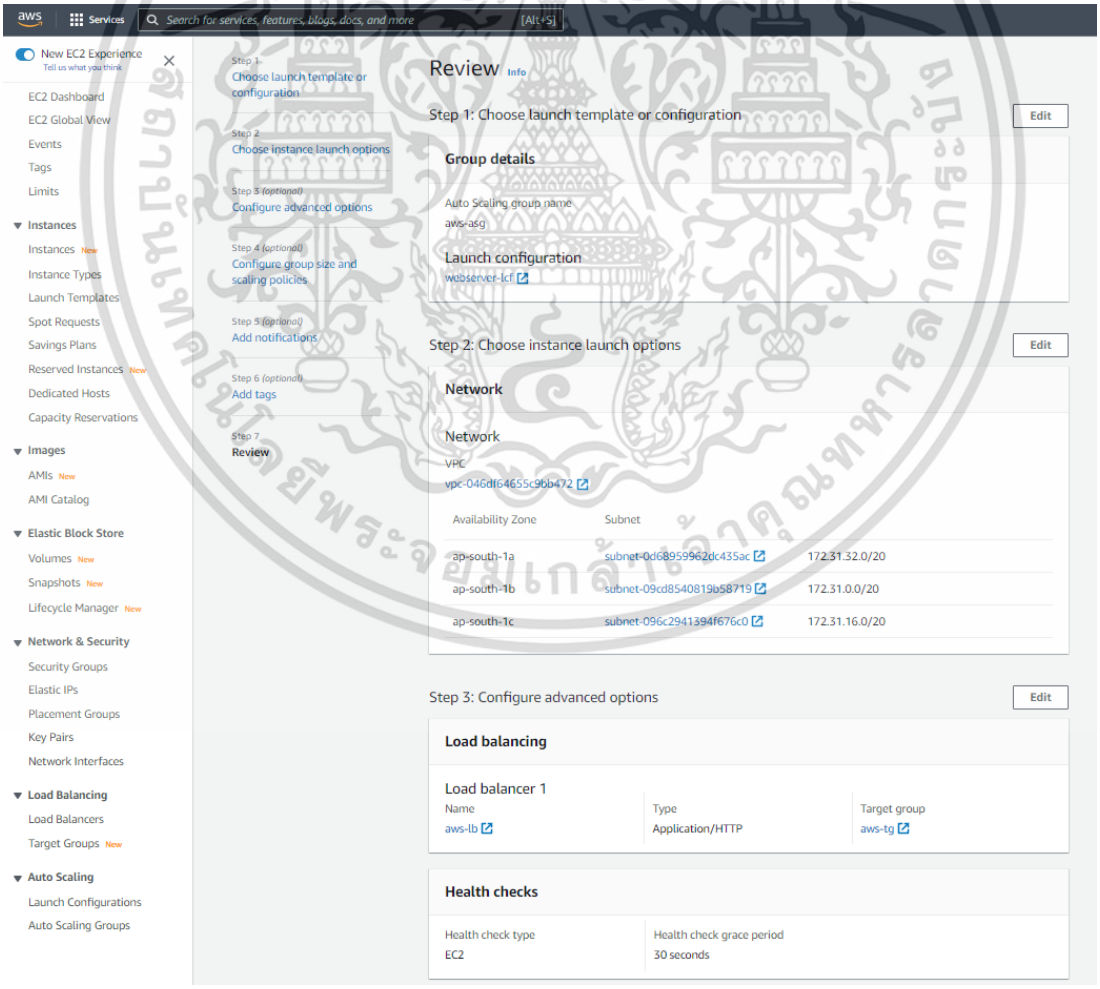


Fig. 5.25 Configure instance on AWS

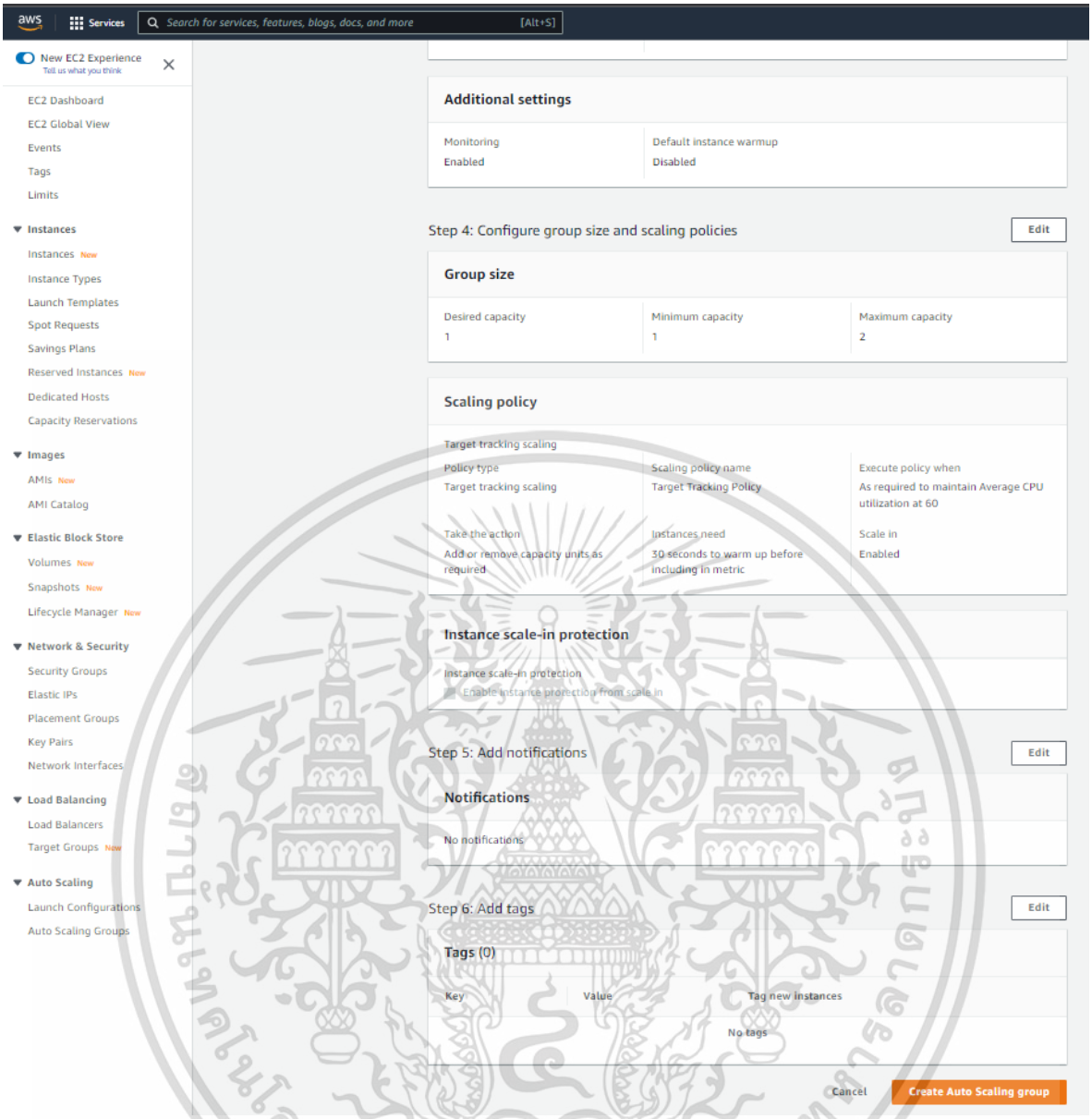


Fig. 5.26 Create Auto scaling group on AWS

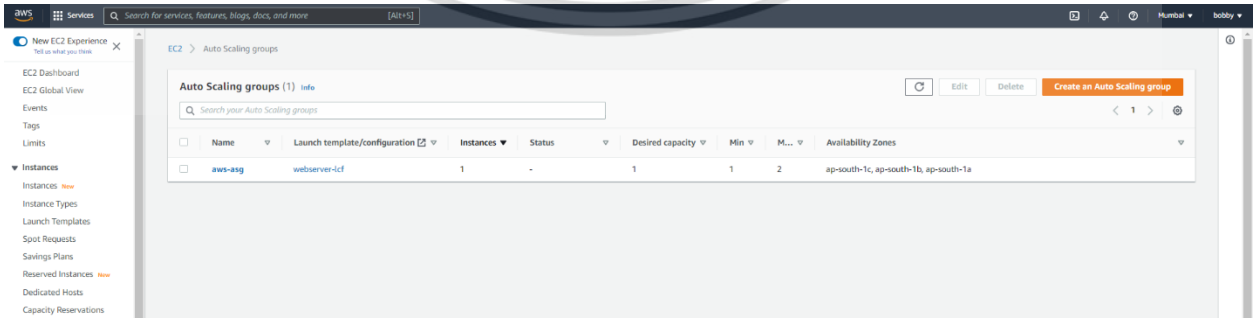


Fig. 5.27 Review the Auto Scaling groups on AWS

As shown in Fig. 5.27, we can review the instances that we created in the Auto Scaling groups of AWS EC2. It can be seen that the policy that we defined is working as starting with the minimum of one instance created. In Fig. 5.28, we can view our instance in the Instances section which provided information such as public IP address “3.110.196.227”. However, we will try to test that deployed web server is usable by connecting to the Public IP with the HTTP request method on a browser as shown in Fig. 5.29.

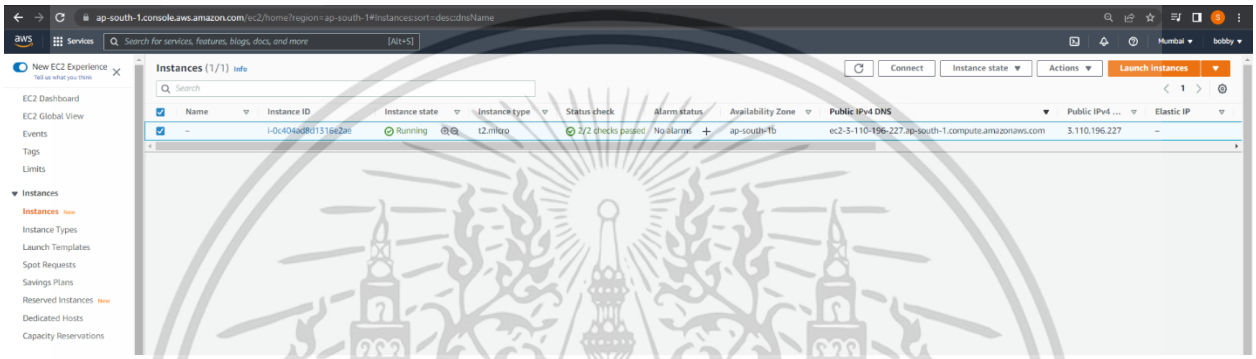


Fig. 5.28 View instances on AWS

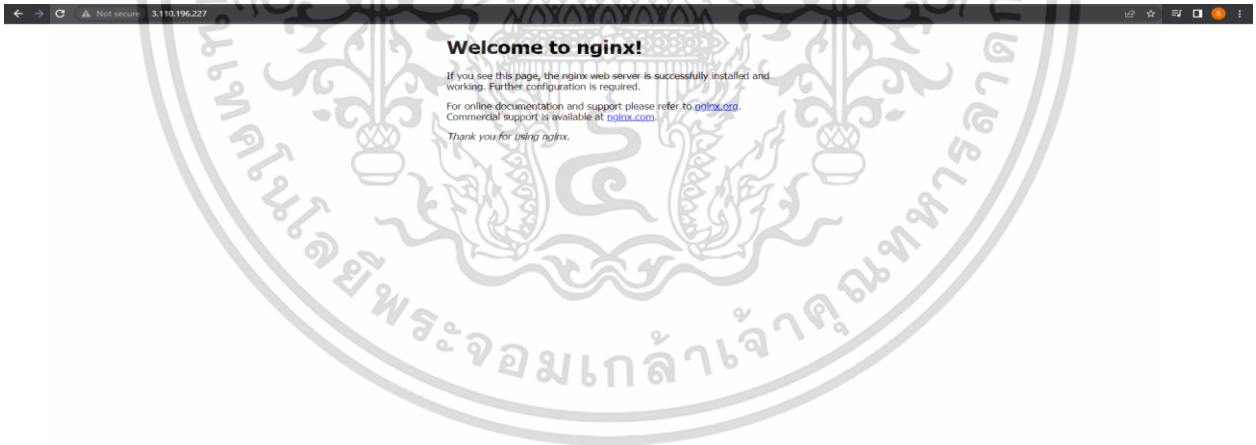


Fig. 5.29 Nginx web server interface of AWS

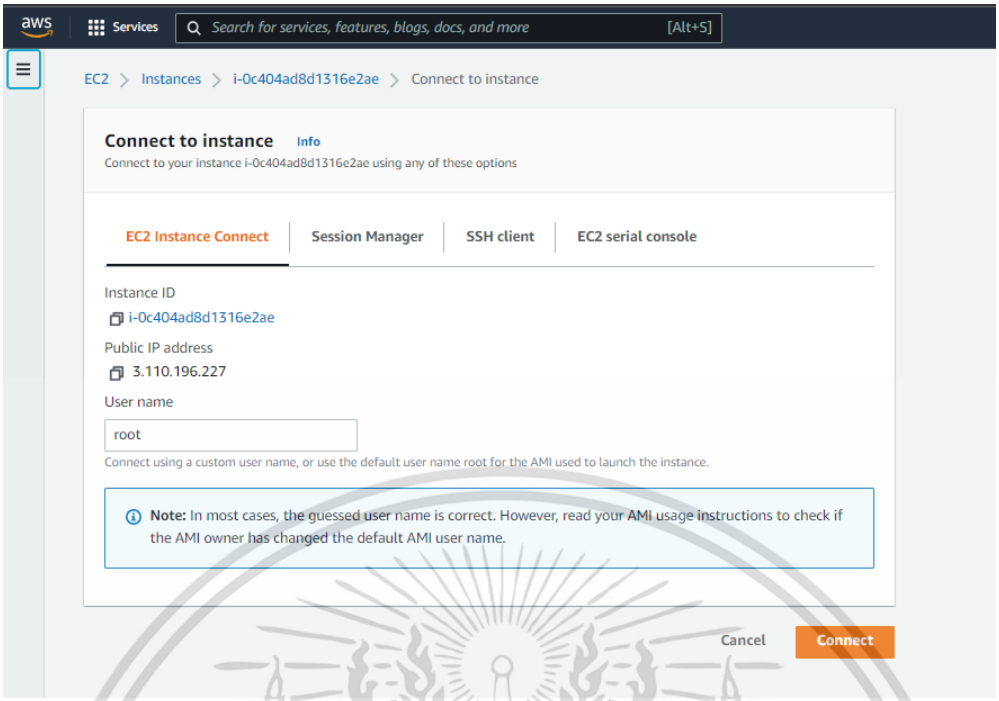


Fig. 5.30 Connect to instance on AWS browser

AWS EC2 allows us to connect to the virtual machine via browser same as Google Compute Engine. Therefore, we use this feature to SSH requests to our virtual machine by just clicking connect as shown in Fig. 5.30 and the terminal will pop up on the browser view as shown in Fig. 5.31.

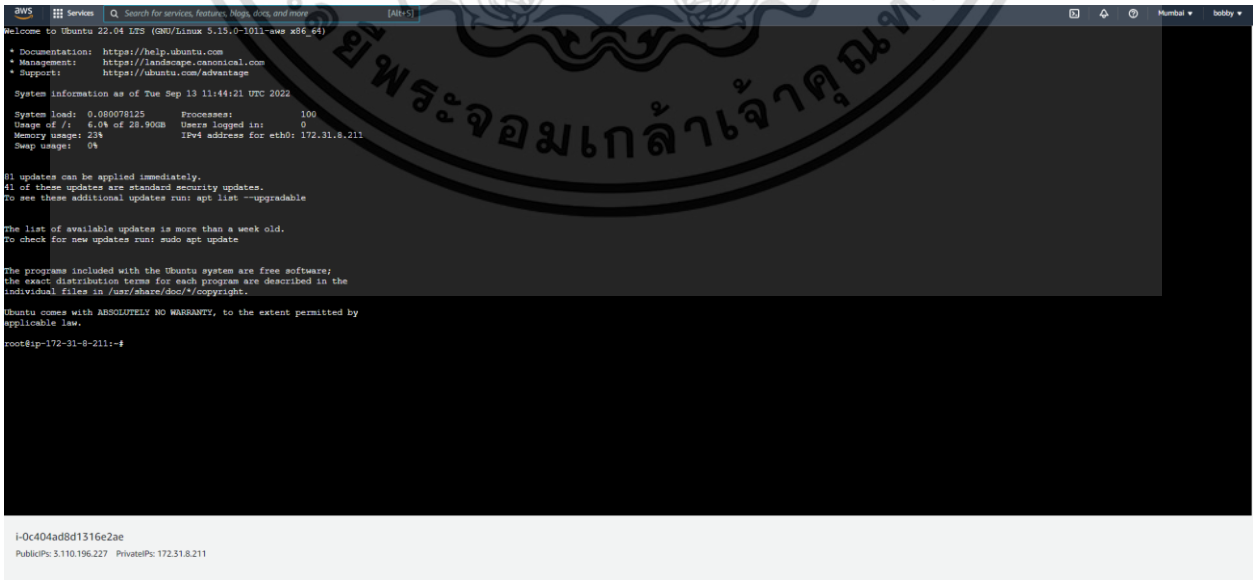


Fig. 5.31. Connect to instance via SSH of AWS

After we connected to our instance, we are going to test the scalability of the Autoscaling of AWS EC2. We will generate artificial loads as shown in Fig. 5.32 which same as when we install stress and generate loads on Google and Azure instance. This will make the CPU utilization exceed 60% and let's look back on the AWS EC2 interface to monitor its behavior.

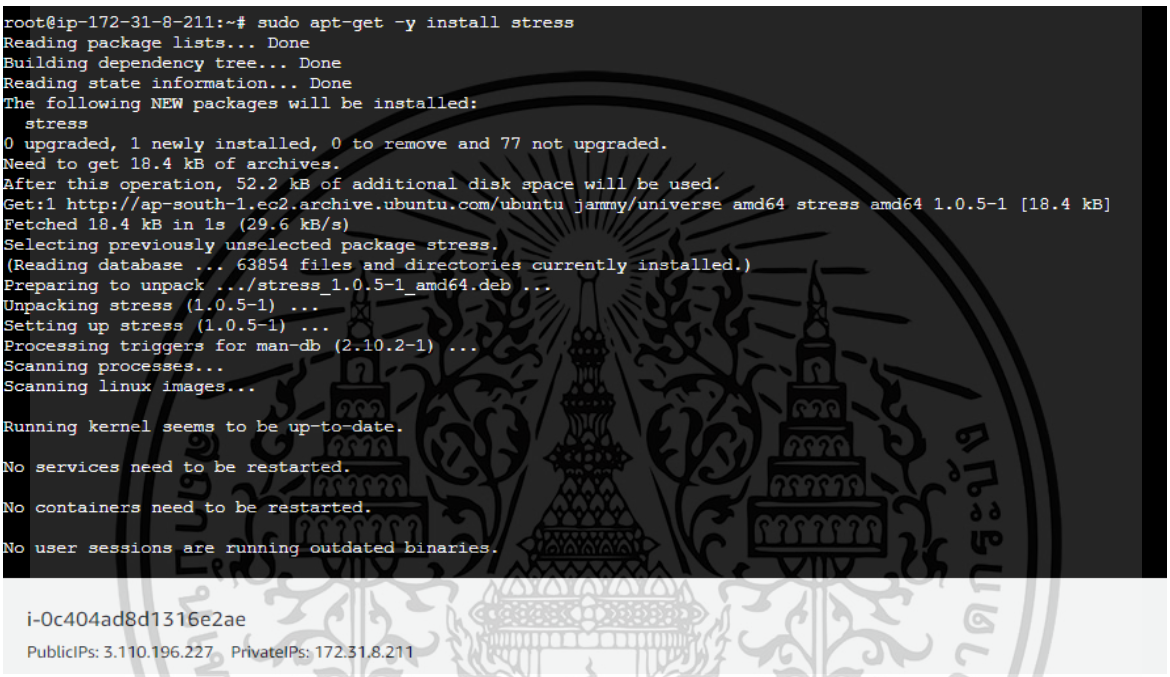


Fig. 5.32 Install stress via SSH of AWS

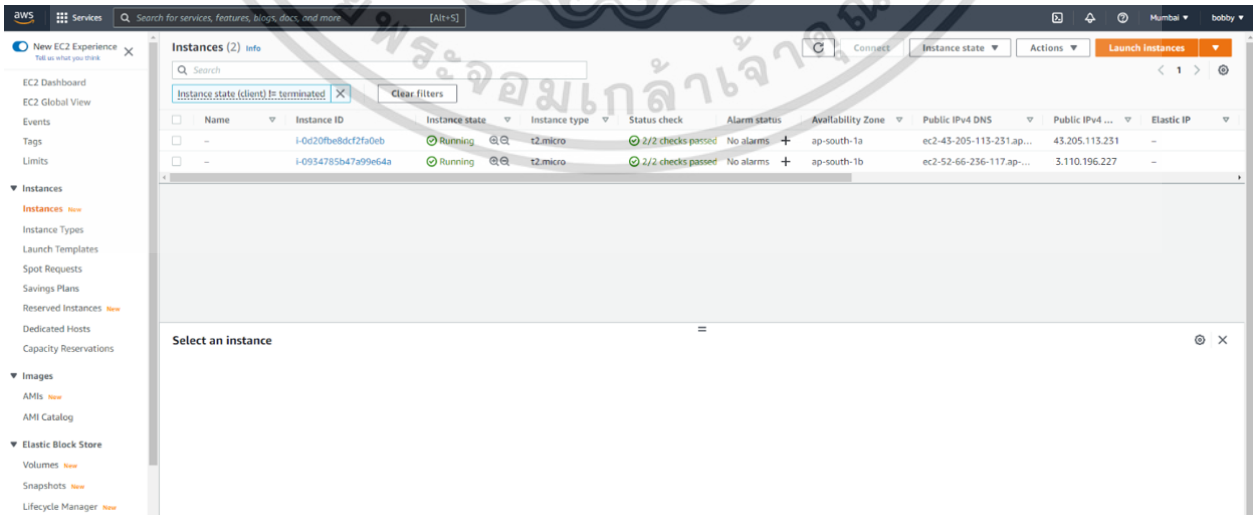


Fig. 5.33 Review instances on AWS

As we can see in Fig. 5.33, the instance group members are updating one new instance IP “43.205.113.231” so this is accurate as the Autoscaling policy that we defined. We will provide and analyze the monitoring result in the scalability testing section to ensure that the stress loads actually exceed CPU utilization of 60%.

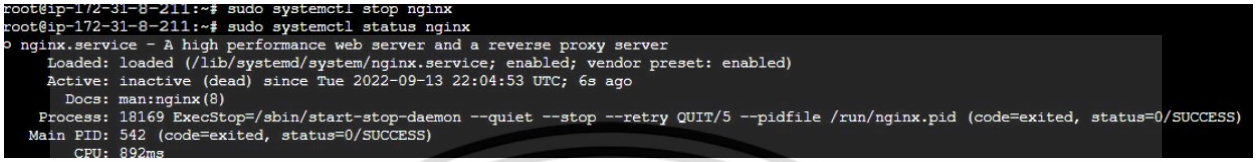


Fig. 5.34 Stop nginx web server via SSH of AWS

Fig. 5.35. Recheck instance health on AWS

Lastly, focusing on recovery instance process by stopping the web server service then we will collect down time and ensure that the web server is inactive as shown in Fig 5.34. and recheck the instance by health as shown in Fig. 5.35 and wait until the AWS EC2 handles this issue by terminating the unhealthy virtual machine and recreating another based on the policy that we defined whether a minimum of one instance with the same default custom image template and collect the creation time to compare both of instance down time and the new instance creation time for getting recovery time.

5.5 SCALABILITY TESTING

Starting with scalability testing, we tested the Auto Scaling feature of CSPs simultaneously based on the CPU utilization metric threshold at 60%. Figs. 5.36–5.39 can be clearly seen that there are 2 comparing graphs between Auto Scaling and CPU utilization of those in which we imposed an artificially high CPU workload three times by stress with a

timeout of 300, 600, and 900 seconds to those instances for approximately 2 hours of this period.

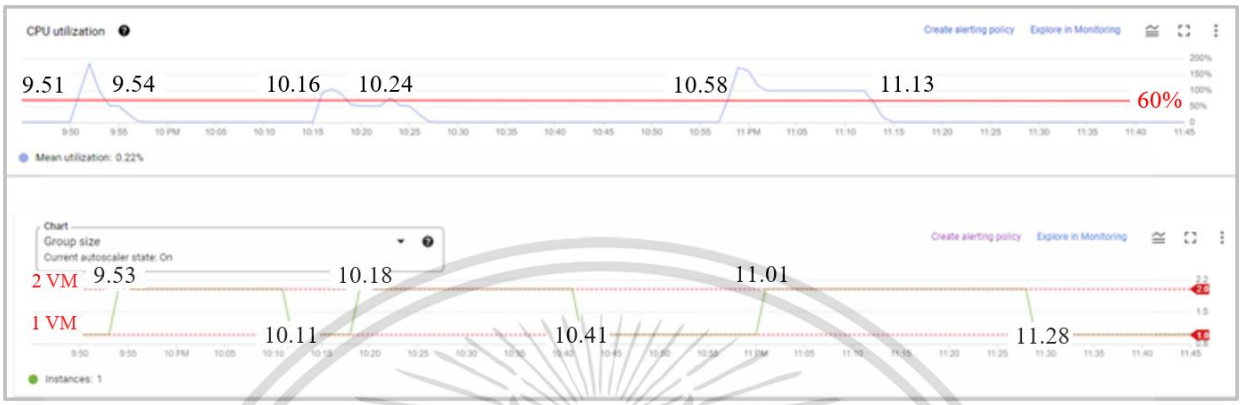


Fig. 5.36 CPU Utilization and Auto Scaling Monitoring Graphs of GCE

Looking first at GCE in Fig. 5.36, the CPU utilization initially started at 9.50 PM and then reached 60% at 9.51 PM. It scale-out from 1 to 2 instances 120 seconds later compared to the moment when it reached the threshold. After that, it dramatically decreased to 60% again at 9.54 PM but the Auto Scaling scale-in at 10.11 PM contrasted by 1020 seconds. Following this, the CPU utilization grew to hit the scaling metric for the second time at 10.16 PM, before dropping to 60% at 10.18 PM. Compared to the Auto Scaling, the instance was added at 10.18 PM differed by 120 seconds and was removed at 10.41 PM differed by 1380 seconds. Although the final reach was 10.58 PM and finished at 11.13 PM, the virtual machine was expanded and deleted at 11.01 PM and 11.28 PM. The final scale-out and scale-in times were 180 and 900 seconds.

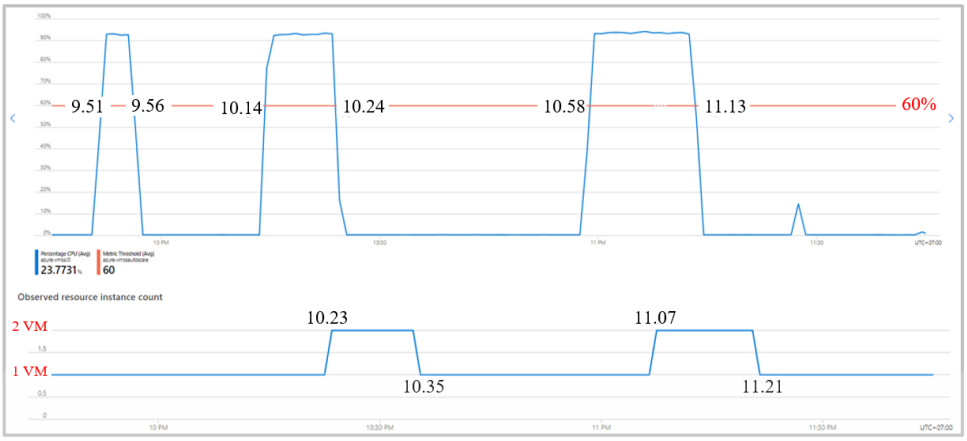


Fig. 5.37 CPU Utilization and Auto Scaling Monitoring Graphs of Azure VMs

Moving to Azure VMs in Fig. 5.37, during CPU utilization hit the threshold for the first time, the Auto Scaling did not scale because it was a very short period of 300 seconds. The condition of Azure VMs is based on the average time, not the exact time metric. Next, it recovered to 60% at 10.14 PM, during the second period of testing, 540 seconds later was quite long enough to cause the scaling out at 10.23 PM. Before falling to 1 instance at 10.35 PM by 660 seconds later compared to metric at 10.24 PM. The scaling metric was reached by the time at 10.58 PM and 11.03 PM. Compared to scaling, it was scaling after 540 and 480 seconds respectively.

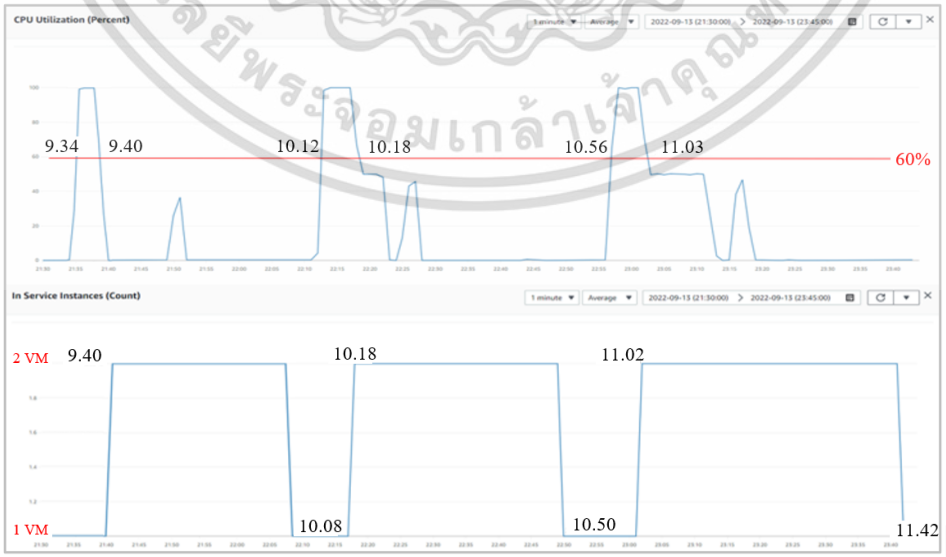


Fig. 5.38 CPU Utilization and Auto Scaling Monitoring Graphs of AWS EC2

Focusing on AWS EC2 in Fig. 5.38, the CPU utilization was more than 60% which is the scaling metric threshold at 9.34, 10.12, and 10.56 PM while the instance was added from 1 to 2 at 9.40, 10.18, and 11.02 PM. To compare those, the scale-out times were 360, 360, and 360 seconds. Besides, the CPU utilization was less than the scaling metric threshold at 9.40, 10.18, and 10.30 PM while the instance was removed from 2 to 1 at 10.08, 10.50, and 11.42 PM. To compare those, the scale-in times were quite long which were 1680, 1920, and 2340 seconds.

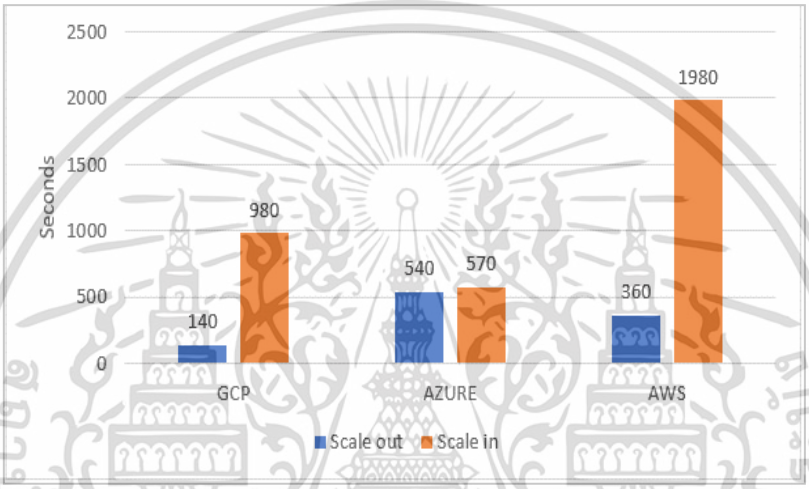


Fig. 5.39 Auto Scaling Average Time

To conclude the average time for Auto Scaling of CSPs in Fig. 5.39, GCE has by far the fastest scale-out which is 140 seconds while Azure VMs has 540 seconds which is the slowest. The scale-in of AWS EC2 has the highest figure of 1980 seconds which is more than GCP by a thousand seconds. The lowest figure of scale-in is 570 seconds for Azure which has almost the same value as scale-out.

Scale in refers to the process of reducing resources, such as computing power or storage, in response to decreasing demand. This can be done quickly to free up resources that are no longer needed, allowing for more efficient resource management during periods of low traffic. On the other hand, scale out involves increasing resources to handle increased demand, such as a sudden spike in website traffic.

Fast scaling allows for the quick deployment of additional resources to handle sudden bursts in traffic, ensuring that the system remains responsive and available. Slow scaling, on the other hand, is a more gradual approach to scaling that allows for more controlled resource management over longer periods. This can help to prevent over-provisioning of resources, which can be expensive and wasteful.

Nevertheless, fast scale-in does not mean it is always better because it has more cost for creating a new instance. On the other hand, slow scale-out probably has disadvantageous results as more charging. Overall, the choice between scale in and scale out, as well as the speed of scaling, will depend on the specific needs of the system and the demands placed on it.

Therefore, Azure has overall good value in scale-out and scale-in of 540 and 570 seconds which means it can provide both performance and pricing. GCP has the best performance in high scalability but is quite expensive, not as much as AWS

5.6 RECOVERY TESTING

During the recovery testing, the objective was to assess how cloud service providers recover from a failure or disaster scenario. The tests were carried out by simulating a sudden increase in the number of requests sent to the applications. This sudden increase in traffic causes the applications to fail, and the recovery testing evaluates how quickly the cloud service provider detects the failure, recovers from it, and restores the normal operation of the application.

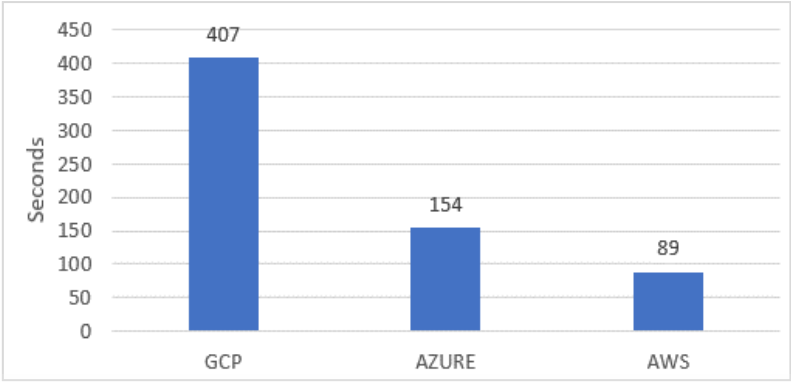


Fig. 5.40 Auto Recovery Instance Time

In the recovery testing results, as we had mentioned in the test plan section, we collected both creation and crashing time attributes of the web server provided by CSPs for generating the graph. In Fig. 5.40, AWS demonstrated the fastest recovery time, taking only 89 seconds to recover from the failure scenario. Azure came in second with a recovery time of 154 seconds, while GCP had the slowest recovery time of 407 seconds. The results indicate that AWS has a more efficient recovery process than Azure and GCP, which could be due to AWS's significant investment in disaster recovery and business continuity.

During the tests, it was observed that all cloud service providers were able to detect the failure scenario immediately and initiate the recovery process. AWS and Azure used automated recovery mechanisms that restored the normal operation of the application within the shortest possible time. GCP, on the other hand, used a manual recovery process that required human intervention, resulting in a slower recovery time.

It is worth noting that the recovery testing results could be affected by several factors, such as the size and complexity of the applications, the type of failure, and the availability of backup resources. In this study, the tests were carried out on relatively small applications, and the failure scenario was a sudden increase in traffic. Therefore, the recovery testing results could differ when applied to larger and more complex applications or different types of failure scenarios.

The recovery testing results suggest that AWS is the most reliable cloud service provider for applications that require high availability and quick recovery from failure scenarios.

Azure is also a viable option, especially for applications that require moderate recovery time. GCP, however, may not be the best option for applications that require quick recovery from failure scenarios, as its manual recovery process could result in longer recovery times.

Overall, the recovery testing results emphasize the importance of having a robust disaster recovery and business continuity plan in place when using cloud service providers. It is also essential to evaluate the recovery capabilities of cloud service providers before selecting a provider for hosting critical applications.

5.7 LOAD TESTING

In this section, we will provide load testing results of CSPs with the Apache JMeter tool including Aggregate graph, Response time graph, and Summary report respectively. Focusing first on Aggregate graph which shows trends in performance metrics over time. The information as shown in Fig. 5.41 can be represented in graphical form as well through this listener, using the Display Graph option. It makes it easy to analyze the data and to work on it as Graphical representation is easy to understand and analyze showing as metrics below:

#Samples: Total number of Samples.

Average: Average Response Time.

Median: is a number which divides the samples into two equal halves. Half of the samples are smaller than the median, and half are larger (Some samples may equal the median.)

90% Line: It represents that 10% of the samplers have exceeded time to reach the server.

95% Line: It represents that 5% of the samplers have exceeded time to reach the server.

99% Line: It represents that 1% of the samplers have exceeded time to reach the server.

Min: This is the minimum time a sampler has taken to go to the server.

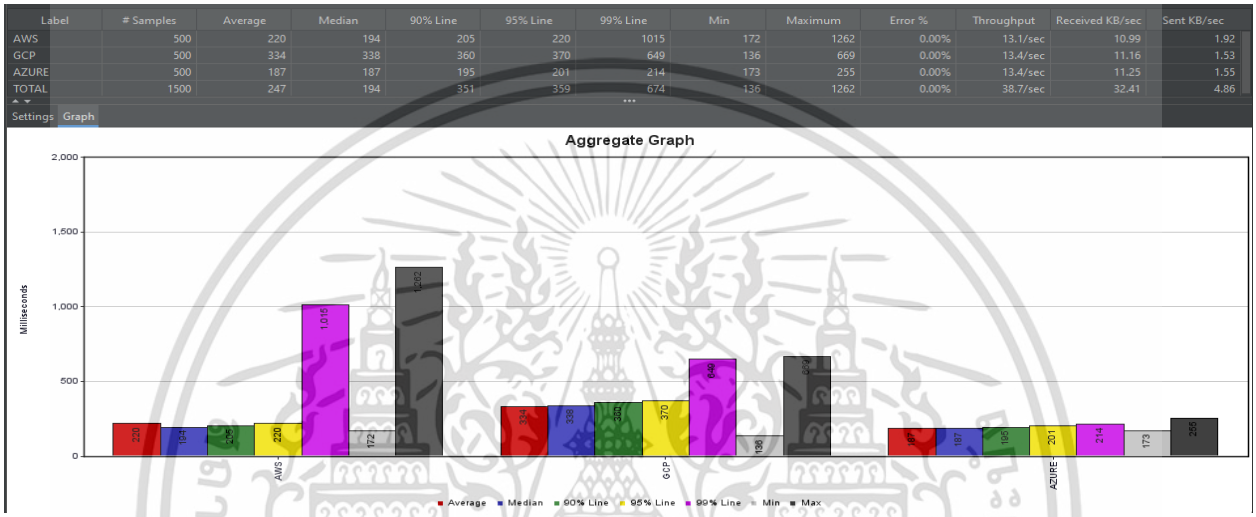
Max: This is the maximum time request taken to go to the server.

Error%: Number of error sampler/Total number of Sampler.

Throughput: Throughput is the per second sample received by the server.

Received KB/second: This defines how many kilobytes per second received by the Client.

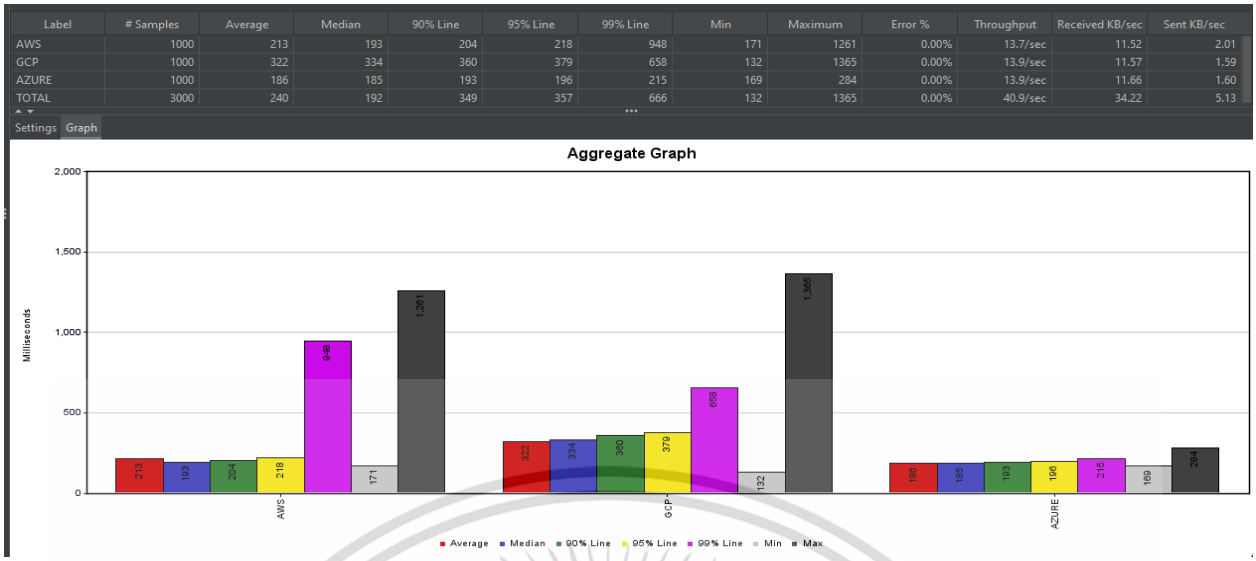
Sent KB/second: This defines how many Kilobytes per second are sent to the server.



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
AWS	500	220	194	205	220	1015	172	1262	0.00%	13.1/sec	10.99	1.92
GCP	500	334	338	360	370	649	136	669	0.00%	13.4/sec	11.16	1.53
AZURE	500	187	187	195	201	214	173	255	0.00%	13.4/sec	11.25	1.55
TOTAL	1500	247	194	351	359	674	136	1262	0.00%	38.7/sec	32.41	4.86

Fig. 5.41 Aggregate graph of 500 requests

In Fig. 5.41 with 500 samples of each CSP, it can be seen that there are no errors for those of 0.00%. AWS endpoint falls in the middle with an average response time of 220 ms, median response time of 194 ms, and 90th, 95th, and 99th percentile response times of 205 ms, 220 ms, and 1015 ms respectively. indicating that 1% of the samplers have exceeded 1015 ms to reach the server but only 1% may conclude as overall faster than GCP. GCP endpoint has the highest average response time of 334 ms, median response time of 338 ms, and 90th, 95th, and 99th percentile response times of 360 ms, 370 ms, and 649 ms respectively, indicating that it performs slowest compared to others. Azure endpoint has the lowest average response time of 187 ms, median response time of 187 ms, and 90th, 95th, and 99th percentile response times of 195 ms, 201 ms, and 214 ms respectively, indicating that it consistently performs well across all levels of traffic.



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
AWS	1000	213	193	204	218	948	171	1261	0.00%	13.7/sec	11.52	2.01
GCP	1000	322	334	360	379	658	132	1365	0.00%	13.9/sec	11.57	1.59
AZURE	1000	186	185	193	196	215	169	284	0.00%	13.9/sec	11.66	1.60
TOTAL	3000	240	192	349	357	666	132	1365	0.00%	40.9/sec	34.22	5.13

Fig. 5.42 Aggregate graph of 1000 requests

In Fig. 5.42 with 1000 samples of each CSP, it can be seen that there are no errors for those of 0.00%. AWS endpoint falls in the middle with an average response time of 213 ms, median response time of 193 ms, and 90th, 95th, and 99th percentile response times of 204 ms, 218 ms, and 948 ms respectively. indicating that 1% of the samplers have exceeded 948 ms to reach the server but only 1% may conclude as overall faster than GCP. GCP endpoint has the highest average response time of 322 ms, median response time of 334 ms, and 90th, 95th, and 99th percentile response times of 360 ms, 379 ms, and 658 ms respectively, indicating that it performs slowest compared to others. Azure endpoint has the lowest average response time of 186 ms, median response time of 185 ms, and 90th, 95th, and 99th percentile response times of 193 ms, 196 ms, and 215 ms respectively, indicating that it consistently performs well across all levels of traffic.

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
AWS	3000	735	194	1192	1212	15223	170	43124	0.57%	14.6/sec	12.38	2.12
GCP	3000	693	330	1138	1354	15386	129	21055	0.97%	14.6/sec	12.38	1.65
AZURE	3000	525	186	223	1199	7223	168	21055	0.83%	14.6/sec	12.44	1.67
TOTAL	9000	651	193	1152	1254	15226	129	43124	0.79%	43.7/sec	37.13	5.43

Settings Graph

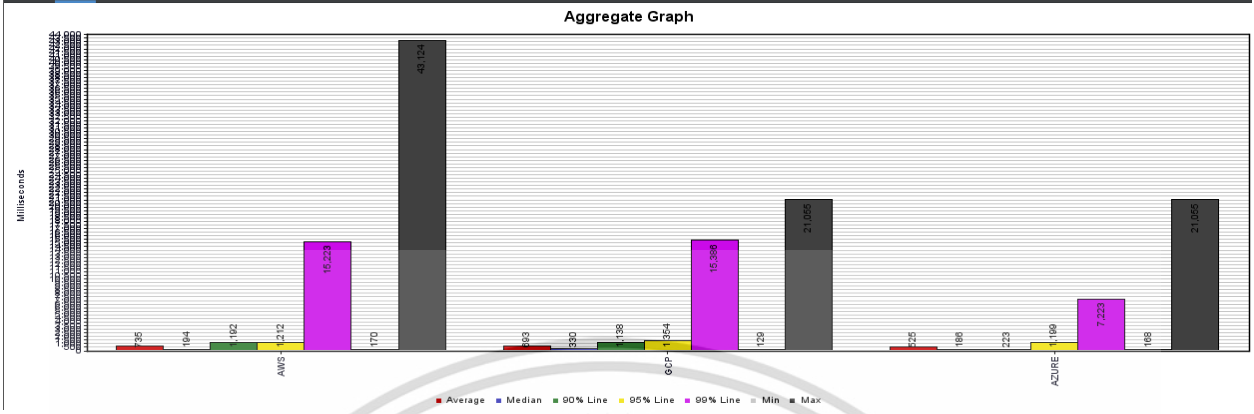
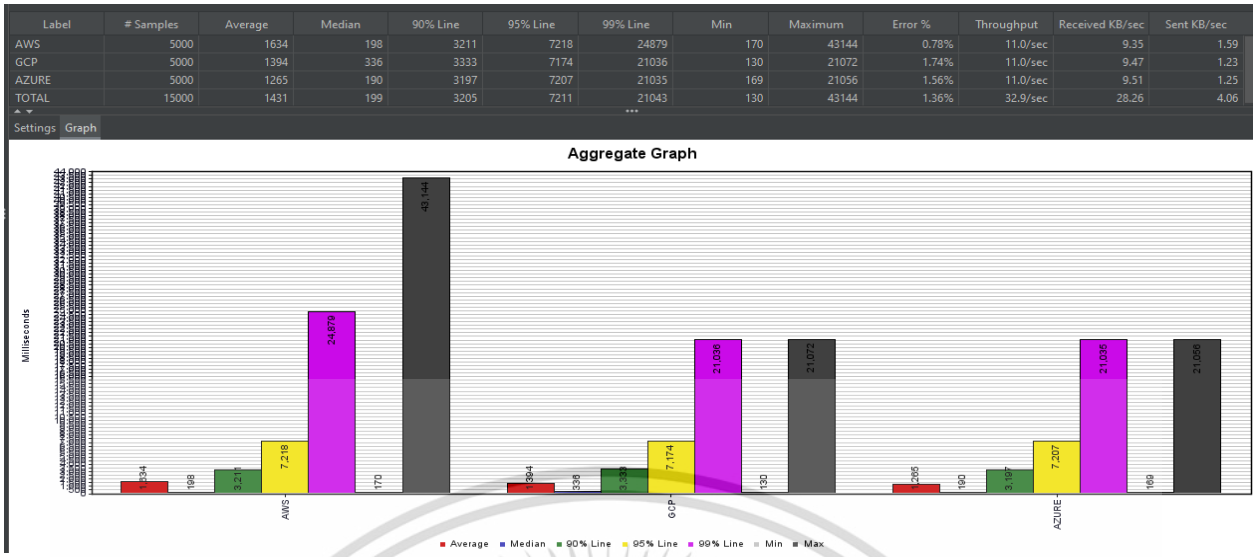


Fig. 5.43 Aggregate graph of 3000 requests

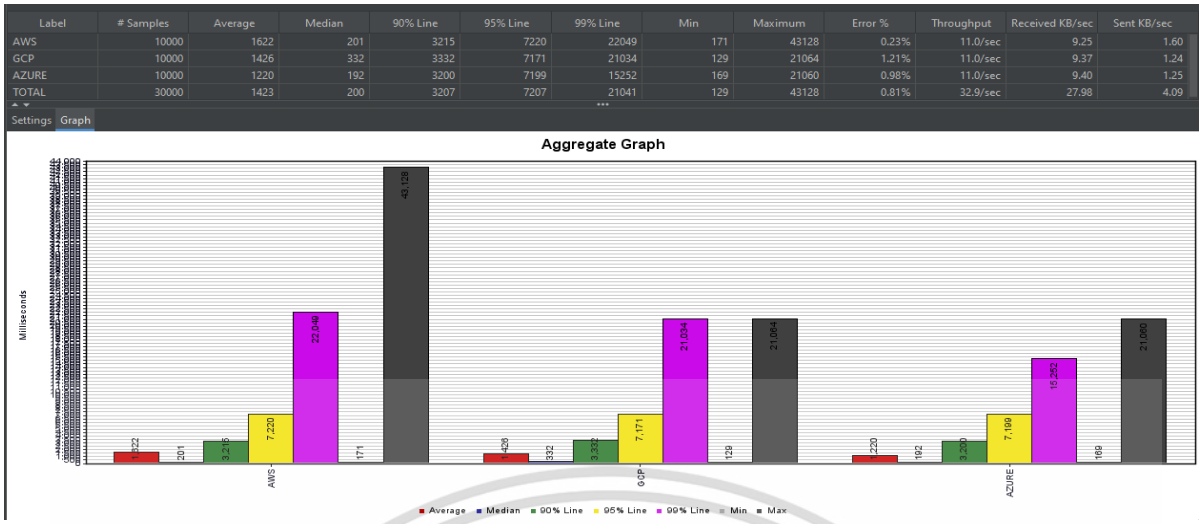
In Fig. 5.43 with 3000 samples of each CSP, it can be seen that the errors starting occur to those. AWS endpoint has the highest average response time of 735 ms, median response time of 194 ms, and 90th, 95th, and 99th percentile response times of 1192 ms, 1212 ms, and 15223 ms respectively with an error percentage of 0.57%, indicating that it performs slowest compared to others when the requests exceed 3000 samples but performing the most stable according to the lowest error percentage. GCP endpoint falls in the middle with an average response time of 693 ms, median response time of 330 ms, and 90th, 95th, and 99th percentile response times of 1138 ms, 1354 ms, and 15386 ms respectively with an error percentage of 0.97%, indicating that it performs the most unsteady according to the highest error percentage although slightly faster than AWS. Azure endpoint has the lowest average response time of 525 ms, median response time of 186 ms, and 90th, 95th, and 99th percentile response times of 223 ms, 1199 ms, and 7223 ms respectively with an error percentage of 0.83%, indicating that it consistently performs well across all levels of traffic with the fastest response time and middle place of the error rate.



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
AWS	5000	1634	198	3211	7218	24879	170	43144	0.78%	11.0/sec	9.35	1.59
GCP	5000	1394	336	3333	7174	21036	130	21072	1.74%	11.0/sec	9.47	1.23
AZURE	5000	1265	190	3197	7207	21035	169	21056	1.56%	11.0/sec	9.51	1.25
TOTAL	15000	1431	199	3205	7211	21043	130	43144	1.36%	32.9/sec	28.26	4.06

Fig. 5.44 Aggregate graph of 5000 requests

In Fig. 5.44 with 5000 samples of each CSP, it can be seen that the errors occur to those. AWS endpoint has the highest average response time of 1634 ms, median response time of 198 ms, and 90th, 95th, and 99th percentile response times of 3211 ms, 7218 ms, and 24879 ms respectively with an error percentage of 0.78%, indicating that it performs slowest compared to others but performing the most stable according to the lowest error percentage. GCP endpoint falls in the middle with an average response time of 1394 ms, median response time of 336 ms, and 90th, 95th, and 99th percentile response times of 3333 ms, 7174 ms, and 21036 ms respectively with an error percentage of 1.74%, indicating that it performs the most unsteady according to the highest error percentage although slightly faster than AWS. Azure endpoint has the lowest average response time of 1265 ms, median response time of 190 ms, and 90th, 95th, and 99th percentile response times of 3197 ms, 7207 ms, and 21035 ms respectively with an error percentage of 1.56%, indicating that it consistently performs well across all levels of traffic with the fastest response time and middle place of the error rate.



Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/sec	Sent KB/sec
AWS	10000	1622	201	3215	7220	22049	171	43128	0.23%	11.0/sec	9.25	1.60
GCP	10000	1426	332	3332	7171	21034	129	21064	1.21%	11.0/sec	9.37	1.24
AZURE	10000	1220	192	3200	7199	15252	169	21060	0.98%	11.0/sec	9.40	1.25
TOTAL	30000	1423	200	3207	7207	21041	129	43128	0.81%	32.9/sec	27.98	4.09

Fig. 5.45 Aggregate graph of 10000 requests

In Fig. 5.45 with 10000 samples of each CSP, it can be seen that the errors occur to those. AWS endpoint has the highest average response time of 1622 ms, median response time of 201 ms, and 90th, 95th, and 99th percentile response times of 3215 ms, 7220 ms, and 22049 ms respectively with an error percentage of 0.23%, indicating that it performs slowest compared to others but performing the most stable according to the lowest error percentage. GCP endpoint falls in the middle with an average response time of 1426 ms, median response time of 332 ms, and 90th, 95th, and 99th percentile response times of 3332 ms, 7171 ms, and 21034 ms respectively with an error percentage of 1.21%, indicating that it performs the most unsteady according to the highest error percentage although slightly faster than AWS. Azure endpoint has the lowest average response time of 1220 ms, median response time of 192 ms, and 90th, 95th, and 99th percentile response times of 3200 ms, 7199 ms, and 15252 ms respectively with an error percentage of 0.98%, indicating that it consistently performs well across all levels of traffic with the fastest response time and middle place of the error rate.

Followed by the Response time graph as shown which displays the response time of each request during the test, allowing us to identify any specific areas where performance was slow or there were delays. This graph can be helpful to identify the root cause of any performance issues and optimize the application accordingly. We will analyze response times for individual requests, identify slowest requests, look for trends or patterns in response times, compare response times for different requests

Response time graphs of 500, 1000, 3000, 5000, and 10000 user requests were also plotted as shown in Figs. 5.46–5.50 for supporting the response time that we analyzed in Table 3. Overall, It can be seen that Azure has the lowest response time followed by AWS, and GCP but we can notice that AWS has the highest response time the first time that it reached the server then it decreased dramatically to be the second place fastest response time.

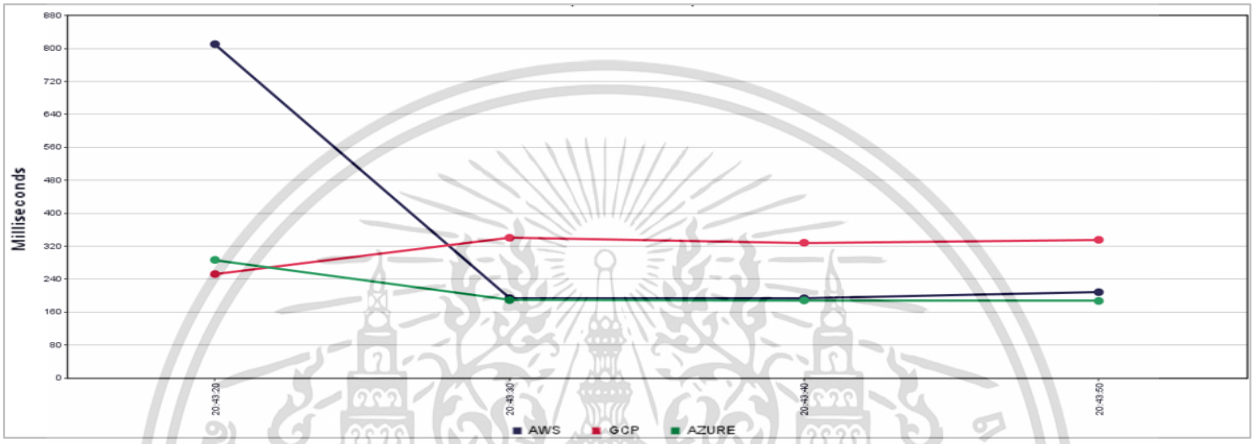


Fig. 5.46 Response time graph of 500 requests

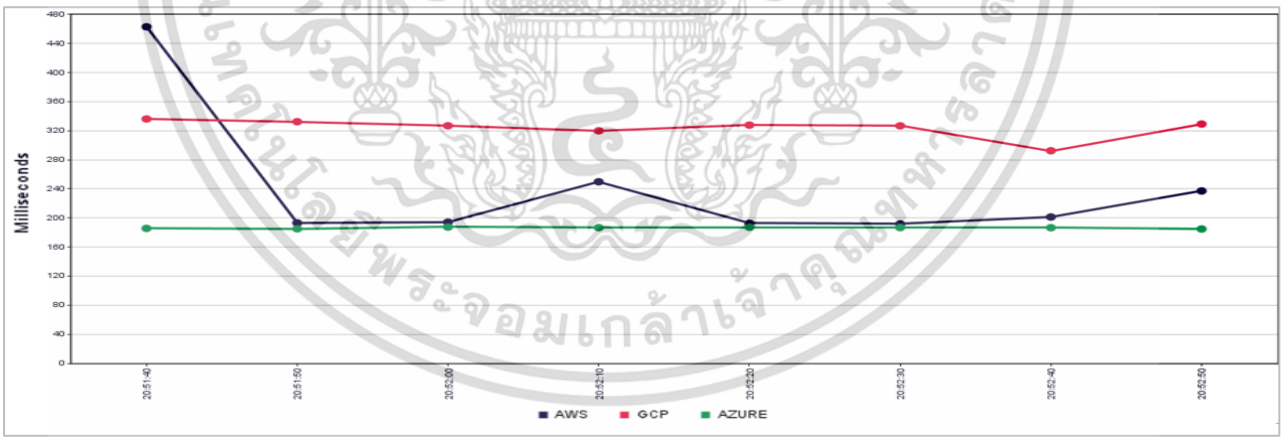


Fig. 5.47 Response time graph of 1000 requests

In Figs. 5.46-5.47, it can be clearly noticeable that the response times of 500 and 1000 users are quite constant despite the fact that they never get any error or the error rate is 0%.

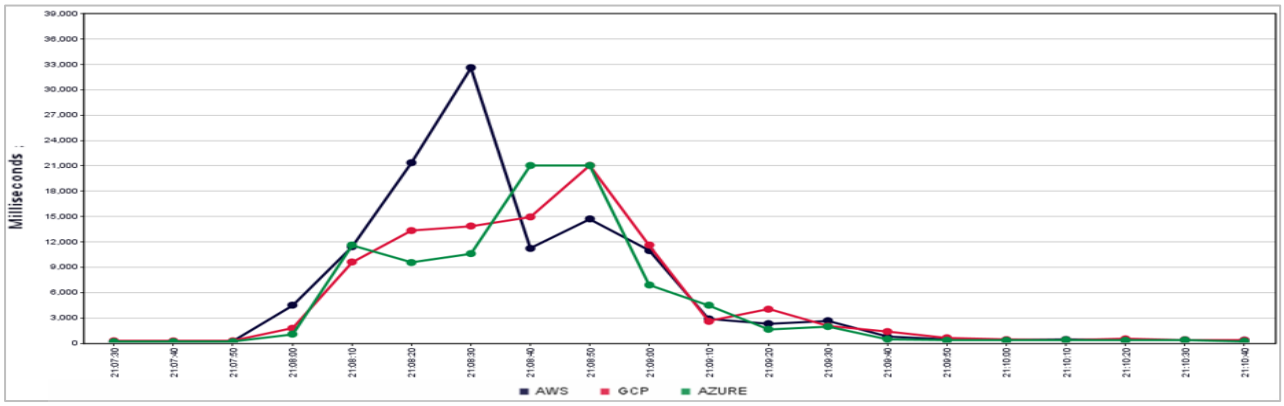


Fig. 5.48 Response time graph of 3000 requests

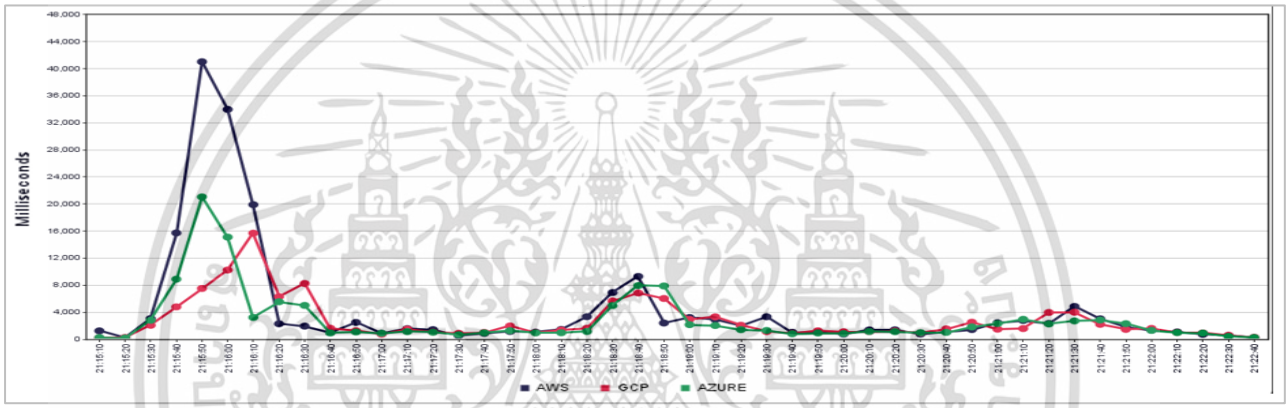


Fig. 5.49 Response time graph of 5000 requests

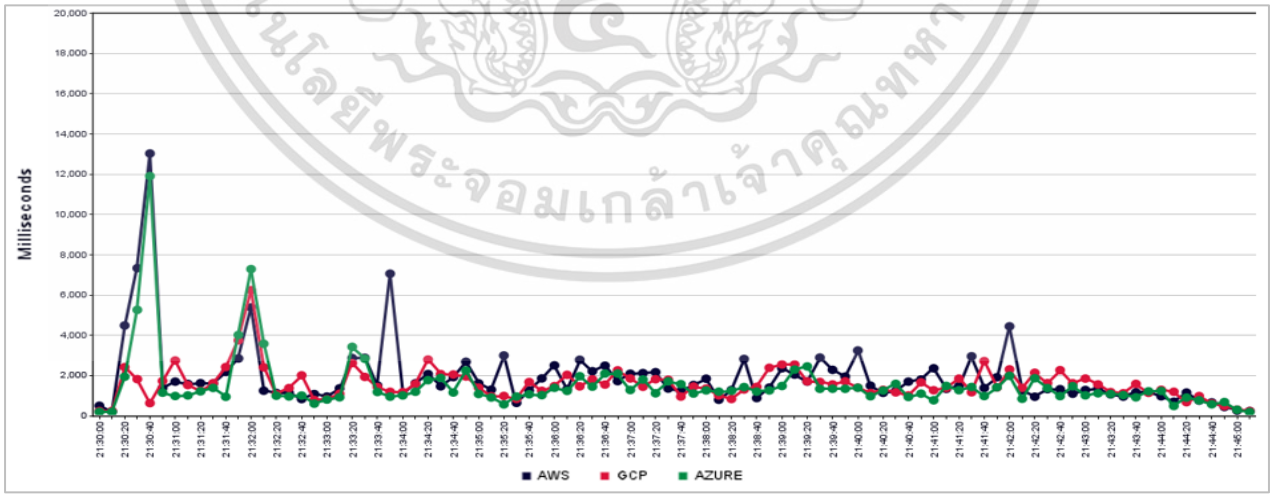


Fig. 5.50 Response time graph of 10000 requests

In Figs 5.48-5.50 the response time graphs from 3000 to 10000 very fluctuate because the error started arising the same as we analyzed.

To summarize, we analyze the overall performance of the application, compare metrics to goals/requirements, identify errors or performance issues, make informed decisions about improving performance by Summary Report which provides an overall of the CSPs performance under load. In Table 3, it presents performance metrics for three major cloud service providers: GCP, Azure, and AWS. The metrics include response time, error rate, and throughput, which are evaluated for 500, 1000, 3000, 5000, and 10000.

Table 3. Summary Report

CSP	Users	Avg	Min	Max	Std Dev	Err%	TP	Received KB/sec	Sent KB/sec
GCP	500	334 ms	136 ms	669 ms	74.45	0.00	13.4	11.16	1.53
	1000	322 ms	132 ms	1365 ms	104.40	0.00	13.9	11.57	1.59
	3000	693 ms	129 ms	21055 ms	2302.56	0.97	14.6	12.38	1.65
	5000	1394 ms	130 ms	21072 ms	3344.56	1.74	11.0	9.47	1.23
	10000	1426 ms	129 ms	21064 ms	3187.43	1.21	11.0	9.37	1.24
Azure	500	187 ms	173 ms	255 ms	7.94	0.00	13.4	11.25	1.55
	1000	186 ms	169 ms	284 ms	8.55	0.00	13.9	11.66	1.60
	3000	525 ms	168 ms	21055 ms	2061.39	0.83	14.6	12.44	1.67
	5000	1265 ms	169 ms	21056 ms	3287.01	1.56	11.0	9.51	1.25
	10000	1220 ms	169 ms	21060 ms	2894.14	0.98	11.0	9.40	1.25
AWS	500	220 ms	172 ms	1252 ms	140.43	0.00	13.1	10.99	1.92
	1000	213 ms	171 ms	1261 ms	123.51	0.00	13.7	11.52	2.01
	3000	735 ms	170 ms	43124 ms	3564.14	0.57	14.6	12.38	2.12
	5000	1634 ms	170 ms	43144 ms	4927.64	0.78	11.0	9.35	1.59
	10000	1622 ms	171 ms	43128 ms	4159.19	0.23	11.0	9.25	1.60

Remark:

Users: Total number of samples is the number of users per request

Avg: Average time is calculated based on the time taken by samples to run and the unit is a millisecond.

Min: This is the minimum time request taken to go to the server and the unit is a millisecond.

Max: This is the maximum time request taken to go to the server and the unit is a millisecond.

Std Dev: It is a deviation from the average value of sample response time.

Err%: Percentage of failed requests.

TP: Throughput is the number of requests per second received by the server.

KB/Sec: This shows how much information was downloaded from the server during the test execution.

Starting with response time, we can see that Azure had the fastest average response time across all user counts, with an average response time ranging from 186 ms to 1265 ms. In contrast, GCP and AWS had slower average response times, with response times ranging from 322 ms to 1426 ms for GCP, and 213 ms to 1634 ms for AWS. Interestingly, GCP had the lowest minimum response time across all user counts, with response times as low as 129 ms, while Azure and AWS had higher minimum response times, with response times ranging from 173 ms to 669 ms for Azure, and 170 ms to 172 ms for AWS.

Moving on to error rates, we can see that all three cloud providers had very low error rates for the smaller user counts of 500 and 1000, with error rates of 0%. However, as the number of users increased, error rates also increased significantly for all three providers, with error rates as high as 0.97% for GCP, 0.83% for Azure, and 0.57% for AWS at the highest user count of 3000. In user count of 5000, error rates are 1.74% for GCP, 1.56% for Azure, and 0.78% for AWS. Interestingly, AWS had the lowest error rate across all three providers for the highest user count of 10000, with an error rate of only 0.23% while 0.98% for Azure and 1.21% for GCP.

In terms of throughput, we can see that all three cloud providers had similar throughputs across all user counts, with throughputs ranging from 11.0 requests per second (rps) to 14.6 rps. However, there were some variations in throughput depending on the user count and cloud provider. The throughput of the CSPs are similar except for AWS which has 13.1 for 500 users and 13.7 for 1000 users.

The total data transfer rates (received and sent) per second for each CSP. To analyze the performance and potential bottlenecks related to data transfer rates, let's compare the calculated bandwidth values for each CSP. For AWS, received data rate is 53.49 KB/sec, sent data rate is 9.24 KB/sec, the total data transfer rate is 62.73 KB/sec. For GCP, received data rate is 53.95 KB/sec, sent data rate is 7.24 KB/sec, the total data transfer rate is 61.19 KB/sec. For Azure, received data rate is 54.26 KB/sec, sent data rate is 7.32 KB/sec, the total data transfer rate is 61.58 KB/sec. From the provided data, we can observe that AWS has the highest bandwidth among the three CSPs, followed closely by Azure and GCP. This indicates that AWS has a relatively higher data transfer rate compared to the other two CSPs.

Analyzing the bandwidth values can provide insights into potential bottlenecks related to data transfer rates. Higher bandwidth generally indicates better performance in handling data transfer. Therefore, in this comparison, AWS may have better performance and efficiency in terms of data transfer rates compared to GCP and Azure.

It's also worth noting the standard deviation, which is a measure of the variability in response time across the different samples. Azure had the lowest standard deviation across all user counts, indicating that its performance was more consistent than that of GCP and AWS. In contrast, GCP and AWS had relatively high standard deviations, suggesting more variability in performance.

Overall, the table provides valuable insights into the performance of different cloud service providers, and highlights the importance of considering multiple performance metrics when evaluating different providers. While Azure had the fastest average response time and the lowest standard deviation, AWS had the lowest error rate for the highest user count. Additionally, while GCP had the lowest minimum response time, it also had the highest error rates for larger user counts. Therefore, organizations should carefully evaluate their specific requirements and choose the cloud service provider that best meets their needs based on a range of performance metrics and other factors

CHAPTER 6

DISCUSSION

6.1 FINDINGS AND IMPLICATIONS

The findings of our analysis reveal several important implications for organizations considering cloud service providers for their applications. First and foremost, response time is a critical performance metric that should be carefully considered. Our analysis found that Azure had the fastest average response time across all user counts, while GCP and AWS had slower average response times. However, it is important to note that GCP had the lowest minimum response time, which may be important for applications that require very fast response times for specific requests.

In terms of error rates, all three cloud providers had very low error rates for smaller user counts, but error rates increased significantly as the number of users increased. This underscores the importance of testing and monitoring applications under realistic load conditions to ensure that error rates remain acceptable even as usage scales up. Our analysis also found that AWS had the lowest error rate for the highest user count, which may be an important consideration for applications that require high levels of reliability and availability.

Throughput was found to be a less important differentiating factor among the three cloud providers, with all three having similar throughputs across all user counts. However, organizations should still carefully evaluate their specific throughput requirements and choose a provider that can meet those requirements.

Another important implication of our analysis is the importance of considering multiple performance metrics when evaluating cloud service providers. For example, while Azure had the fastest average response time, it also had the highest standard deviation, indicating that its performance was more variable than that of GCP and AWS. This may be a consideration for applications that require consistent performance.

Additionally, our analysis highlights the importance of testing for scalability and recovery. Scalability testing is important to ensure that applications can handle increasing levels of usage without significant degradation in performance or stability. Recovery testing, on the other hand, is important to ensure that applications can recover quickly and reliably from failures, such as server crashes or network outages.

Finally, it is worth noting that while performance is an important consideration when evaluating cloud service providers, it is not the only consideration. Other factors, such as cost, reliability, security, and ease of use, are also important factors to consider. Organizations should carefully evaluate all these factors and choose a cloud service provider that best meets their specific needs.

Our analysis provides valuable insights into the performance of three major cloud service providers – Azure, GCP, and AWS – across multiple performance metrics and user counts. The findings of our analysis have important implications for organizations considering cloud service providers for their applications and highlight the importance of considering multiple performance metrics, testing for scalability and recovery, and evaluating other factors beyond performance alone. By carefully evaluating their options and choosing a cloud service provider that best meets their specific needs, organizations can ensure that their applications perform reliably and meet their business requirements.

6.2 LIMITATIONS AND FUTURE WORK

There are several limitations to this study that should be considered when interpreting the results. First, the study was conducted using synthetic workloads rather than real-world workloads, which may limit the generalizability of the findings. Future studies could examine the performance of different cloud service providers using real-world workloads to determine if the results are consistent with those found in this study.

Another limitation is that the study only examined the performance of three major cloud service providers, which may not be representative of the entire cloud computing

market. Future studies could expand the scope of the analysis to include a wider range of cloud service providers to provide a more comprehensive comparison.

Additionally, this study only focused on these performance tests: auto-scaling, recovery, and load. There are many other important factors that organizations should consider when evaluating cloud service providers, such as cost, reliability, and security. Future studies could examine these factors to provide a more complete picture of the strengths and weaknesses of different cloud service providers.

Furthermore, our study only tested CSPs in a normal scenario. It did not explore their performance under abnormal conditions, such as network congestion, power outages, or malicious attacks. Future work should consider exploring the impact of such scenarios on CSPs' performance to help organizations to better understand their performance under stress and in various scenarios.

Finally, while the performance of different CSPs has been analyzed, this study did not consider the security aspects of the CSPs. Security is one of the most important considerations when it comes to selecting a CSP for an organization. Future work should focus on evaluating the security of different CSPs, including their security policies, data protection, network security, and other aspects that could affect the security of the organization's data and applications.

In terms of future work, there is an urgent need to evaluate the security of different CSPs. Cloud computing has become a critical component of modern computing, and it is increasingly being used to store and process sensitive data, such as personal health information, financial data, and intellectual property. As a result, security has become a major concern for organizations that use cloud services. Future studies could examine the security features of different cloud service providers to determine their strengths and weaknesses.

Moreover, future studies could also explore the performance of CSPs in hybrid cloud environments, which combine public and private clouds. Hybrid cloud environments offer organizations more flexibility and control over their data and applications, but they also present unique challenges in terms of performance and security. By examining the

performance of CSPs in hybrid cloud environments, organizations can gain a better understanding of how to optimize their cloud infrastructure to meet their specific needs.

Another area for future research could be the impact of CSPs on the environment. Cloud computing consumes a significant amount of energy, and many organizations are concerned about the environmental impact of their cloud services. Future studies could examine the energy consumption of different cloud service providers and identify ways to optimize their energy usage to reduce their carbon footprint.

This research provides valuable insights into the performance of different cloud service providers, there are several limitations that should be considered when interpreting the results. Future studies could expand the scope of the analysis to include a wider range of performance metrics, as well as other important factors such as security.

In future work, we are looking forward to taking more non-functional testing, especially focusing on security testing. To ensure that a system is protected against deliberate and instantaneous attacks from internal and external sources. Definite impacts will occur if the system has vulnerabilities or unauthorized access. This could put the system at risk so it is necessary to avoid threats from criminals. Therefore, we are going to identify the weaknesses in the system as well as suggest solutions or mitigate risks that may arise.

Additionally, future research could explore the performance of CSPs under abnormal conditions and in hybrid cloud environments, as well as their impact on the environment. Overall, the findings of this study provide a useful starting point for organizations that are evaluating different cloud service providers and looking to optimize their cloud infrastructure.

CHAPTER 7

CONCLUSION

7.1 SUMMARY OF THE STUDY

The objective of this study was to evaluate the performance of Infrastructure as a Service (IaaS) across three major Cloud Service Providers: Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP). The study focused on the testing of virtual machines (VMs) and the web server deployed on them. The study evaluated the performance of each cloud provider in terms of Auto Scaling Testing, Recovery Testing, and Load Testing. The findings and implications of this study are discussed in this section.

Overall, the performance of all three cloud service providers was found to be satisfactory. The VMs of all three providers demonstrated good scalability, stability, and fault tolerance. The Auto Scaling Testing results showed that all three providers were able to scale up and down resources as per demand. The Recovery Testing results showed that all three providers were able to recover from failures in a reasonable amount of time. The Load Testing results showed that all three providers were able to handle high-traffic loads without any major issues.

There were some differences in performance between the three providers. In this research, we provided the analysis and results of performance testing including Scaling testing, Recovery testing, and Load testing respectively of those clouds which are Google Cloud Platform, Amazon Web Service, and Microsoft Azure. From these different behaviors of IaaS providers in this experiment, Azure performs the best overall in scaling including cost and the greatest average response time but its hardest configuration, AWS can handle the crashing webserver well with the fastest recovery, and GCP has the best performance in high scalability but quite expensive, not as much as AWS and GCP also has the simplest configuration suitable for initiates.

Table 4. Functionality Score

	GCP	AWS	AZURE
Scalability	★★★	★	★★
Low Cost (Scaling)	★	★	★★★
Recovering	★	★★★	★★
Response Time	★	★★	★★★
Data Transfer	★★	★★★	★★
Error Rate	★	★★★	★★

Based on the findings of this study, it can be concluded that all three cloud service providers offer reliable and high-performing IaaS solutions. However, the choice of provider should be based on specific business requirements, as each provider has its own strengths and weaknesses. According to the Table 4, if overall performance is desired, Azure may be the best choice. If fast recovery times are critical and need stable server due to the low error rate and has a relatively higher data transfer rate, AWS may be the best choice. If we need high scalability and not concerning about cost, GCP may be the best choice.

In addition to the specific findings regarding the performance of each provider, this study also highlights the importance of proper testing and evaluation of IaaS solutions before deployment. The findings of this study demonstrate that performance can vary significantly between providers and can have a significant impact on the success of a cloud deployment. Therefore, businesses should carefully evaluate their options and conduct thorough testing before making a decision.

This study provides valuable insights into the performance of IaaS solutions across three major cloud service providers. The findings of this study can be used by businesses to make informed decisions regarding their cloud deployment strategy. The study demonstrates that proper testing and evaluation are critical to ensuring the success of cloud deployment and that each provider has its own strengths and weaknesses that should be carefully considered before making a decision. Overall, this study demonstrates the importance of

performance evaluation in the selection of an IaaS provider and highlights the need for ongoing monitoring and optimization of cloud deployments to ensure continued success.

7.2 CONTRIBUTION AND SIGNIFICANCE OF THE STUDY

The contribution and significance of the study lies in its comprehensive evaluation of the performance of three major cloud service providers in delivering infrastructure as a service. The study addresses the lack of objective information on the performance of cloud services, which is a major obstacle in making informed decisions in cloud infrastructure management.

By conducting various tests, including auto-scaling, recovery, and load testing, the study provides a detailed analysis of the performance of Google Cloud Platform, Microsoft Azure, and Amazon Web Services. The findings of the study offer insights on the strengths and weaknesses of each provider in terms of scalability, recovery, and response time.

One of the major contributions of the study is the identification of the best use cases for each cloud service provider. For instance, the study highlights that Microsoft Azure is the most efficient in scaling and has the greatest average response time, making it ideal for businesses that require high scalability and fast response times. Meanwhile, Amazon Web Services stands out in terms of its ability to handle crashing web servers well, making it suitable for applications that require high availability. Finally, Google Cloud Platform offers a simple configuration, making it a great choice for beginners and small businesses.

Another significant contribution of the study is the provision of recommendations to help businesses optimize their cloud infrastructure. By identifying the areas that need improvement, the study provides actionable insights that can be used to improve application performance and reduce costs. For example, the study suggests that users on the cloud should focus on optimizing the virtual machine size and scaling rules to maximize scalability.

Furthermore, the study highlights the importance of performance evaluation in selecting cloud service providers. By providing objective information on the performance of cloud services, the study can guide businesses in making informed decisions that align with their specific infrastructure needs. The study also emphasizes the need for cloud service providers to improve their offerings to meet the demands of businesses and organizations.

The study offers valuable insights on the performance of cloud service providers in delivering infrastructure as a service. Its contributions to the body of knowledge on cloud infrastructure management and its recommendations on optimizing cloud infrastructure can guide businesses in making informed decisions that improve their application performance.

7.3 RECOMMENDATIONS FOR FUTURE RESEARCH

In the present study has investigated the performance evaluation of Infrastructure as a Service (IaaS) across three major Cloud Service Providers (CSPs) including Google Cloud Platform (GCP), Microsoft Azure, and Amazon Web Services (AWS). Although the study has provided significant insights into the performance of these CSPs, there are still some areas that can be further explored for future research.

Firstly, it is recommended that future research might examine the impact of different types of applications on the performance of CSPs. While the present study evaluated the performance of a web server, it would be interesting to explore the performance of other types of applications such as machine learning algorithms, database management systems, or gaming applications. This would provide a more comprehensive understanding of the capabilities and limitations of different CSPs for different types of applications.

Future research might focus on the impact of network latency on the performance of CSPs. The present study did not consider the impact of network latency on the performance evaluation, and it would be interesting to investigate how CSPs perform when the distance between the user and the CSP is increased. This would be particularly important for organizations that have global operations and need to ensure that their applications can be accessed efficiently from different regions.

If we examine the impact of different pricing models on the performance of CSPs. The present study did not consider the impact of pricing on the performance evaluation, and it would be interesting to investigate how CSPs perform when different pricing models are used. For example, some CSPs may offer better performance for applications with higher resource requirements, but it can result in a higher cost.

In addition, the impact of different security measures on the performance of CSPs is also curiosity. The present study did not consider the impact of security on the performance evaluation, and it would be interesting to investigate how CSPs perform when different security measures are implemented. This would be particularly important for organizations that handle sensitive data and need to ensure that their applications are secure.

However, Hybrid cloud environments are becoming increasingly popular, and it would be interesting to investigate how CSPs perform when they are used in conjunction with on-premises infrastructure. This would provide a more comprehensive understanding of the capabilities and limitations of different CSPs in hybrid cloud environments.

Moreover, Future research might examine the impact of different disaster recovery strategies on the performance of CSPs. The present study evaluated the performance of CSPs in recovery testing, but it would be interesting to investigate how CSPs perform when different disaster recovery strategies are used. This would be particularly important for organizations that need to ensure that their applications can be quickly recovered in the event of a disaster.

In conclusion, the present study has provided valuable insights into the performance evaluation of IaaS across CSPs. However, there are still several areas that can be further explored for future research, including the impact of different types of applications, network latency, pricing models, security measures, hybrid cloud environments, and disaster recovery strategies on the performance of CSPs. By investigating these areas, researchers can further enhance our understanding of the capabilities and limitations of different CSPs, and provide useful insights for organizations that are considering moving their applications to the cloud.

REFERENCES

[1] M. Eisa Suliman, “A Brief Analysis of Cloud Computing Infrastructure as a Service (IaaS)”, International Journal of Innovative Science and Research Technology, pp. 1325-1333, 2021, ISSN No:-2456-2165

[2] N. Chauhana, R. Agarwalb, K. Gargc, and T. Choudhuryd, “Redundant Iaas Cloud Selection With Consideration Of Multi Criteria Decision Analysis”, International Conference on Computational Intelligence and Data Science (ICCIDS 2019) , pp. 1409-1412, Chennai, India, 2019, DOI: 10.1016/j.procs.2020.03.448.

[3] S. Shahzadi, M. Iqbal, Z. Ul Qayyum, and T. Dagiuklas, “Infrastructure as a Service (IaaS): A Comparative Performance Analysis of Open-Source Cloud Platforms”, 2017 IEEE 22nd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Lund, Sweden, 2017, DOI: 10.1109/CAMAD.2017.8031522

[4] T. Ylonen, “SSH Key Management Challenges and Requirements”, 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Canary Islands, Spain, 2019, DOI: 10.1109/NTMS.2019.8763773

[5] S. Kaur, and T. Sharma, “Efficient load balancing using improved central load balancing technique”, 2018 2nd International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 2018, DOI: 10.1109/ICISC.2018.8398857

[6] V. Podolskiy, A. Jindal, and M. Gerndt, “IaaS Reactive Autoscaling Performance Challenges”, 2018 IEEE 11th International Conference on Cloud Computing, pp. 954-957, San Francisco, CA, USA, 2018, DOI: 10.1109/CLOUD.2018.00144

[7] J. Sithiyopasakul, T. Archevapanich, B. Purahong, P. Sithiyopasakul, and C. Benjangkaprasert, “Automated Resource Management System based on Kubernetes Technology”, 2021 18th International Conference on Electrical Engineering/Electronics, Computer,

Telecommunications and Information Technology (ECTI-CON), pp. 1146-1149, Chiang Mai, Thailand, 2021, DOI: 10.1109/ECTI-CON51831.2021.9454911

[8] H. Li, X. Li, H. Wang, J. Zhang, and Z. Jiang, “Research on Cloud Performance Testing Model”, 2019 IEEE 19th International Symposium on High Assurance Systems Engineering (HASE), pp. 179-183, Hangzhou, China, 2019, DOI: 10.1109/HASE.2019.00035

[9] R. Kumar Lenka, M. Rani Dey, P. Bhanse, and R. Kumar Barik, “Performance and Load Testing: Tools and Challenges”, 2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE), pp. 2257-2261, Bhubaneswar, India, 2018, DOI: 10.1109/ICRIEECE44171.2018.9009338

[10] R. Kumar Lenka, S. Mamgain, S. Kumar, and R. Kumar Barik, “Performance Analysis of Automated Testing Tools: JMeter and TestComplete”, 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), pp. 399-407, Greater Noida, India, 2019, DOI: 10.1109/ICACCCN.2018.8748521

APPENDIX

PUBLISHED RESEARCH ARTICLES

[1] S. Sithiyopasakul, T. Archevapanich, B. Purahong, P. Sithiyopasakul, A. Lasakul and C. Benjangkaprasert, " Performance Evaluation of Infrastructure as a Servic-e across Cloud Service Providers," 2023 International Electrical Engineering Congress (iEECON), 8-10 March, 2023, pp. 58-63, doi: 10.1109/iEECON56657.2023.10127100.

AUTHOR BIOGRAPHY

Name-Surname	Saran Sithiyopasakul
Date of birth	15 November 2542
Address	130/144 Ramkhamhaeng 43/1 Plubpla Wangthonglang Bangkok 10310 Tel. 097-234-0683
Education History	2021 School of Engineering, Computer Engineering, King Mongkut's Institute of Technology Ladkrabang