

PROFESSIONAL TRAINING REPORT
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award
of Bachelor of Technology Degree in Information Technology

By

NAREN.D (40120062)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119

APRIL 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC
Jeppiyaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344
www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the Bonafide work of **Naren.D(40120062)** who carried the project entitled "**Creation of static port VM with ping and apache server**" under my supervision from January 2023 to April 2023.

27/4/23

Internal Guide

Dr. A. Sivasangari M.E., Ph.D.,

for R. Subhashini
Head of the Department

Dr. R. Subhashini M.E., Ph.D.,

Submitted for Viva voce Examination held on 28.4.23

[Signature]

Internal Examiner

[Signature]

External Examiner

DECLARATION

I, Naren D hereby declare that the Project Report entitled "**Creation of static port VM with ping and apache server**" done by me under the guidance of **Dr. A. Sivasangari M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE: 28.4.23

PLACE: Chennai



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. R. Subhashini, M.E., Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. A. Sivasangari M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE

Ref.No.: 24 /ESPCC/BC001/04/2023



इलेक्ट्रॉनिक्स एवं
सूचना प्रौद्योगिकी मंत्रालय
MINISTRY OF
ELECTRONICS AND
INFORMATION TECHNOLOGY



COURSE COMPLETION LETTER

It is certified that

Naren D

Sathyabama Institute of Science and Technology

has successfully undergone the Bridge Course Training Program on

Cloud Computing Technology

Under "FutureSkills PRIME" Programme

commenced from 11 Mar, 2023


Shri. V. Sridharan

Chief Investigator
Cloud Computing Technology
Lead Resource Centre

L R PRAKASH Digitally signed by L R PRAKASH
Date: 2023.04.25 02:11:47
+05'30'

Cdr.(Retd.) L.R. Prakash
Senior Director
Centre Head
C-DAC, Chennai

ABSTRACT

In modern cloud computing environments, virtual machines (VMs) are widely used to run various applications and services. To ensure secure and controlled access to VMs, static ports can be created and attached to VMs, along with firewall rules that allow only specific types of traffic to pass through. This abstract outline the steps to create a static port, attach it to a VM, and apply firewall rules for ping and Apache server access. Firstly, a static port can be created in the cloud networking configuration, specifying a unique port number. This static port can be associated with a specific VM, allowing incoming traffic to be directed to the VM through this port.

Next, firewall rules can be applied to the VM's networking configuration to restrict access to only ping and Apache server traffic. For ping access, an ingress rule can be created to allow ICMP (Internet Control Message Protocol) traffic, which is used for ping requests, to pass through the static port. For Apache server access, an ingress rule can be created to allow HTTP (Hypertext Transfer Protocol) or HTTPS (HTTP Secure) traffic, depending on the configuration of the Apache server, to pass through the static port.

By implementing these steps, the VM will only allow ping and Apache server traffic to pass through the static port, while blocking other types of traffic, ensuring a secure and controlled environment for accessing the VM. This can be beneficial in scenarios where specific types of traffic need to be allowed, while others are restricted, based on the security and operational requirements of the environment.

TABLES OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	vi
	LIST OF FIGURES	ix
	LIST OF ABBREVIATION	x
1	INTRODUCTION	1
	1.1 Introduction	1
	1.2 Problem Statement	2
	1.3 Aim and Scope	2
	1.3.1 Aim	2
	1.3.2 Scope	2
	1.4 Objective	3
	1.5 Limitation	3
	1.6 Assumption and Hypothesis	4
	1.7 Conclusion	4
2	LITERATURE SURVEY	5
	2.1 Introduction	5
	2.2 Literature Review	5
	2.3 Conclusion	6
3	METHODOLOGY AND TECHNOLOGIES	7
	3.1 Development Design	7
	3.2 Methodology Phrase	8
	3.3 System Requirements	9
	3.3.1 Windows	9

	3.3.2 Web browser	9
	3.4 Application Technologies	10
	3.4.1 AWS	10
	3.4.2 AWS EC2	10
	3.5 Network Terminologies	11
	3.5.1 TCP AND UDP	11
	3.5.2 ICMP	13
	3.5.3 Firewall	14
	3.5.3.1 Packet Filtering	15
	3.5.3.2 Stateful Inspection	17
	3.5.3.3 Application Firewall	18
	3.5.3.4 Application Proxy Server	19
	3.5.3.5 Dedicated Proxy Server	21
	3.5.3.6 Virtual Private Network	22
	3.5.4 Network Access Control	24
	3.5.5 Universal Threat Management	24
4	RESULT AND DISCUSSION	25
	4.1 Sample Screenshots	25
5	SUMMARY AND CONCLUSION	31
	5.1 Conclusion	31
	5.2 Future Work	31
	REFERENCE	33

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
Fig 3.1	TCP and UDP	11
Fig 3.2	ICMP	13
Fig 3.3	Firewall	14
Fig 3.4	Packet Filtering	15
Fig 3.5	Virtual Private Network	22
Fig 5.1	EC2 Interface	25
Fig 5.2	Creating Instance	26
Fig 5.3	Creating Instance	26
Fig 5.4	Creating Instance	27
Fig 5.5	Running Instance	27
Fig 5.6	Security Group	28
Fig 5.7	Inbound Rules	28
Fig 5.8	Ping	29
Fig 5.9	Apache Server	29
Fig 5.10	AWS Management Console	30

LIST OF ABBREVIATION

TCP	Transmission Control Protocol
VM	Virtual Machine
AWS	Amazon Web Services
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
DNS	Domain Name System
VPN	Virtual Private Network

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION

This project is about creating a static port and attaching it to a Virtual Machine (VM), while applying firewall rules to allow only ping and Apache server access. In this project, we will be focusing on setting up a secure network configuration for a VM by creating a static port, associating it with the VM, and configuring firewall rules to allow only necessary network access for ping (ICMP) and Apache server (HTTP/HTTPS) traffic.

By following the steps in this project, you will learn how to create a static port, attach it to a VM, and implement firewall rules to restrict network access, ensuring that only authorized traffic is allowed to reach the VM. This will enhance the security of the VM by limiting potential attack vectors and reducing the risk of unauthorized access.

Firewall rulesets should be as specific as possible with regards to the network traffic they control. To create a ruleset involves determining what types of traffic are required, including protocols the firewall may need to use for management purposes. The details of creating rulesets vary widely by type of firewall and specific products, but many firewalls can have their performance improved by optimizing firewall rulesets. For example, some firewalls check traffic against rules in a sequential manner until a match is found; for these firewalls, rules that have the highest chance of matching traffic patterns should be placed at the top of the list wherever possible.

Throughout this project, we will provide clear and concise instructions, along with relevant examples, to guide you in setting up a secure network configuration for your VM. It is important to follow best practices for network security to protect your VM and ensure that it is accessible only by the necessary services while preventing any unauthorized access. So, let's get started and create a secure network setup for your VM by creating a static port and applying firewall rules for ping and Apache server access.

1.2 PROBLEM STATEMENT

In today's era of virtualization and cloud computing, securing virtual machines (VMs) is critical to prevent unauthorized access and potential security breaches. Many VMs require specific network access for certain services, such as ping (ICMP) and Apache server (HTTP/HTTPS), while restricting access to other ports and protocols.

The problem at hand is to create a secure network configuration for a VM by creating a static port and attaching it to the VM, and then applying firewall rules to allow only ping and Apache server access. This requires setting up a network configuration that allows only the necessary traffic to reach the VM while blocking all other traffic, ensuring that the VM is protected from potential security threats.

1.3 AIM AND SCOPE

1.3.1 Aim

The aim is to create a secure network configuration for a VM by creating a static port and attaching it to the VM, and then applying firewall rules to allow only ping and Apache server access. This requires setting up a network configuration that allows only the necessary traffic to reach the VM while blocking all other traffic, ensuring that the VM is protected from potential security threats.

1.3.2 Scope

This document seeks to assist organizations in understanding the capabilities of firewall technologies and firewall policies. It provides practical guidance on developing firewall policies and selecting, configuring, testing, deploying, and managing firewalls.

It's important to note that the scope of a static port on a VM is determined by the networking configuration and security settings implemented in the virtualization environment, such as virtual switches, firewalls, and routing rules. It's essential to follow best practices for network security and adhere to organizational policies when configuring static ports on VMs to ensure proper network isolation and protection of sensitive data.

1.4 OBJECTIVE

The main objective of creating a static port for a Virtual Machine (VM) is to assign a specific, predetermined port number to the VM's network interface. This allows for consistent and predictable communication with the VM over the network, as the assigned port number remains constant even if the VM is restarted or its IP address changes.

By using a static port, you can establish a fixed point of entry for network traffic to the VM, which can be useful for various purposes such as remote access, service communication, or network configuration. It also helps in managing network traffic and implementing firewall rules or security measures more effectively, as you can specify the allowed incoming or outgoing traffic based on the static port number.

Overall, the main objective of creating a static port for a VM is to enable reliable and controlled network communication to and from the VM using a specific port number, ensuring consistent access and security measures.

1.5 LIMITATION

Static port allocation on virtual machines (VMs) has certain limitations. Firstly, it may conflict with the dynamic port allocation mechanism used by the host operating system, which could result in port conflicts and disruptions in network communication. Secondly, the number of static ports available on a VM may be limited by the operating system or virtualization software, which could restrict the total number of simultaneous connections or services that can be hosted. Additionally, static port allocation on VMs may not be portable across different virtualization platforms or VM migration scenarios, as the port allocation may be tied to specific VM configurations or host systems. Lastly, managing and maintaining static port allocations across multiple VMs can be cumbersome and error-prone, as it requires careful coordination and configuration management to avoid conflicts and ensure proper network communication.

1.6 ASSUMPTION AND HYPOTHESIS

The researchers assume the following assumptions:

1. A static port assigned to a VM remains constant and does not change over time, unless manually modified or reconfigured.
2. A major assumption of survey research configuring a static port on a VM may improve security by limiting the number of open ports and reducing the risk of port scanning attacks.

The researchers identify the following hypothesis:

1. If a static port is assigned to a VM, it will consistently maintain the same port number, allowing for predictable communication and network configurations.
2. If a static port is assigned to a VM, it may reduce the attack surface by only allowing communication through a specific port, potentially enhancing security and mitigating potential vulnerabilities.

It is a tentative answer to your research question that has not yet been tested. For some research projects, you might have to write several hypotheses that address different aspects of your research question.

1.7 CONCLUSION

This chapter basically delivers the early stages about this project development. It explains more about the initial project development process.

CHAPTER-2

LITERATURE SURVEY

2.1 INTRODUCTION

A literature survey on static port virtual machines (VMs) with only ping and Apache access focuses on the use of VMs with static ports for networking and web server purposes. The survey aims to explore existing research, publications, and studies related to the configuration, deployment, and management of VMs with static ports, specifically those that allow only ping and Apache access. The survey may delve into various aspects, such as the motivations behind using static port VMs, the benefits and limitations of this approach, the security implications, performance considerations, and best practices for implementation. Additionally, the survey may cover case studies or real-world examples of organizations or researchers utilizing static port VMs for specific use cases, such as in testing, development, or production environments. The literature survey serves as a comprehensive overview of the existing body of knowledge on static port VMs with only ping and Apache access, providing a foundation for further research and exploration in this area.

2.2 LITERATURE REVIEW

A literature survey on static port virtual machines (VMs) that allow only ping and Apache server access reveals several findings.

First, static port VMs are a type of virtualization technology that enables the creation of isolated and self-contained virtual instances of an operating system, running on a physical host server. These VMs are configured with fixed or static ports, meaning that specific network ports are reserved exclusively for their use, allowing for consistent and predictable communication.

Second, static port VMs are often used in scenarios where network security is a critical concern. By utilizing fixed ports, these VMs can limit the surface area for potential cyber-attacks, reducing the risk of unauthorized access or data breaches. This makes them particularly useful in high-security environments or situations where

compliance requirements are stringent.

Third, static port VMs typically implement strict access controls, allowing only select network protocols or services to be enabled. In the case of ping and Apache server access, these VMs are configured to allow incoming ping requests for network connectivity testing, and incoming requests to the Apache server for serving web content. This granular control over network access helps to minimize the attack surface and reduce the risk of security vulnerabilities.

Several research studies have been conducted to evaluate the security benefits and performance implications of using static port VMs with limited network access. These studies have shown that restricting network access to only essential protocols, such as ping and Apache server, can effectively reduce the risk of security breaches and improve overall system security posture. Overall, the literature survey highlights various approaches for creating a static port and attaching it to a VM, along with the application of firewall rules to restrict access to specific services. These techniques provide enhanced security and accessibility to VMs in virtualized environments, ensuring that only the desired services are accessible while keeping other services protected from unauthorized access.

In conclusion, they provide isolated and predictable communication channels, implement strict access controls, and have been shown to improve security posture in research studies. Further research in this area may continue to explore the performance, scalability, and applicability of static port VMs in various use cases and deployment scenarios.

2.3 CONCLUSION

This chapter discusses literature review that had been reviewed during feasibility studies. The literature review helps developer to discover the problem of previous research or system which needs to be improves and overcome in this system development.

CHAPTER-3

METHODOLOGY AND TECHNOLOGIES

The project methodology is important because it helps to organize investigation in a scientific way to overcome problems, structure, plan, and control the process of developing an information system. After system has been completely developed, it should be tested to make sure it achieves the objectives of the project.

3.1 DEVELOPMENT DESIGN

When designing a development environment on a static port virtual machine (VM) with restricted network access limited to ping and Apache server, it's important to prioritize security. First, the VM should be configured with the necessary operating system and development software, keeping the installation minimal to minimize potential vulnerabilities. Firewall rules and network security groups should be implemented to allow only incoming and outgoing traffic for ping (ICMP) and Apache server access (HTTP/S) on the designated static ports. Access should be restricted to trusted IP addresses or networks, and regular audits should be conducted to ensure the rules are up-to-date and aligned with the development requirements. Access to the VM should be strictly limited to authorized developers using strong authentication mechanisms, such as secure usernames and passwords or public key authentication. Two-factor authentication should be considered for an additional layer of security. Regular review and update of user accounts and access privileges are crucial to ensure that only necessary access is granted.

Monitoring and logging should be enabled to detect any potential security incidents. Log files, including system logs and web server logs, should be regularly analyzed for any signs of suspicious activities or anomalies. Intrusion detection and prevention systems (IDPS) can be deployed to provide real-time monitoring and alerting on potential security threats. Regular backups of the development VM should be performed to protect against data loss due to hardware failures, accidents, or security breaches. Backups should be stored securely and offsite to ensure data integrity and availability. Keeping the VM and its software up-to-date with the latest security

patches and updates is critical. Regular security audits and vulnerability assessments should be conducted to identify and address any potential weaknesses in the VM's configuration and setup, including timely updates for the Apache server and other software used for development.

By implementing these security measures during the design phase of a development environment on a static port VM with limited network access, developers can ensure a secure environment that allows only ping and Apache server access, reducing the risk of unauthorized access or security breaches and safeguarding the confidentiality, integrity, and availability of the development environment.

3.2 METHODOLOGY PHRASE

Static port virtual machines (VMs) are a type of virtualization technology that allows for the creation of virtual instances of a server with pre-assigned, fixed network ports. These VMs are isolated from the host system and can have their own operating system, software, and network configurations. When configuring static port VMs, several methodologies can be followed to ensure proper network access and security. One common approach is to restrict inbound and outbound traffic to only necessary protocols, such as ICMP for ping and HTTP/S for Apache server access. This can be achieved through firewall rules and network security groups. Additionally, the VMs can be configured with minimal installed software and services to reduce potential attack vectors. Unnecessary services and ports should be disabled or closed to minimize the attack surface.

Regular security updates and patches should also be applied to keep the VM and its software up-to-date and protected against known vulnerabilities. Access to the static port VMs should be restricted to authorized users or systems only. Authentication mechanisms, such as secure usernames and passwords or public key authentication, should be implemented to ensure that only authorized users can access the VMs. Two-factor authentication can provide an additional layer of security. Monitoring and logging should also be implemented to detect any potential security breaches. Log files, including system logs, web server logs, and firewall logs, should be analyzed

regularly for any suspicious activities or anomalies. Intrusion detection and prevention systems (IDPS) can be deployed to provide real-time monitoring and alerting on potential security threats. Regular backups of the static port VMs should be performed to ensure data integrity and availability. Backups should be stored securely and offsite to protect against data loss due to hardware failures, accidents, or security breaches. Finally, regular security audits and vulnerability assessments should be conducted to identify and address any potential weaknesses in the static port VMs' configuration and setup. These audits can help identify any vulnerabilities that may have been missed and provide recommendations for further security enhancements.

By following these methodologies, organizations can ensure that their static port VMs are configured securely, allowing only ping and Apache server access while minimizing the risk of unauthorized access or security breaches.

3.3 SYSTEM REQUIREMENTS

There some System Requirement for creating static port VMs with only ping and Apache server access. Here are some System Requirements:

3.3.1 Windows

- Microsoft Windows 8/10 64-bit.
- Memory (RAM): Minimum 1 GB; Recommended 4GB or above.
- Hard Drive: Minimum 32 GB; Recommended 64 GB or more.
- Processor: Minimum 1GHz; Recommended 2GHz or more...
- Ethernet connection (LAN) or a Wireless adapter (Wi-Fi).

3.3.2 Web browser

HTML5-capable web browsers that can be used include the following:

- Google Chrome
- Mozilla Firefox
- Safari
- Microsoft Edge

3.4 APPLICATION TECHNOLOGIES

3.4.1 AWS

When deploying an application on a static port virtual machine (VM) with restricted network access limited to ping and Apache server on Amazon Web Services (AWS), organizations can utilize various technologies to achieve the desired functionality. AWS provides a range of services that can be used in combination to achieve this, including Amazon EC2 for VM deployment, Amazon VPC for network configuration, IAM for access control, Amazon CloudWatch for monitoring and logging, Amazon S3 and Amazon EBS for backup and disaster recovery, and regular security patching and updates for the VM and software used, including Apache server. By leveraging these AWS technologies, organizations can create a secure environment that allows only ping and Apache server access, ensuring the confidentiality, integrity, and availability of their application while minimizing the risk of unauthorized access or security breaches.

3.4.2 AWS EC2

Amazon EC2 (Elastic Compute Cloud) is a cloud computing service provided by Amazon Web Services (AWS) that allows businesses to run virtual machines in the cloud. With EC2, businesses can easily deploy and manage virtual servers to meet their computing needs. EC2 offers efficient resource management, allowing businesses to scale their compute capacity up or down based on demand, optimizing costs while maximizing performance. It also provides flexible deployment options, with a variety of instance types and operating systems to choose from, as well as options for high availability through multiple availability zones.

EC2 simplifies operations with easy management and monitoring tools, including a web-based console, API, and command-line interface, allowing businesses to monitor performance metrics, set up automated backups, and configure security groups. It also integrates with other AWS services, enabling businesses to create a comprehensive cloud computing solution tailored to their needs.

Security is a priority with EC2, offering features such as virtual private clouds (VPCs) for isolation, data encryption at rest with AWS Key Management Service (KMS), and security groups for access control. EC2 also includes built-in monitoring and automatic recovery features for high availability and reliability.

In conclusion, Amazon EC2 is a powerful cloud computing solution that provides businesses with scalability, flexibility, easy management, and robust security features. Whether for small startups or large enterprises, EC2 empowers businesses to build and deploy applications in the cloud with confidence, optimizing costs and simplifying operations.

3.5 NETWORK TERMINOLOGIES

Server-side programming allows us to instead store the information in a database and dynamically construct and return HTML and other types of files (e.g. PDFs, images, etc.).

3.5.1 TCP and UDP

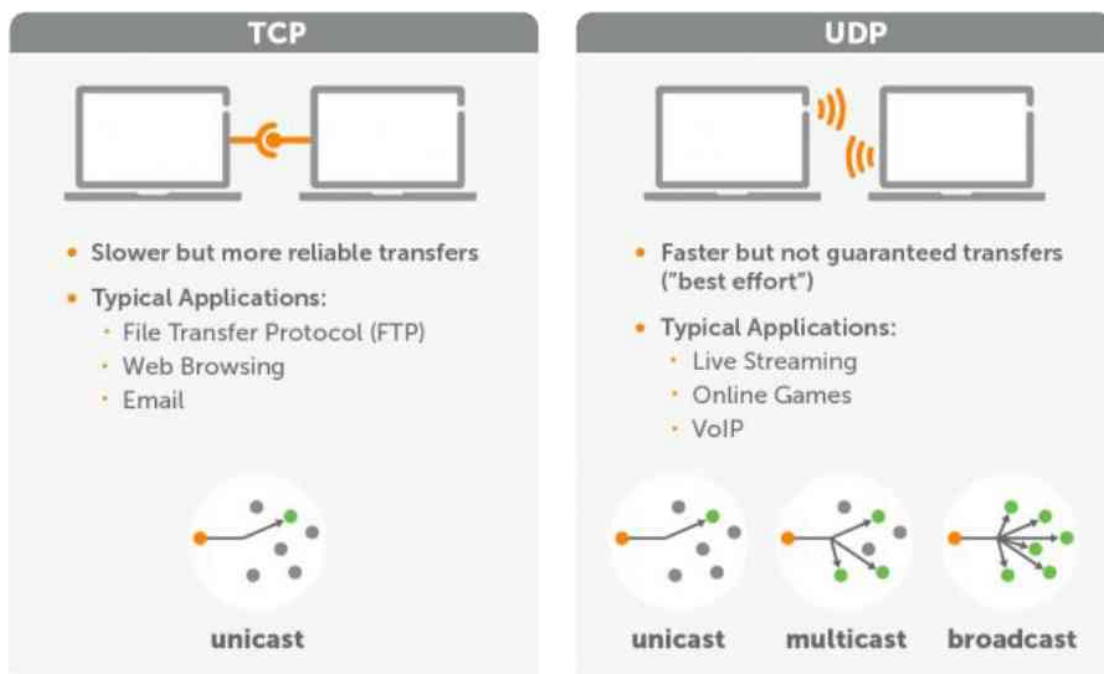


Fig 3.1 TCP and UDP

Application protocols can use TCP, UDP, or both, depending on the design of the protocol. An application server typically listens on one or more fixed TCP or UDP ports. Some applications use a single port, but many applications use multiple ports. For example, although SMTP uses TCP port 25 for sending mail, it uses TCP port 587 for mail submission. Similarly, FTP uses at least two ports, one of which can be unpredictable, and while most web servers use only TCP port 80, it is common to have web sites that also use additional ports such as TCP port 8080.

Application clients typically use any of a wide range of ports. As with other aspects of firewall rulesets, deny by default policies should be used for incoming TCP and UDP traffic. Less stringent policies are generally used for outgoing TCP and UDP traffic because most organizations permit their users to access a wide range of external applications located on millions of external hosts. In addition to allowing and blocking UDP and TCP traffic, many firewalls are also able to report or block malformed UDP and TCP traffic directed towards the firewall or to hosts protected by the firewall. This traffic is frequently used to scan for hosts, and may also be used in certain types of attacks. The firewall can help block such activity—or at least report when such activity is happening.

To allow only Apache server access on a system, you can configure the firewall settings to permit incoming connections on the specific ports used by Apache for HTTP (port 80) or HTTPS (port 443) traffic, while blocking all other incoming connections. There are some commands to configure the firewall to allow incoming connections on ports 80 and 443, which are commonly used by Apache for HTTP and HTTPS traffic, respectively. All other incoming connections will be dropped, effectively blocking access to any other services or ports on the system.

3.5.2 ICMP

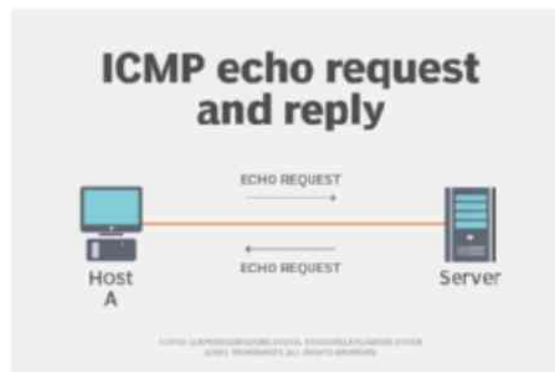


Fig 3.2 ICMP

ICMP (Internet Control Message Protocol) is a network protocol that is commonly used for sending diagnostic messages and error reporting between network devices. One of the most well-known uses of ICMP is for ping, which is a simple network troubleshooting tool that sends ICMP Echo Request messages to a target device and waits for ICMP Echo Reply messages in response to determine the availability and round-trip time of the target device.

Ping is widely used to test the connectivity and reachability of network devices, such as servers, routers, and switches. It works by sending ICMP Echo Request messages from the source device to the destination device, and the destination device responds with ICMP Echo Reply messages if it is online and reachable. Ping can be used to verify if a network device is up and running, measure network latency or round-trip time, and identify potential network connectivity issues. ICMP and ping can be configured and controlled in various ways, including allowing or blocking ICMP messages at the firewall level, setting specific ICMP types and codes to be allowed or blocked, and configuring ICMP rate limits to prevent network flooding or denial of service attacks. Properly configuring ICMP and ping can help organizations monitor and troubleshoot their network devices effectively while also ensuring security and protecting against potential vulnerabilities or attacks.

ICMP is often used by low-level networking protocols to increase the speed and reliability of networking. Therefore, ICMP within an organization's network generally

should not be blocked by firewalls that are not at the perimeter of the network, unless security needs outweigh network operational needs. Similarly, if an organization has more than one network, ICMP that comes from or goes to other networks within the organization should not be blocked.

3.5.3 Firewall

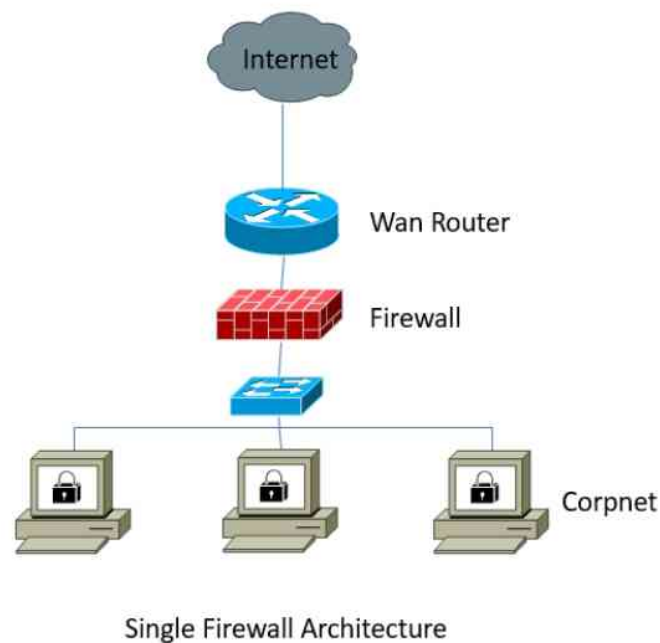


Fig 3.3 Firewall

Firewalls are devices or programs that control the flow of network traffic between networks or hosts that employ differing security postures. While firewalls are often discussed in the context of Internet connectivity, they may also have applicability in other network environments. For example, many enterprise networks employ firewalls to restrict connectivity to and from the internal networks used to service more sensitive functions, such as accounting or personnel. By employing firewalls to control connectivity to these areas, an organization can prevent unauthorized access to its systems and resources. Inclusion of a proper firewall provides an additional layer of security. Organizations often need to use firewalls to meet security requirements from mandates (e.g., FISMA); some mandates, such as the Payment Card Industry (PCI)

Data Security Standard,1 specifically require firewalling.

Several types of firewall technologies are available. One way of comparing their capabilities is to look at the Transmission Control Protocol/Internet Protocol (TCP/IP) layers that each is able to examine. TCP/IP communications are composed of four layers that work together to transfer data between hosts. When a user wants to transfer data across networks, the data is passed from the highest layer through intermediate layers to the lowest layer, with each layer adding more information. The lowest layer sends the accumulated data through the physical network, with the data then passed upwards through the layers to its destination. Simply put, the data produced by a layer is encapsulated in a larger container by the layer below it.

3.5.3.1 Packet Filtering

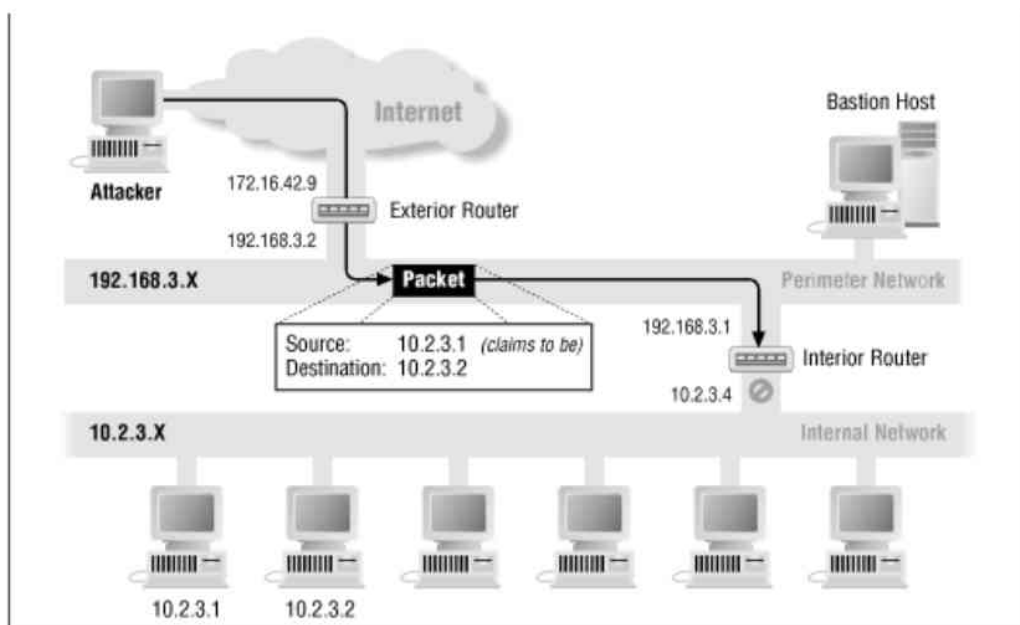


Fig 3.4 Packet Filtering

The most basic feature of a firewall is the packet filter. Older firewalls that were only packet filters were essentially routing devices that provided access control functionality for host addresses and communication sessions. These devices, also known as stateless inspection firewalls, do not keep track of the state of each flow of traffic that passes through the firewall; this means, for example, that they cannot

associate multiple requests within a single session to each other. Packet filtering is at the core of most modern firewalls, but there are few firewalls sold today that only do stateless packet filtering. Unlike more advanced filters, packet filters are not concerned about the content of packets. Their access control functionality is governed by a set of directives referred to as a ruleset.

Packet filtering capabilities are built into most operating systems and devices capable of routing; the most common example of a pure packet filtering device is a network router that employs access control lists. In their most basic form, firewalls with packet filters operate at the network layer. This provides network access control based on several pieces of information contained in a packet, including: The packet's source IP address—the address of the host from which the packet originated (such as 192.168.1.1) The packet's destination address—the address of the host the packet is trying to reach (e.g., 192.168.2.1). The network or transport protocol being used to communicate between source and destination hosts, such as TCP, UDP, or ICMP Possibly some characteristics of the transport layer communications sessions, such as session source and destination ports (e.g., TCP 80 for the destination port belonging to a web server, TCP 1320 for the source port belonging to a personal computer accessing the server) The interface being traversed by the packet, and its direction (inbound or outbound). Filtering inbound traffic is known as ingress filtering.

Outgoing traffic can also be filtered, a process referred to as egress filtering. Here, organizations can implement restrictions on their internal traffic, such as blocking the use of external file transfer protocol (FTP) servers or preventing denial of service (DoS) attacks from being launched from within the organization against outside entities. Organizations should only permit outbound traffic that uses the source IP addresses in use by the organization—a process that helps block traffic with spoofed addresses from leaking onto other networks. Spoofed addresses can be caused by malicious events such as malware infections or compromised hosts being used to launch attacks, or by inadvertent misconfigurations. Stateless packet filters are generally vulnerable to attacks and exploits that take advantage of problems within the TCP/IP specification and protocol stack. Firewalls that operate at higher layers

can thwart some spoofing attacks by verifying that a session is established, or by authenticating users before allowing traffic to pass. Because of this, most firewalls that use packet filters also maintain some state information for the packets that traverse the firewall. Some packet filters can specifically filter packets that are fragmented. Packet fragmentation is allowed by the TCP/IP specifications and is encouraged in situations where it is needed. However, packet fragmentation has been used to make some attacks harder to detect (by placing them within fragmented packets), and unusual fragmentation has also been used as a form of attack. For example, some network-based attacks have used packets that should not exist in normal communications, such as sending some fragments of a packet but not the first fragment, or sending packet fragments that overlap each other.

3.5.3.2 Stateful Inspection

Stateful inspection improves on the functions of packet filters by tracking the state of connections and blocking packets that deviate from the expected state. This is accomplished by incorporating greater awareness of the transport layer. As with packet filtering, stateful inspection intercepts packets at the network layer and inspects them to see if they are permitted by an existing firewall rule, but unlike packet filtering, stateful inspection keeps track of each connection in a state table. While the details of state table entries vary by firewall product, they typically include source IP address, destination IP address, port numbers, and connection state information. Three major states exist for TCP traffic—connection establishment, usage, and termination (which refers to both an endpoint requesting that a connection be closed and a connection with a long period of inactivity.) Stateful inspection in a firewall examines certain values in the TCP headers to monitor the state of each connection. Each new packet is compared by the firewall to the firewall's state table to determine if the packet's state contradicts its expected state. For example, an attacker could generate a packet with a header indicating it is part of an established connection, in hopes it will pass through a firewall. If the firewall uses stateful inspection, it will first verify that the packet is part of an established connection listed in the state table. In the simplest case, a firewall will allow through any packet that

seems to be part of an open connection (or even a connection that is not yet fully established). However, many firewalls are more cognizant of the state machines for protocols such as TCP and UDP, and they will block packets that do not adhere strictly to the appropriate state machine.

For these protocols, most firewalls with stateful inspection are only able to track the source and destination IP addresses and ports. UDP packets must still match an entry in the state table based on source and destination IP address and port information to be permitted to pass—a DNS response from an external source would be permitted to pass only if the firewall had previously seen a corresponding DNS query from an internal source. Since the firewall is unable to determine when a session has ended, the entry is removed from the state table after a preconfigured timeout value is reached. Application-level firewalls that are able to recognize DNS over UDP will terminate a session after a DNS response is received, and may act similarly with the Network Time Protocol (NTP).

3.5.3.3 Application Firewall

A newer trend in stateful inspection is the addition of a stateful protocol analysis capability, referred to by some vendors as deep packet inspection. Stateful protocol analysis improves upon standard stateful inspection by adding basic intrusion detection technology—an inspection engine that analyzes protocols at the application layer to compare vendor-developed profiles of benign protocol activity against observed events to identify deviations. This allows a firewall to allow or deny access based on how an application is running over the network. For instance, an application firewall can determine if an email message contains a type of attachment that the organization does not permit (such as an executable file), or if instant messaging (IM) is being used over port 80 (typically used for HTTP). Another feature is that it can block connections over which specific actions are being performed (e.g., users could be prevented from using the FTP “put” command, which allows users to write files to the FTP server). This feature can also be used to allow or deny web pages that contain particular types of active content, such as Java or ActiveX, or that have SSL certificates signed by a particular certificate authority (CA), such as a compromised or

revoked CA. Application firewalls can enable the identification of unexpected sequences of commands, such as issuing the same command repeatedly or issuing a command that was not preceded by another command on which it is dependent. These suspicious commands often originate from buffer overflow attacks, DoS attacks, malware, and other forms of attack carried out within application protocols such as HTTP. Another common feature is input validation for individual commands, such as minimum and maximum lengths for arguments. For example, a username argument with a length of 1000 characters is suspicious—even more so if it contains binary data.

Application firewalls are available for many common protocols including HTTP, database (such as SQL), email (SMTP, Post Office Protocol [POP], and Internet Message Access Protocol [IMAP]), voice over IP (VoIP), and Extensible Markup Language (XML).⁴ Another feature found in some application firewalls involves enforcing application state machines, which are essentially checks on the traffic's compliance to the standard for the protocol in question. This compliance checking, sometimes call "RFC compliance" because most protocols are defined in RFCs issued by the Internet Engineering Task Force (IETF), can be a mixed blessing. Many products implement protocols in ways that almost, but not completely, match the specification, so it is usually necessary to let such implementations communicate across the firewall. Compliance checking is only useful when it detects and blocks communication that can be harmful to protected systems. Firewalls with both stateful inspection and stateful protocol analysis capabilities are not full-fledged intrusion detection and prevention systems (IDPS), which usually offer much more extensive attack detection and prevention capabilities.

3.5.3.4 Application Proxy Server

An application-proxy gateway is a feature of advanced firewalls that combines lower-layer access control with upper-layer functionality. These firewalls contain a proxy agent that acts as an intermediary between two hosts that wish to communicate with each other, and never allows a direct connection between them. Each successful connection attempt actually results in the creation of two separate connections—one

between the client and the proxy server, and another between the proxy server and the true destination. The proxy is meant to be transparent to the two hosts—from their perspectives there is a direct connection. Because external hosts only communicate with the proxy agent, internal IP addresses are not visible to the outside world. The proxy agent interfaces directly with the firewall ruleset to determine whether a given instance of network traffic should be allowed to transit the firewall.

In addition to the ruleset, some proxy agents have the ability to require authentication of each individual network user. This authentication can take many forms, including user ID and password, hardware or software token, source address, and biometrics. Like application firewalls, the proxy gateway operates at the application layer and can inspect the actual content of the traffic. These gateways also perform the TCP handshake with the source system and are able to protect against exploitations at each step of a communication. In addition, gateways can make decisions to permit or deny traffic based on information in the application protocol headers or payloads. Once the gateway determines that data should be permitted, it is forwarded to the destination host.

Application-proxy gateways are quite different than application firewalls. First, an application-proxy gateway can offer a higher level of security for some applications because it prevents direct connections between two hosts and it inspects traffic content to identify policy violations. Another potential advantage is that some application-proxy gateways have the ability to decrypt packets (e.g., SSL-protected payloads), examine them, and re-encrypt them before sending them on to the destination host. Data that the gateway cannot decrypt is passed directly through to the application. When choosing the type of firewall to deploy, it is important to decide whether the firewall actually needs to act as an application proxy so that it can match the specific policies needed by the organization. Firewalls with application-proxy gateways can also have several disadvantages when compared to packet filtering and stateful inspection. First, because of the “full packet awareness” of application-proxy gateways, the firewall spends much more time reading and interpreting each packet. Because of this, some of these gateways are poorly suited to high-bandwidth

or real-time applications—but application-proxy gateways rated for high bandwidth are available. To reduce the load on the firewall, a dedicated proxy server can be used to secure less time-sensitive services such as email and most web traffic. Another disadvantage is that application-proxy gateways tend to be limited in terms of support for new network applications and protocols—an individual, application-specific proxy agent is required for each type of network traffic that needs to transit a firewall. Many application-proxy gateway firewall vendors provide generic proxy agents to support undefined network protocols or applications. Those generic agents tend to negate many of the strengths of the application-proxy gateway architecture because they simply allow traffic to “tunnel” through the firewall.

3.5.3.5 Dedicated Proxy Server

Dedicated proxy servers differ from application-proxy gateways in that while dedicated proxy servers retain proxy control of traffic, they usually have much more limited firewalling capabilities. They are described in this section because of their close relationship to application-proxy gateway firewalls. Many dedicated proxy servers are application-specific, and some actually perform analysis and validation of common application protocols such as HTTP. Because these servers have limited firewalling capabilities, such as simply blocking traffic based on its source or destination, they are typically deployed behind traditional firewall platforms. Typically, a main firewall could accept inbound traffic, determine which application is being targeted, and hand off traffic to the appropriate proxy server (e.g., email proxy). This server would perform filtering or logging operations on the traffic, then forward it to internal systems. A proxy server could also accept outbound traffic directly from internal systems, filter or log the traffic, and pass it to the firewall for outbound delivery. An example of this is an HTTP proxy deployed behind the firewall—users would need to connect to this proxy en-route to connecting to external web servers.

Dedicated proxy servers are generally used to decrease firewall workload and conduct specialized filtering and logging that might be difficult to perform on the firewall itself. In recent years, the use of inbound proxy servers has decreased dramatically. This is because an inbound proxy server must mimic the capabilities of

the real server it is protecting, which becomes nearly impossible when protecting a server with many features. Using a proxy server with fewer capabilities than the server it is protecting renders the non-matched capabilities unusable. Additionally, the essential features that inbound proxy servers should have (logging, access control, etc.) are usually built into the real servers. Most proxy servers now in use are outbound proxy servers, with the most common being HTTP proxies. The HTTP proxy would handle outbound connections to external web servers and possibly filter for active content. Requests from users first go to the proxy, and the proxy then sends the request (possibly changed) to the outside web server. The response from that web server then comes back to the proxy, which relays it to the user.

3.5.3.6 Virtual Private Network

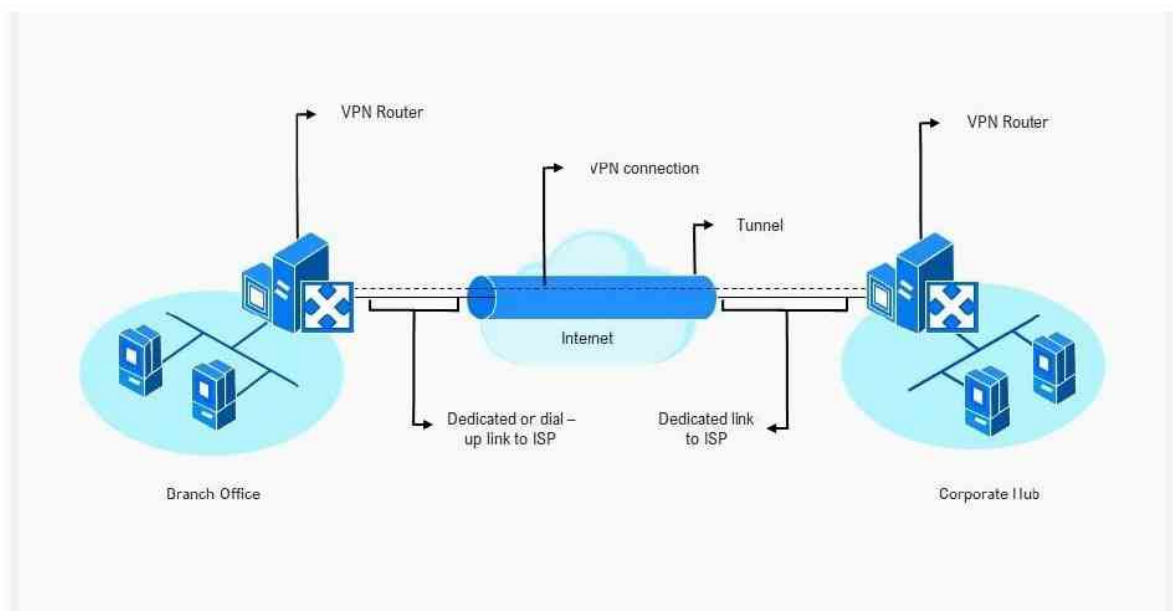


Fig 3.5 Virtual Private Network

Firewall devices at the edge of a network are sometimes required to do more than block unwanted traffic. A common requirement for these firewalls is to encrypt and decrypt specific network traffic flows between the protected network and external networks. This nearly always involves virtual private networks (VPN), which use additional protocols to encrypt traffic and provide user authentication and integrity checking. VPNs are most often used to provide secure network communications

across untrusted networks. Two common choices for secure VPNs are IPsec6 and Secure Sockets Layer (SSL)/Transport Layer Security (TLS). The two most common VPN architectures are gateway-to-gateway and host-to-gateway.

Gateway-to-gateway architectures connect multiple fixed sites over public lines through the use of VPN gateways—for example, to connect branch offices to an organization's headquarters. A VPN gateway is usually part of another network device such as a firewall or router. When a VPN connection is established between the two gateways, users at branch locations are unaware of the connection and do not require any special settings on their computers. The second type of architecture, host-to-gateway, provides a secure connection to the network for individual users, usually called remote users, who are located outside of the organization (at home, in a hotel, etc.) Here, a client on the user machine negotiates the secure connection with the organization's VPN gateway.

For gateway-to-gateway and host-to-gateway VPNs, the VPN functionality is often part of the firewall itself. Placing it behind the firewall would require VPN traffic to be passed through the firewall while encrypted, preventing the firewall from inspecting the traffic. All remote access (host-to-gateway) VPNs allow the firewall administrator to decide which users have access to which network resources. This access control is normally available on a per-user and per-group basis; that is, the VPN policy can specify which users and groups are authorized to access which resources, should an organization need that level of granularity. VPNs generally rely on authentication protocols such as Remote Authentication Dial In User Service (RADIUS). RADIUS uses several different types of authentication credentials, with the most common examples being username and password, digital signatures, and hardware tokens. Another authentication protocol often used by VPNs is the Lightweight Directory Access Protocol (LDAP); it is particularly useful for making access decisions for individual users and groups. To run VPN functionality on a firewall requires additional resources that depend on the amount of traffic flowing across the VPN and the type of encryption being used. Many firewalls include hardware acceleration for encryption to minimize the impact of VPN services.

3.5.4 Network Access Control

Another common requirement for firewalls at the edge of a network is to perform client checks for incoming connections from remote users and allow or disallow access based on those checks. This checking, commonly called network access control (NAC) or network access protection (NAP), allows access based on the user's credentials and the results of performing "health checks" on the user's computer. Health checks typically consist of verifying that one or more of the following comply with organizational policy.

- Latest updates to antimalware and personal firewall software
- Configuration settings for antimalware and personal firewall software
- Elapsed time since the previous malware scan
- Patch level of the operating system and selected applications
- Security configuration of the operating system and selected applications

These health checks require software on the user's system that is controlled by the firewall. If the user has acceptable credentials but the device does not pass the health check, the user and device may get only limited access to the internal network for remediation purposes.

3.5.5 Universal Threat Management

Many firewalls combine multiple features into a single system, the idea being that it is easier to set and maintain policy on a single system than on many systems that are deployed at the same location on a network. A typical unified threat management (UTM) system has a firewall, malware detection and eradication, sensing and blocking of suspicious network probes, and so on. There are pros and cons to merging multiple, not-completely-related functions into a single system. For example, deploying a UTM reduces complexity by making a single system responsible for multiple security objectives, but it also requires that the UTM have all the desired features to meet every one of the objectives. Another tradeoff is in performance: a single system handling multiple tasks has to have enough resources such as CPU speed and memory to handle every task assigned to it.

CHAPTER-4

RESULT AND DISCUSSION

4.1 SAMPLE SCREENSHOTS

Implementing and testing are the important parts in order to produce the good quality of system. It refers to the process of testing implementations of technology specifications. Technique adopted for the evaluation and testing of static port VM is test case testing which test the system functionalities modules by modules to ensure that the software confirms to its specification and meets the requirements. Below are the sample screenshots of the developed system.

The following diagram shows the EC2 Interface.

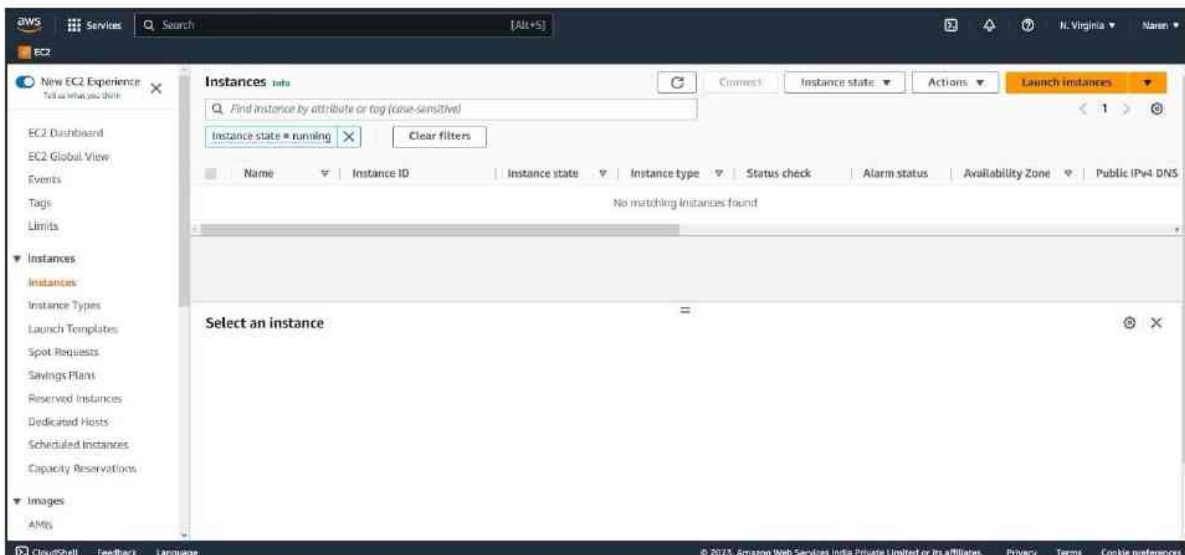


Fig 5.1 EC2 Interface

The following diagrams show the creation of instance.

The screenshot shows the AWS Management Console interface for creating an EC2 instance. The top navigation bar includes the AWS logo, 'Services', a search bar, and a user profile icon. The left sidebar shows the 'EC2' icon. The main content area is titled 'Search our full catalog including 1000s of application and OS images'. Below this is a 'Quick Start' section with a grid of operating system logos: Amazon Linux, macOS, Ubuntu, Windows, and Red Hat. A 'Browse more AMIs' button is also present. The 'Amazon Linux 2023 AMI' is selected, showing its ID (ami-06e46074ae430fba6), architecture (64-bit x86), and boot mode (uefi-preferred). A 'Verified provider' badge is visible. The right sidebar contains a 'Summary' section with fields for 'Number of instances' (set to 1), 'Software Image (AMI)', 'Virtual server type (instance type)' (t2.micro), 'Firewall (security group)' (New security group), and 'Storage (volumes)' (1 volume(s) - 8 GiB). At the bottom right, there are 'Cancel' and 'Launch instance' buttons, with a 'Review commands' link below the launch button.

Fig 5.2 Creating Instance

The screenshot shows the second step of the 'Create Instance' wizard in the AWS Management Console. The top navigation bar and left sidebar are consistent with the previous screenshot. The main content area is titled 'On-Demand SUSE pricing: 0.0124 USD per Hour'. Below this is a 'Key pair (login)' section with a dropdown menu showing 'demo' and a 'Create new key pair' button. The 'Network settings' section is expanded, showing 'Network' (vpc-02b74d688d1de03f1), 'Subnet' (No preference (Default subnet in any availability zone)), 'Auto-assign public IP' (Enabled), and 'Firewall (security groups)' (New security group). The right sidebar 'Summary' section is updated to show 'Software Image (AMI)' as 'Amazon Linux 2023 AMI 2023.0.2...' and 'Storage (volumes)' as '1 volume(s) - 8 GiB'. The 'Launch instance' button is now orange, and the 'Review commands' link is visible below it.

Fig 5.3 Creating Instance

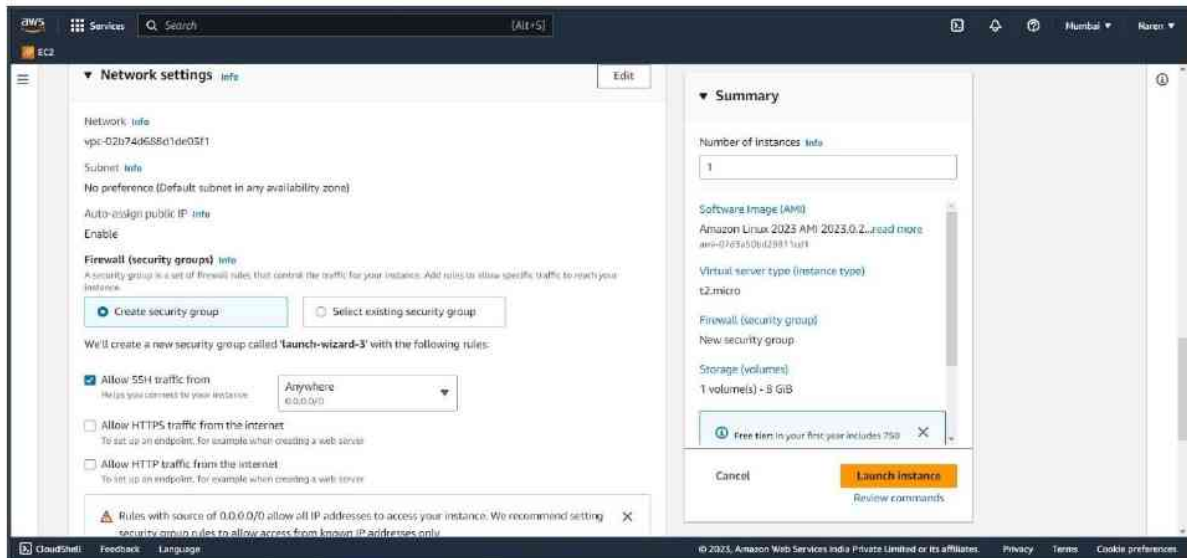


Fig 5.4 Creating Instance

The following diagram shows the running instances.

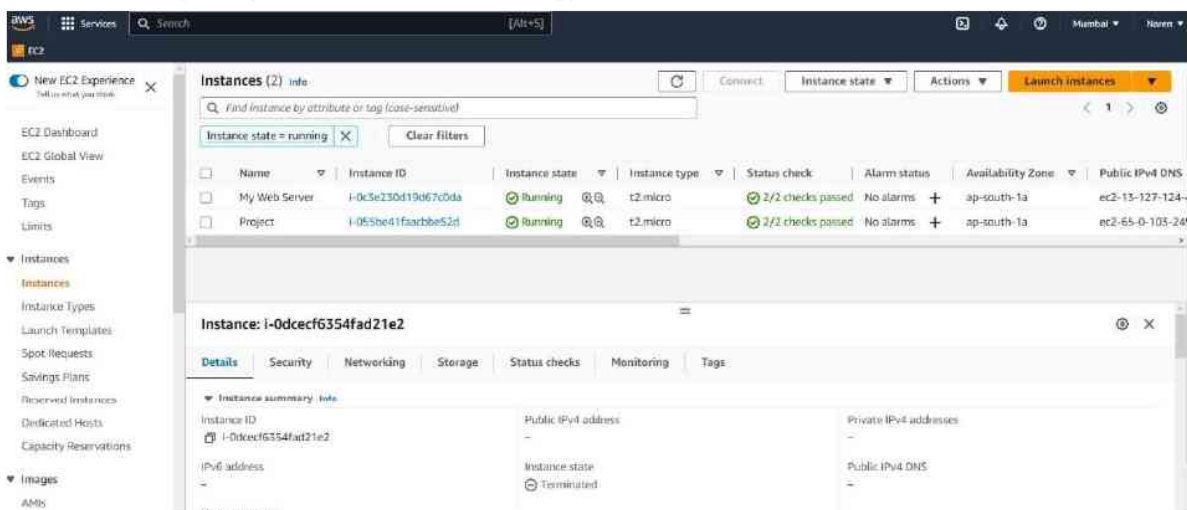


Fig 5.5 Running Instance

The following diagram shows the security group of the instance.

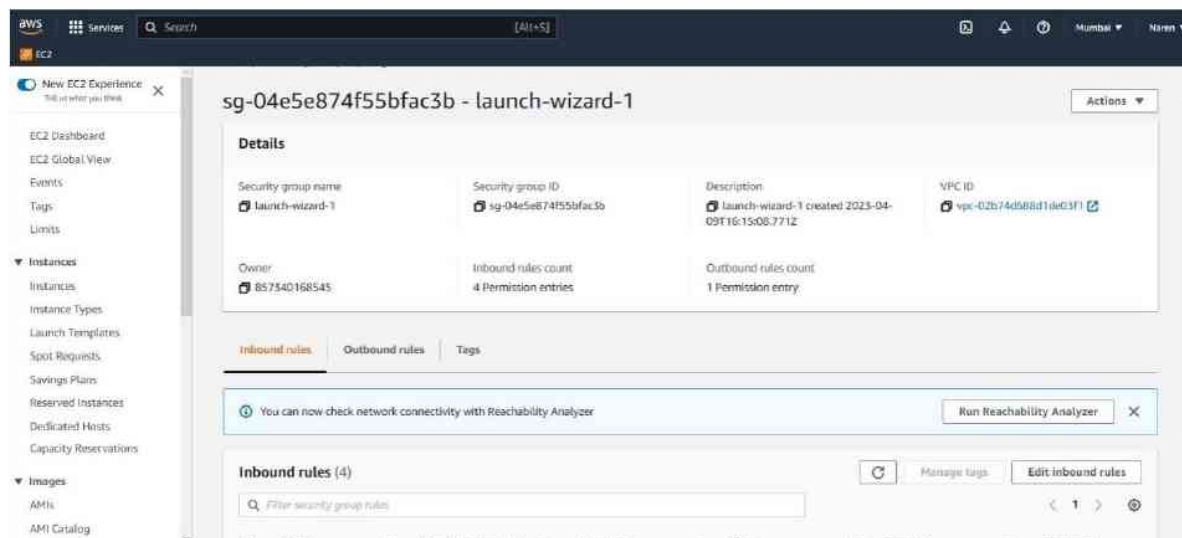


Fig 5.6 Security Group

The following diagram shows the inbound rules of the instance.

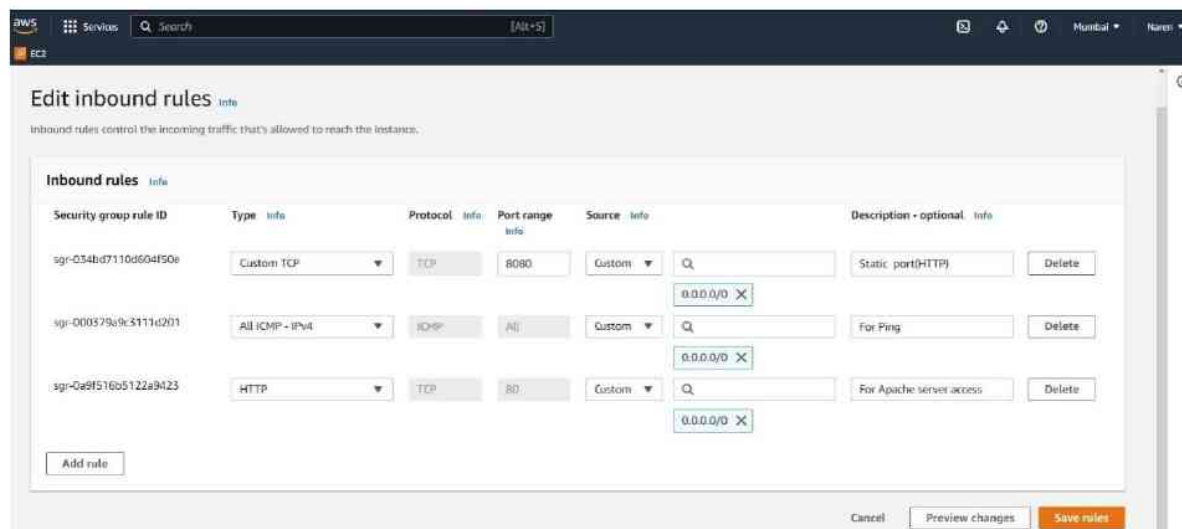
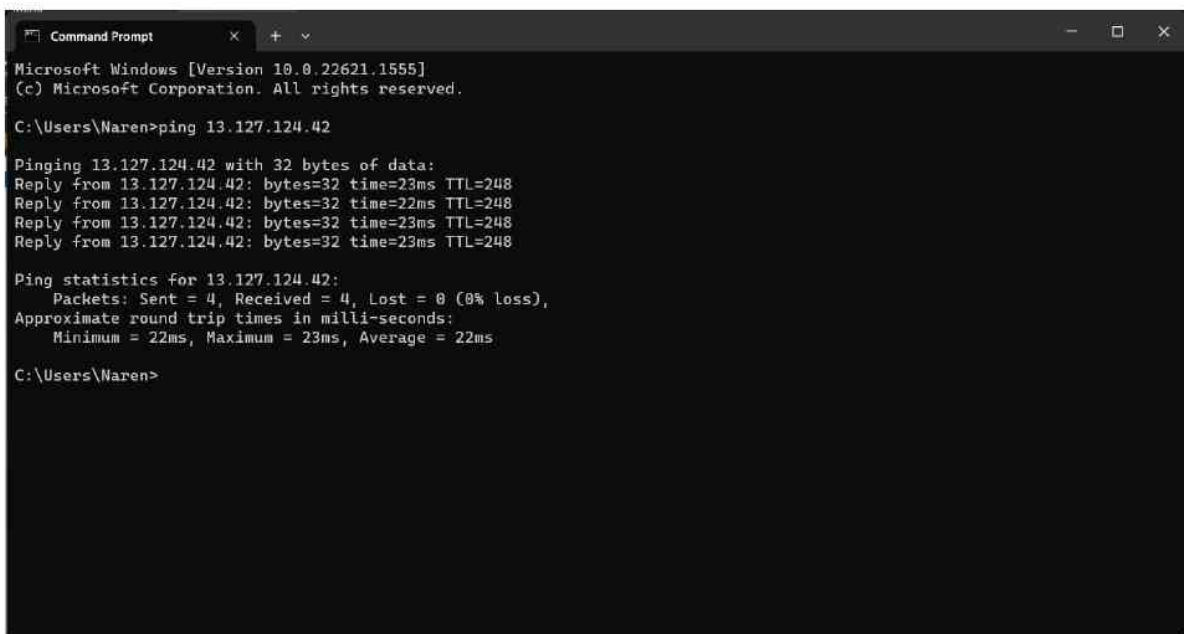


Fig 5.7 Inbound Rules

The following diagram shows the pinging of instance from other device.



```
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Naren>ping 13.127.124.42

Pinging 13.127.124.42 with 32 bytes of data:
Reply from 13.127.124.42: bytes=32 time=23ms TTL=248
Reply from 13.127.124.42: bytes=32 time=22ms TTL=248
Reply from 13.127.124.42: bytes=32 time=23ms TTL=248
Reply from 13.127.124.42: bytes=32 time=23ms TTL=248

Ping statistics for 13.127.124.42:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 22ms, Maximum = 23ms, Average = 22ms

C:\Users\Naren>
```

Fig 5.8 Ping

The following diagram shows the access of Apache server.

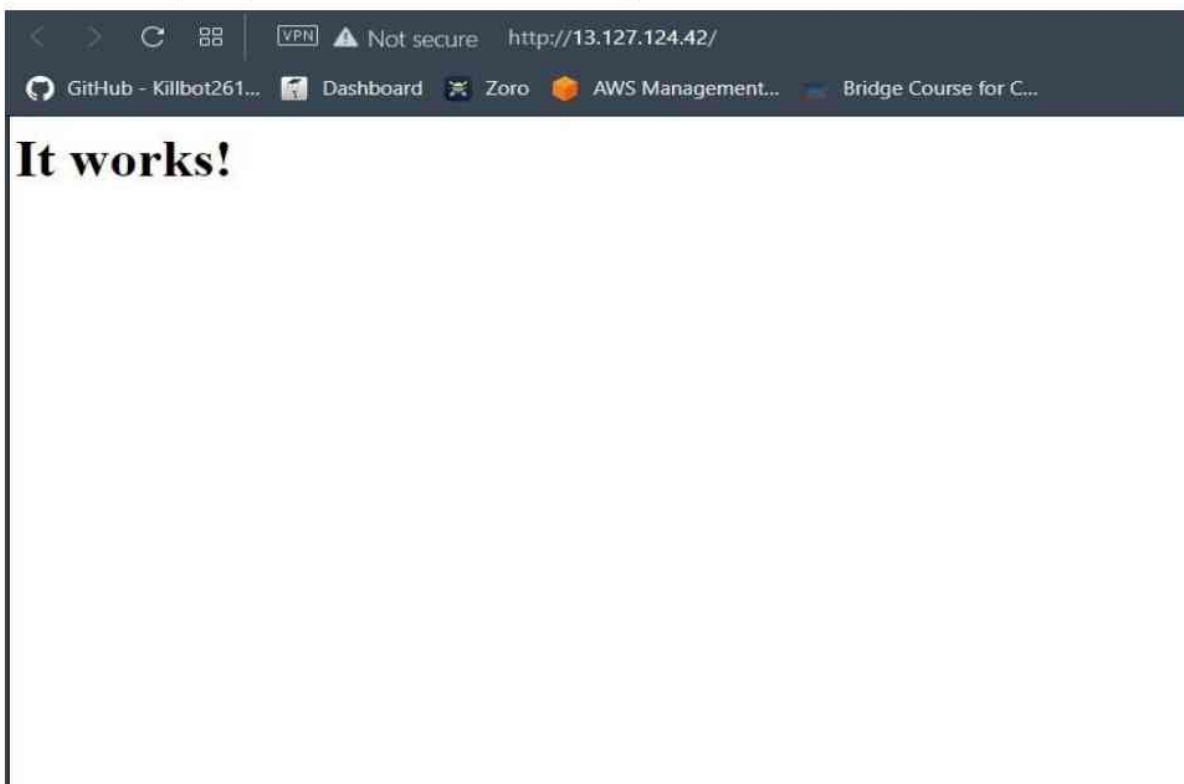
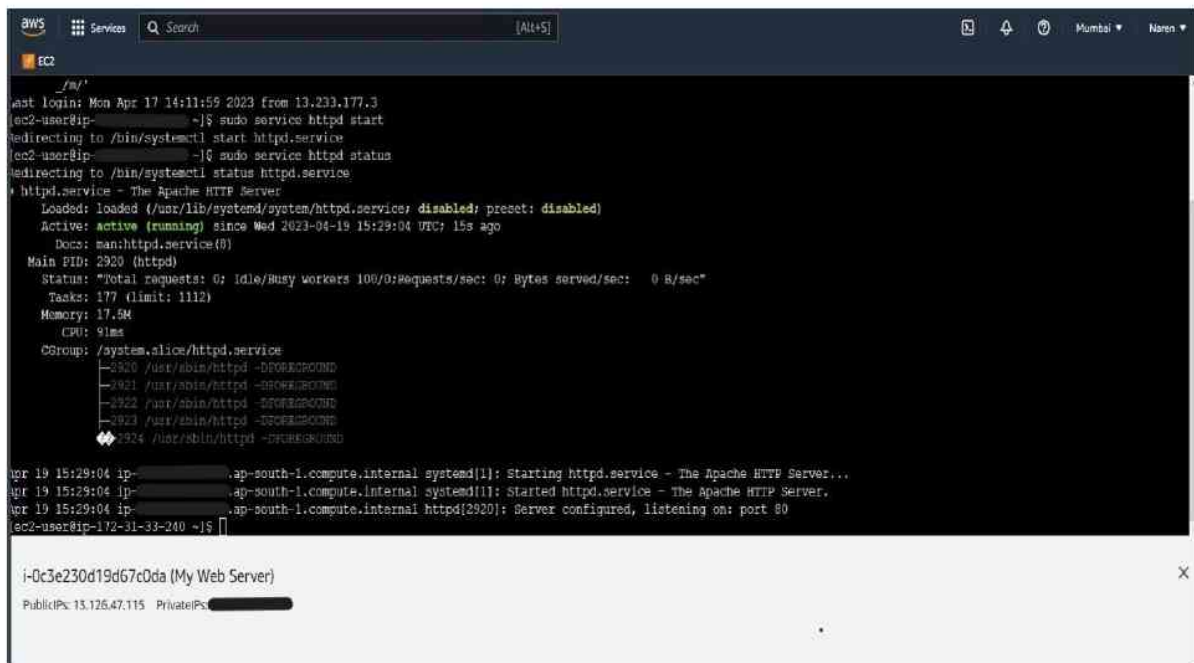


Fig 5.9 Apache Server

The following diagram shows the AWS Management console.



The screenshot displays the AWS Management console interface. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and a '[Alt+S]' shortcut. Below this, the 'EC2' tab is selected. The main area shows a terminal window for an EC2 instance. The terminal output includes the following text:

```
/mi/
last login: Mon Apr 17 14:11:59 2023 from 13.233.177.3
[ec2-user@ip-172-31-33-240 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-172-31-33-240 ~]$ sudo service httpd status
Redirecting to /bin/systemctl status httpd.service
* httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
   Active: active (running) since Wed 2023-04-19 15:29:04 UTC; 15s ago
     Docs: man:httpd.service(8)
   Main PID: 2920 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0 B/sec"
     Tasks: 177 (limit: 1112)
    Memory: 17.5M
       CPU: 91ms
   CGroup: /system.slice/httpd.service
           └─2920 /usr/sbin/httpd -DFOREGROUND
             └─2921 /usr/sbin/httpd -DFOREGROUND
               └─2922 /usr/sbin/httpd -DFOREGROUND
                 └─2923 /usr/sbin/httpd -DFOREGROUND
                   └─2924 /usr/sbin/httpd -DFOREGROUND

Apr 19 15:29:04 ip-172-31-33-240 systemd[1]: Starting httpd.service - The Apache HTTP Server...
Apr 19 15:29:04 ip-172-31-33-240 systemd[1]: Started httpd.service - The Apache HTTP Server.
Apr 19 15:29:04 ip-172-31-33-240 httpd[2920]: Server configured, listening on: port 80
[ec2-user@ip-172-31-33-240 ~]$
```

Below the terminal window, a summary box for the instance 'i-0c3e230d19d67c0da (My Web Server)' is visible, showing its Public IP as 13.126.47.115 and a redacted Private IP.

Fig 5.10 AWS Management console

CHAPTER-5

SUMMARY AND CONCLUSION

5.1 CONCLUSION

In conclusion, the creation of a static port virtual machine (VM) with ping and Apache server access using Amazon EC2 can provide businesses with a flexible and scalable solution for hosting web applications. By leveraging EC2's features, such as efficient resource management, flexible deployment options, easy management and monitoring, and robust security features, businesses can create a reliable and secure environment for hosting their web applications.

5.2 FUTURE WORK

There are several potential areas for future work in the creation of a static port VM with ping and Apache server access. Some possible directions for further exploration include:

- **Customizing Security Groups:** Businesses can further enhance the security of their VM by customizing security groups to allow only necessary inbound and outbound traffic, restrict access based on IP addresses or CIDR blocks, and configure advanced firewall rules. This can help mitigate potential security risks and ensure that the VM is accessible only to authorized users.
- **Automating Deployment:** EC2 provides tools like AWS CloudFormation and AWS Elastic Beanstalk that allow for automated provisioning and management of resources, including VMs. Businesses can explore automating the deployment of static port VMs with ping and Apache server access using these tools, enabling faster and more consistent deployments.
- **Performance Optimization:** Fine-tuning the performance of the VM, such as optimizing the Apache server settings, tuning the VM's CPU and RAM configurations, and leveraging caching mechanisms, can further optimize the performance of the web applications hosted on the VM.

- **Monitoring and Alerting:** Implementing comprehensive monitoring and alerting solutions using AWS CloudWatch or other monitoring tools can provide real-time visibility into the health and performance of the VM, allowing for proactive identification and resolution of issues.
- **Scaling and Load Balancing:** As web applications grow, businesses may need to scale their VMs horizontally or vertically, or implement load balancing to distribute traffic across multiple VMs. Exploring EC2's auto-scaling and load balancing capabilities can help ensure that the web applications hosted on the VM remain highly available and responsive to user traffic.

In summary, the creation of a static port VM with ping and Apache server access using Amazon EC2 opens up opportunities for further customization, automation, performance optimization, monitoring, and scaling, offering businesses potential avenues for future work to enhance their web hosting capabilities.

REFERENCES

1. Karen Scarfone and Paul Hoffman, Guidelines on Firewalls and Firewall Policy, National Institute of Standards and Technology, NIST Special Publication 800-41 Revision 1.
2. Whyman, B.: Cloud Computing, Information Security and Privacy Advisory Board, pp.11–13 (December 5, 2008).
3. J.Kirch, Virtual Machine Security Guidelines Version 1.0. The Center for Internet Security, September 2007.
4. B. R. Kandukuri, R. P. V. and A. Rakshit, "Cloud Security Issues" in *Proceedings of the 2009 IEEE International Conference on Services Computing* , 2009, pp. 517-520.
5. Shanthi Bala, P. “Intensification of educational cloud computing and crisis of data security in publicclouds”, International Journal on Computer Science and Engineering (IJCSE), Vol. 02, No. 03, 2010,741-745.
6. T. Dillon, C. Wu and E. Chang, “Cloud Computing: Issues and Challenges”, 24th IEEE International Conference on Advanced Information Networking and Applications, 2010.
7. <https://blogs.vmware.com/vsphere/2012/05/why-use-static-port-binding-on-vds.html>