

STRENGTHENING IIOT CYBER SECURITY THROUGH SSH AND VPN INTEGRATION

A PROJECT REPORT

Submitted by

**MOHAMED FAISAL.S
CHARUPRAKASH.M.K
RAGUL PRASATH.M
THIRUMALAI RAJ.R**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

THENI KAMMAVAR SANGAM COLLEGE OF TECHNOLOGY

THENI

ANNA UNIVERSITY:: CHENNAI 600 025

MAY 2024

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “**STRENGTHENING IIOT CYBER SECURITY THROUGH SSH AND VPN INTEGRATION**” is the Bonafide work of **MOHAMED FAISAL.S, CHARUPRAKASH.M.K, RAGULPRASATH.M, THIRUMALAI RAJ.R** who carried out the project work under my supervision.

SIGNATURE

Mr. D.Ananth

HEAD OF THE DEPARTMENT

Department of Computer Science and
Engineering,
Theni Kammavar Sangam College
of Technology, Koduvilarpatti,
Theni.

SIGNATURE

Dr.M.Rajashanthi

SUPERVISOR

Assistant Professor
Theni Kammavar Sangam College
of Technology, Koduvilarpatti,
Theni.

Submitted for project viva voce held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

The rapid expansion of the Industrial Internet of Things (IIoT) brings unprecedented opportunities for industrial advancement. However, with these opportunities come heightened cybersecurity challenges, particularly in safeguarding sensitive data transmission, addressing threats, and implementing stringent access controls. This project proposes a comprehensive cybersecurity strategy that integrates Secure Shell (SSH) and Virtual Private Network (VPN) technologies to fortify the security posture of IIoT systems. As IIoT continues to drive industrial innovation, the proposed cybersecurity strategy incorporating SSH and VPN technologies aims to create a resilient defense against evolving threats. This project aligns with industry best practices and regulatory requirements, ensuring the integrity, confidentiality, and availability of sensitive data within IIoT ecosystems. Additionally, it evaluates the performance and effectiveness of the proposed solution in enhancing the security posture of IIoT deployments, offering recommendations for optimizing cybersecurity measures in AWS-based IIoT environments.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iii
	LIST OF FIGURES	vi
	LIST OF ABBREVIATIONS	vii
1.	INTRODUCTION	1
	1.2 PURPOSE OF PROJECT	2
	1.3 MOTIVATION	3
2.	LITERATURE REVIEW	4
3.	SYSTEM ANALYSIS	5
	3.1 PROBLEM STATEMENT	5
	3.2 EXISTING SYSTEM	5
	3.3 PROPOSED SYSTEM	6
4.	REQUIREMENTS SPECIFICATION	7
	4.1 HARDWARE REQUIREMENTS	8
	4.2 SOFTWARE REQUIREMENTS	8
	4.3 TERMINOLOGY EXPLAINTION	9
	4.3.1 Amazon web service features	13

	4.4 AWS Backup	22
5.	MODULES	22
	5.1 OVERVIEW OF IMPLEMENTATION	22
	5.1 Architecture diagram	23
	5.1.1 VPC Configuration Module	24
	5.1.2 VPN Configuration Module	26
	5.1.3 Subnet Configuration Module	29
	5.1.4 Security groups Configuration	32
	5.1.5. Login Module	36
	5.1.6 Monitoring and Log Module	38
	5.1.7 Email automation	42
	5.2 RESULT AND ANALYSIS	43
6.	CONCLUSION	46
	6.1 FUTURE SCOPE	47
	APPENDIX	48
	REFERENCES	49

LIST OF FIGURES

FIGURES NO	LIST OF FIGURES	PAGE NO
4.1	Cloud Services Comparison	13
4.2	VPC Subnets	14
4.3	Site to Site VPN	16
4.4	Network Firewall	18
4.5	AWS Backup	20
5.1	Architecture Diagram	23
5.2	VPC configuration	25
5.3	Routing settings	27
5.4	VPN settings	27
5.5	Client VPN UI	28
5.6	Subnet configuration	30
5.7	Subnet configuration 2	30
5.8	Subnet Configuration 3	31
5.9	SSH outbound rules	33
5.10	SSH inbound rules	33
5.11	SSH outbound rule 2	34
5.12	SSH inbound rule 2	34
5.13	Security Groups	35
5.14	Client Login Module	37
5.15	EC2 Monitoring.	39
5.16	VPN User Logs	41
5.17	VPN User Log 2	41
5.18	Email Automation	42
5.19	Wireshark analysis	43
5.20	SSH server	43

LIST OF ABBREVIATIONS

IIoT	Industrial Internet of Things
SSH	Secure Shell
VPN	Virtual Private Network
AWS	Amazon Web Services
IAM	Identity and Access Management
VPC	Virtual Private Cloud
EC2	Elastic Compute Cloud
S3	Simple Storage Service
KMS	Key Management Service

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The proliferation of IIoT devices has revolutionized industrial operations by enabling real-time monitoring, predictive maintenance, and data-driven decision-making. However, this increased connectivity also exposes critical infrastructure to sophisticated cyber threats, ranging from unauthorized access to data breaches and sabotage. As IIoT devices become integral to industrial processes, ensuring cybersecurity is paramount to protect sensitive data, maintain operational continuity, and safeguard against potential disruptions. Amazon Web Services (AWS) offers a robust cloud infrastructure for hosting IIoT applications, providing scalability, reliability, and accessibility. Nonetheless, securing IIoT deployments on AWS requires a multi-layered approach that addresses vulnerabilities at every level of architecture. Integrating SSH and VPN technologies within the AWS environment presents a holistic solution to bolster IIoT cybersecurity. Secure Shell (SSH) serves as a secure protocol for establishing encrypted connections and authenticating users, facilitating secure remote access to IIoT devices deployed on AWS. By implementing SSH, organizations can enforce stringent access controls, mitigate the risk of unauthorized access, and monitor and audit all remote sessions, thereby enhancing the overall security posture of IIoT deployments.

1.2 PURPOSE OF THE PROJECT

The rapid expansion of the Industrial Internet of Things (IIoT) has led to an increased need for robust cybersecurity solutions to protect sensitive data transmission and prevent cyber threats. This project focuses on securing IIoT devices by implementing Secure Shell (SSH) and Virtual Private Network (VPN) technologies on Amazon Web Services (AWS). The proposed system will strengthen the cybersecurity posture of IIoT infrastructure, ensuring secure communication and access control.

The purpose of the project can be broken down into the following key objectives:

Improving Data Security:

The integration of SSH and VPN will provide robust encryption and secure communication channels for data transmitted between IIoT devices and the network, reducing the risk of unauthorized access and potential data breaches.

Enhancing System Reliability:

By implementing these security measures, the project aims to maintain the integrity and availability of IIoT devices and networks, minimizing the risk of downtime and service disruptions that could negatively impact businesses and industries.

Aligning with Industry Standards:

The project seeks to ensure that the IIoT infrastructure remains compliant with relevant cybersecurity standards and guidelines, demonstrating a commitment to best practices and fostering a culture of security within the organizations utilizing these technologies.

1.3 MOTIVATION

The motivation behind this project stems from the growing concerns surrounding the security vulnerabilities of Industrial Internet of Things (IIoT) systems. With the increasing adoption of IIoT devices and networks across various industries, there is a heightened risk of cyber threats targeting these systems. These threats can lead to data breaches, unauthorized access, and potential disruptions in critical operations.

The primary motivation for this project can be summarized in the following points:

Protecting Critical Infrastructure:

IIoT systems are often integral to the functioning of critical infrastructure in various industries, such as energy, transportation, and manufacturing. Ensuring the security of these systems is crucial to maintaining the smooth operation of these industries and preventing potential disruptions that could impact society.

Data Privacy and Intellectual Property Protection:

IIoT devices collect and transmit vast amounts of data, including sensitive information related to business operations, customer data, and intellectual property. Safeguarding this data is essential to protect the privacy of individuals and organizations, as well as maintaining the competitive edge of businesses in the market.

Regulatory Compliance and Industry Standards:

As IIoT systems become more prevalent, regulatory bodies and industry standards are increasingly emphasizing the importance of cybersecurity. By addressing these concerns through the integration of SSH and VPN, the project aims to ensure that the IIoT infrastructure remains compliant and secure according to the latest guidelines.

CHAPTER 2

LITERATURE REVIEW

"Virtual Private Networks (VPNs) for Industrial IoT Security" (2023) by J. Kim, H. Kim. This paper explores the use of VPNs to create secure tunnels for communication between IIoT devices and cloud platforms. It analyzes how VPNs can encrypt data and protect against unauthorized access in IIoT environments.

"Enhancing IIoT Security with Secure Shell (SSH)" (2022) by R. Gupta, S. Sanyal. This research paper investigates the benefits of using SSH for secure remote access and management of IIoT devices. It discusses how SSH can strengthen authentication, encryption, and overall IIoT security posture.

"A Survey on IIoT Security: Communication Architectures, Key Technologies, and Challenges" (2021) by A. Banerjee, S. Roy. This paper explores the security challenges in IIoT communication architectures, discusses key technologies for securing IIoT systems, and identifies areas for future research.

"Securing Industrial IoT Systems: A Review of Challenges and Solutions" (2020) by M. A. Mahmud, H. Khan. This article reviews the security challenges faced by Industrial IoT (IIoT) systems, analyzes existing security solutions, and emphasizes the importance of secure communication protocols like SSH and VPNs for protecting IIoT data and devices.

CHAPTER 3

SYSTEM ANALYSIS

3.1 PROBLEM STATEMENT

How might we safeguard sensitive data in IIoT(Industrial Internet of Things) systems? Addressing cybersecurity threats, securing data transmission, and implementing robust access controls are critical for ensuring the safe deployment of interconnected industrial solutions. Develop robust cybersecurity measures in safeguarding sensitive data in IIoT (Industrial Internet of Things) systems, ensuring secure data transmission, addressing threats, and implementing stringent access controls.

Output Expected:

Cybersecurity measures to ensure privacy in IoT with proper controls.

3.2 EXISTING SYSTEM

- Access control mechanisms are often limited and may not adequately prevent unauthorized access to IIOT systems.
- Remote access management lacks robust monitoring and regulation capabilities, increasing the risk of unauthorized entry.
- Vulnerabilities in the existing system make IIOT environments susceptible to malware infections and data breaches.
- Limited scalability and flexibility in adapting to evolving cybersecurity threats.
- Lack of comprehensive logging and auditing mechanisms hampers the ability to detect and respond to security incidents effectively.

DISADVANTAGE OF EXISTING SYSTEM

- Limited Access Control: Insufficient controls.
- Weak Remote Access Management: Poor oversight.
- Scalability Challenges: Limited scalability.
- Inadequate Logging and Monitoring: Poor logging.

3.3 PROPOSED SYSTEM

The proposed system for strengthening IIoT cyber security through SSH and VPN integration incorporates several key elements to ensure robust protection against cyber threats. Firstly, we use of public key authentication in SSH authentication replaces traditional password-based login methods, enhancing security by requiring cryptographic keys for authentication so the person with key only access the SSH server. Additionally, the integration of a VPN network, specifically utilizing OpenVPN, establishes a secure tunnel for communication between IIoT devices and the server. This VPN setup restricts SSH traffic to only originate from devices within the authorized VPN network, effectively isolating the server from unauthorized access attempts. If a person tries to login outside of our VPN network the login request will be rejected automatically.

Moreover, the implementation of OpenVPN ensures that all traffic from IIoT devices is encrypted using the OpenVPN protocol, safeguarding data integrity and confidentiality. All the traffic between IIoT device to the server are encrypted only the end user have key to decrypt the data So MIT(Man In The Middle) attack is not possible. To further fortify security measures, the system incorporates mechanisms to prevent common cyber-attacks such as SQL injection, man-in-the-middle attacks, brute force attacks, DoS, DDoS, and keylogging. This comprehensive defence strategy encompasses the configuration of security groups and firewall rules to filter out unwanted traffic and restrict server access to authorized users exclusively. System denies the access to user apart from the security group rules.

Furthermore, the utilization of AWS IAM (Identity and Access Management) enables precise control over user privileges, ensuring that only privileged users can access the server resources. The system also implements two databases with disaster recovery capabilities, guaranteeing data integrity and availability in the event of system failures or disasters. Physical security is enhanced by hosting the server in a data center located in Ireland, which adds an additional layer of protection against unauthorized access attempts.

Detailed logging and monitoring mechanisms are employed to track user activity and monitor VPN connections, including logging the IP addresses of connected devices. Additionally, inbound and outbound rules are configured to enforce strict access control, allowing only authorized users to establish connections to the server. By incorporating these measures, the proposed system provides a robust defense against cyber threats, ensuring the security and reliability of IIoT operations.

Finally, if all the above measures are failed and someone gained access to the ssh server. It triggers an event. A python script will be executed that will send a alert message to the admin and gives a heads-up to stop the unauthorized user.

ADVANTAGE OF PROPOSED SYSTEM

- Enhanced Encryption: Robust encryption.
- Improved Access Control: Strict controls.
- Better Remote Access Management: Advanced management.
- Scalability and Flexibility: Flexible scalability.
- Comprehensive Monitoring: Continuous monitoring.
- Automated Incident Response: Rapid response

CHAPTER 4

REQUIREMENTS SPECIFICATION

4.1 HARDWARE SPECIFICATION:

Workstations : Intel Core i5 Processor, 8 GB RAM, and 256 GB SSD.

Network Devices : Routers, and Wireless Access Points.

4.2 SOFTWARE SPECIFICATION:

Operating System : Windows 10 or Linux (Ubuntu).

Development Tools : AWS Management Console, AWS SDKs

Programming Language: Python.

AWS Services : Amazon EC2 Instances, Amazon VPC.

Networking Protocols : SSH,VPN, OpenVPN, and TCP

4.3 TERMINOLOGY EXPLANATION

Routers:

Routers are networking devices that forward data packets between computer networks. They operate at the network layer of the OSI model and use routing tables to determine the best path for forwarding packets. Routers are crucial components in connecting different networks together and directing traffic efficiently.

Wireless Access Points (WAP):

Wireless Access Points are devices that allow wireless devices to connect to a wired network using Wi-Fi. They typically connect to a router and broadcast a wireless signal, enabling devices such as laptops, smartphones, and tablets to access the network and the internet wirelessly.

Amazon EC2 Instances:

Amazon Elastic Compute Cloud (EC2) is a web service that provides resizable compute capacity in the cloud. EC2 instances are virtual servers hosted on the AWS cloud platform. Users can launch instances with various configurations, such as CPU, memory, storage, and networking, to run applications and workloads.

Amazon VPC (Virtual Private Cloud):

Amazon Virtual Private Cloud (VPC) is a virtual network environment within the AWS cloud where users can launch AWS resources. It allows users to define a virtual network topology, including subnets, route tables, and network gateways, providing control over the network configuration and isolation from other VPCs.

AWS Management Console:

The AWS Management Console is a web-based interface provided by Amazon Web Services (AWS) for managing various AWS services and resources. Users can access and configure AWS services, monitor usage, and manage security settings through the console.

AWS SDKs (Software Development Kits):

AWS SDKs are programming libraries and tools provided by Amazon Web Services for interacting with AWS services programmatically. SDKs are available for various programming languages, allowing developers to integrate AWS services into their applications and automate tasks such as provisioning resources, managing data, and monitoring performance.

SSH (Secure Shell):

SSH is a cryptographic network protocol used for secure remote access to computer systems. It provides encrypted communication between two devices, typically a client and a server, allowing users to log in to a remote system securely and execute commands remotely.

VPN (Virtual Private Network):

A Virtual Private Network (VPN) is a secure and encrypted connection established over a public network, such as the internet. VPNs are used to create a private network environment over a public infrastructure, enabling secure communication and access to resources from remote locations.

OpenVPN:

OpenVPN is an open-source VPN software that provides secure remote access and site-to-site connectivity. It uses SSL/TLS protocols for encryption and authentication, offering flexibility and robust security features for VPN implementations.

IPsec (Internet Protocol Security):

IPsec is a suite of protocols used for securing internet protocol (IP) communications by authenticating and encrypting data packets. It provides confidentiality, integrity, and authenticity for IP traffic, commonly used in VPN implementations for secure communication between networks.

TCP (Transmission Control Protocol):

TCP is a core protocol of the internet protocol suite used for transmitting data reliably between devices over a network. It ensures that data packets are delivered in the correct order and without errors by establishing a connection-oriented communication channel between sender and receiver.

AWS Key Management Service (KMS):

Offers secure storage and management of encryption keys used for SSH and VPN communication.

AWS IAM (Identity and Access Management):

Enables centralized control over user access and permissions for resources within the AWS environment.

4.3.1 Difference Between SSH and VPN

SSH	VPN
Secure Shell (SSH): SSH is a cryptographic network protocol that enables secure remote login and command-line execution on a server. It utilizes public-key cryptography for user authentication and encrypts all data transmission during communication sessions. This ensures that only authorized users can access IIoT devices, and any intercepted data remains unreadable.	Virtual Private Network (VPN): A VPN creates a secure tunnel over a public network like the internet. All data traffic sent through the tunnel is encrypted, protecting it from unauthorized access. In the context of IIoT, a VPN encrypts communication between all devices within the network, safeguarding sensitive data from eavesdropping or tampering.
While both the Secure Shell (SSH) protocol and a virtual private network (VPN) create encrypted tunnels for connection traffic, they serve different purposes. SSH is typically used to access remote servers, transfer files and execute codes on remote systems. Meanwhile, a VPN creates a secure connection over a network and encrypts your internet traffic.	A VPN works by encrypting and routing your internet traffic through a VPN server before it reaches its destination. VPN clients can typically connect to VPN servers by using the dedicated apps of their VPN service providers. Once the user establishes a connection between their device and a VPN server.
SSH works by establishing an encrypted connection between your device and a remote server using public key authentication. First, your device and a remote server have to generate public and private key pairs. The public keys are shared, whereas each party keeps its private key in secret.	A VPN tunnel is a secure and encrypted connection between a VPN server and your device. A VPN tunnel ensures that the online traffic stemming from your device would reach the internet without leaking any information about you or your activities to third parties.

4.3.2 AMAZON WEB SERVICE FEATURES

- Powerful code editor with smart editing and code re-factoring.
- Emulator to show your code output in various resolutions,
- Gradle based build support.
- Template-based wizards.

Comparison of other cloud services

The other cloud services in the market are compared in below fig 4.1





			
Free trial	12-month free trial	12-month free trial	12-month free trial
SLA	Amazon EC2- 99.5% annual uptime Amazon S3 - A monthly uptime of at least 99.9% for a billing cycle	99.9% uptime	99.95% monthly uptime
Max processors in Virtual Machines	128 nos	128 nos	96 nos
Maximum Memory in Virtual Machines	3904GiB	3800GiB	1433GiB
Supported OS	Core OS, Windows, SLES, Cloud Linux, Ubuntu, etc	SLES, Windows, CentOS, Oracle Linux, etc	Windows, SLES, CoreOS, FreeBSD, etc
Availability Zones	66 availability zones with 12 on the anvil	54 regions worldwide and is available in 140 countries.	in 20 regions around the world with 3 more on the way
Pros	Reliability, Top Professional Support, Quality	Flexible Infrastructure, Configurations, Ideal for large projects	Reliability, Costeffective, Easily affordable
Cons	Expensive despite low prices	Customer Support is longer process and tedious	Limited features, Less services

Fig 4.1 Cloud Services Comparison

Amazon Virtual Private Cloud (VPC):

It is a service provided by Amazon Web Services (AWS) that allows users to create and manage a logically isolated virtual network in the cloud. It provides users with the capability to launch AWS resources, such as Amazon Elastic Compute Cloud (EC2) instances into a virtual network that they define.

In simpler terms, an Amazon VPC acts as an extension of your on-premises network infrastructure, enabling you to connect your existing resources to the AWS cloud. This virtual network offers you complete control over your resources, including the ability to define your own IP address range, configure subnets, and customize network gateways and route tables.



Fig 4.2 VPC Subnets

4.3.3 Some key features and benefits of Amazon VPC include:

Customizable network architecture:

You can design your VPC to match your on-premises network, making it easier to integrate and manage resources across both environments.

Enhanced security:

VPC provides advanced security features, such as security groups and network access control lists (ACLs), to help protect your resources and control traffic flow within your virtual network.

Flexibility:

Amazon VPC allows you to choose from various options for IP addresses, such as assigning them statically or using Dynamic Host Configuration Protocol (DHCP).

Scalability:

As your business grows and your network requirements change, you can easily scale your VPC to accommodate new resources and workloads.

Cost-effective:

By utilizing Amazon VPC, you only pay for the resources you use, which helps you optimize your cloud expenses.

Virtual Private Gateway (VGW)

A Virtual Private Gateway enables secure communication between your VPC and your on-premises network via VPN or Direct Connect connections. It extends the private network into the cloud securely.

Elastic IP Addresses

VPCs support Elastic IP addresses, which are static IP addresses that can be dynamically assigned to instances within the VPC. Elastic IP addresses facilitate seamless failover and enable instances to maintain consistent public-facing IP addresses.

An Amazon Site-to-Site VPN (Virtual Private Network)

Connection allows you to create a secure and encrypted connection between your Amazon Virtual Private Cloud (VPC) and your on-premises network shown in fig 4.3. This enables seamless communication and data transfer between the two environments.



Fig 4.3 Site to Site VPN

This process requires specifying parameters such as IP addresses, authentication methods, and encryption algorithms. Once the VPN connection is established, you need to update the route tables in your VPC to direct traffic destined for your on-premises network through the VPN tunnel. Concurrently, the configuration of your on-premises VPN device is essential to establish connectivity with the VGW. After setup, thorough testing ensures proper traffic flow through the VPN tunnel, validating the connectivity between resources in your VPC and your on-premises network. AWS provides comprehensive documentation and guides to facilitate this process, enabling seamless integration of your on-premises network with AWS resources through a secure and reliable site-to-site VPN connection.

4.3.4 To set up an Amazon Site-to-Site VPN connection, follow these general steps:

Plan your VPN connection:

Determine the requirements for your VPC, such as IP address ranges, subnets, and the type of VPN connection (e.g., IPsec or SSL).

Create a VPC:

If you haven't already, create a VPC in the AWS Management Console.

Configure a VPN Gateway:

In the VPC dashboard, create a VPN Gateway that will act as the endpoint for your VPN connection.

Create Customer Gateways:

Set up a Customer Gateway on the AWS side and another on your on-premises network. The Customer Gateway represents your local network and is used to terminate the VPN connection.

Create a VPN Connection:

In the VPC dashboard, create a VPN Connection between the VPN Gateway and the Customer Gateways. Configure the required settings, such as pre-shared keys or digital certificates.

Configure Routing:

Update the routing tables on both sides to allow communication between the VPC and your on-premises network.

Test the Connection:

Verify that the VPN connection is established and functioning correctly by testing between resources in your VPC and on-premises network.

A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predefined security rules. It acts as a barrier between your internal network and the internet, helping to prevent unauthorized access and protect against potential threats. It was used in fig 4.4

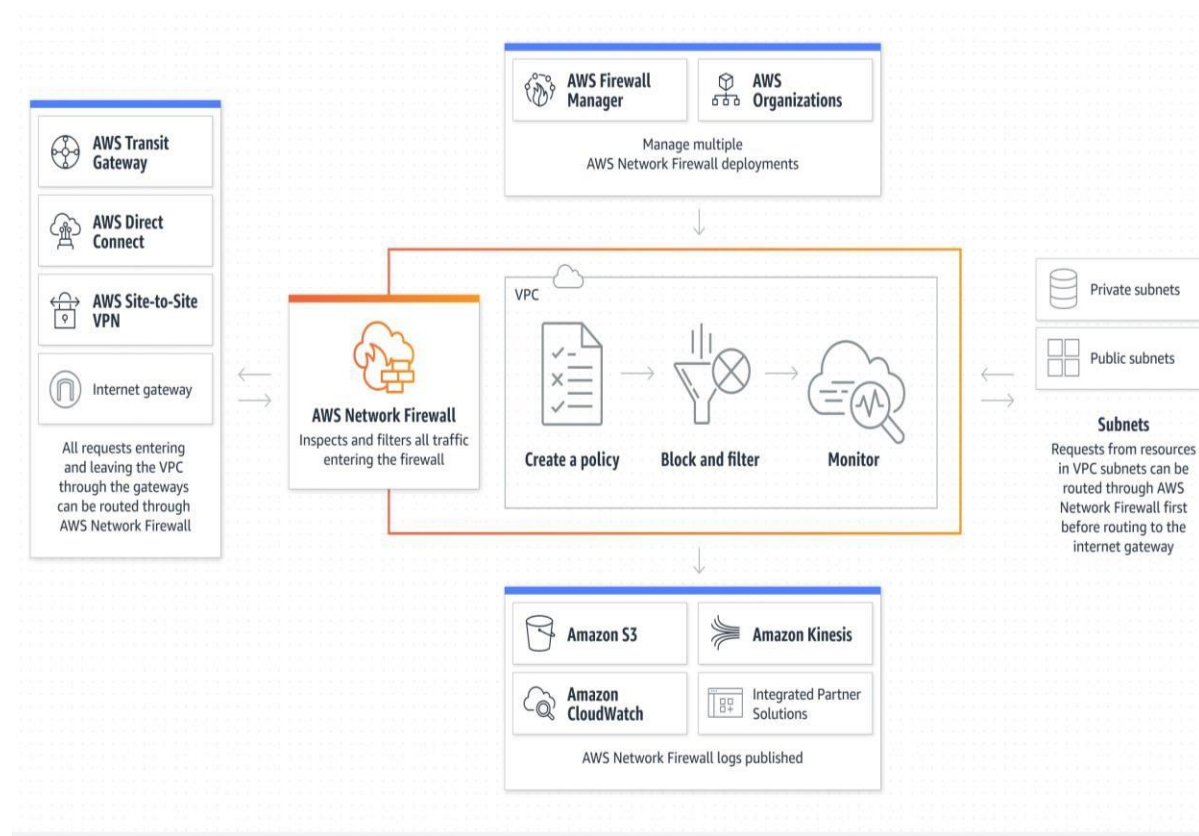


Fig 4.4 Network Firewall

Operating at the application layer (Layer 7) of the OSI model, Network Firewall allows for deep packet inspection and filtering based on application-level protocols like HTTP, HTTPS, and DNS. This granular control enables you to define custom rules specifying criteria such as IP addresses, port numbers, protocols, and domain names, effectively enforcing security policies tailored to your specific requirements. Leveraging stateful packet inspection, Network Firewall tracks the state of active connections to ensure that only legitimate traffic is allowed into your VPC.

4.3.5 Here's a brief overview of how to set up and manage a firewall network:

Determine your firewall requirements:

Identify the goals of your firewall, such as protecting against specific types of attacks, controlling access to certain resources, or enforcing security policies.

Choose a firewall solution:

There are various firewall types and solutions available, including hardware firewalls, software firewalls, and cloud-based firewalls. Select the one that best suits your needs and infrastructure.

Configure the firewall:

Set up the firewall by defining security rules, policies, and access control lists (ACLs). These rules dictate what type of traffic is allowed or blocked, based on factors like source IP address, destination IP address, port numbers, and protocols.

Implement network segmentation:

Divide your network into smaller segments to reduce the impact of potential security breaches and control access between different parts of your network.

Monitor and manage the firewall:

Regularly review and update your firewall rules to ensure they remain effective against evolving threats. Configure alerts and notifications for any suspicious activity and perform periodic security audits to identify potential vulnerabilities.

Keep your firewall updated:

Regularly install security patches, firmware updates, and software upgrades to maintain the latest security features and protect against newly discovered threats.

Educate your users:

Train your employees on best practices for secure internet usage and encourage them to report any suspicious activity or potential security incidents.

4.4 AWS BACKUP:

Amazon Web Services (AWS) offers several options and services for backing up your data and applications. Here's an overview of some key showed in fig 4.5.

AWS backup solutions:

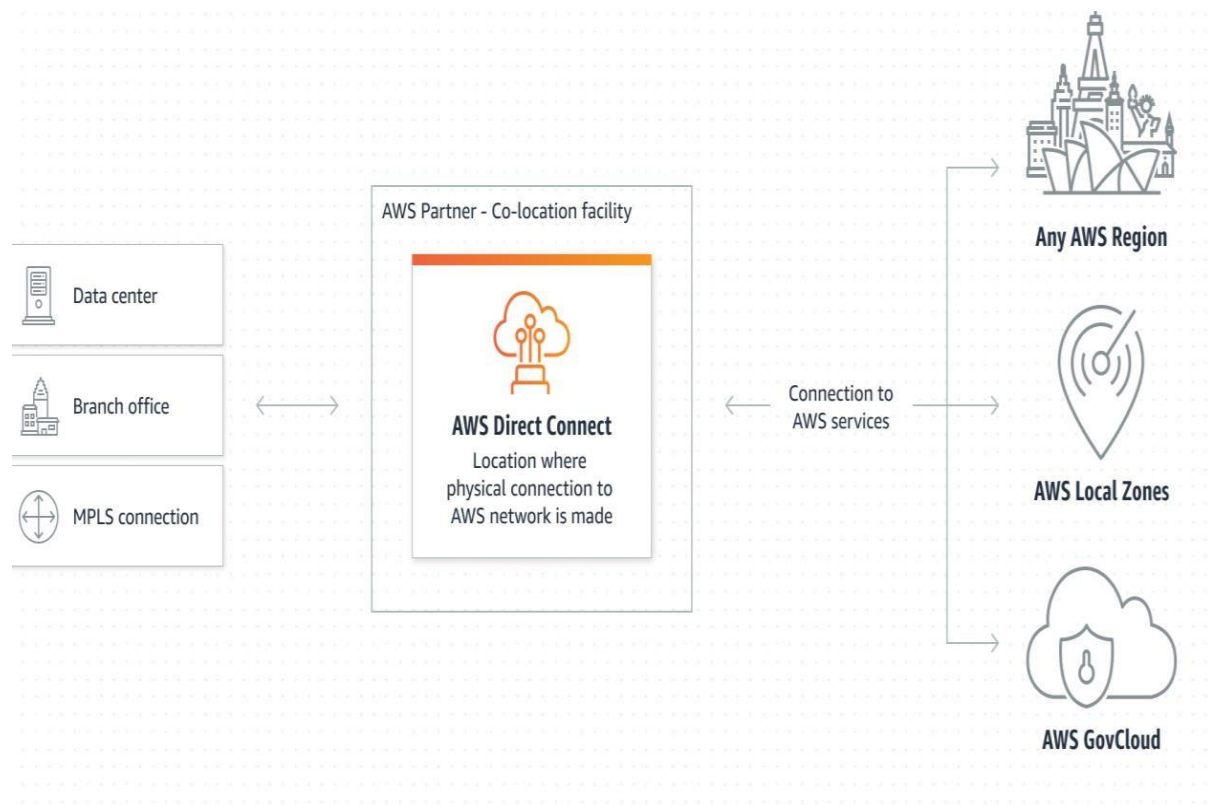


Fig 4.5 AWS Backup

Amazon S3 (Simple Storage Service):

S3 is an object storage service that can be used to store and protect your data. You can create backups by uploading your data to S3 buckets and then configure lifecycle policies, versioning, and other features to manage your backups effectively.

Amazon EBS (Elastic Block Store):

EBS provides block-level storage for your Amazon EC2 instances. You can create backups (snapshots) of your EBS volumes and store them in Amazon S3 or another AWS Region for disaster recovery purposes.

Amazon RDS (Relational Database Service):

RDS allows you to set up and manage relational databases in the cloud. You can create automated backups for your databases, which are stored in the default backup retention period (0-35 days, depending on the database engine). You can also create point-in-time recoveries, database snapshots, and restore your databases from these backups.

This is a centralized backup service that supports multiple AWS services, including RDS, EBS, AmazonFS, and AWS Storage Gateway. AWS Backup simplifies backup management by providing a unified interface for creating, managing, and monitoring backups across these services.

AWS Snowball and Snowball Edge:

These are physical devices used for transferring large amounts of data into and out of the AWS cloud. They can be used for offsite backups and disaster recovery scenarios.

AWS Storage Gateway:

This service enables you to connect on-premises applications to AWS cloud storage. It allows you to use on-premises resources as a part of your backup strategy, providing a hybrid backup solution.

Third-party backup solutions:

Many third-party backup software vendors offer solutions that integrate with AWS services, providing additional features and functionalities for managing and protecting your data

CHAPTER 5

5.1 OVERVIEW OF IMPLEMENTATION

In establishing an AWS cloud infrastructure, the process unfolds through several key stages. Initially, VPC configuration is pivotal, encompassing the determination of VPC requirements such as IP addressing and subnetting. Following this, a VPC is created within the chosen region, accompanied by the setup of an internet gateway (IGW) if internet access is necessary. Subsequently, route tables are configured to govern traffic flow both within the VPC and externally, with optional implementation of network ACLs for subnet-level traffic control. Additionally, provisions are made for VPC peering or VPN connections to facilitate interconnections between VPCs or establish links with on-premises networks. Moving forward, VPN configuration entails decision-making regarding the type of VPN required, be it site-to-site or client-to-site, tailored to specific organizational needs. This is followed by the establishment of a virtual private gateway (VGW) within the VPC, alongside configuration of the customer gateway (CGW) with pertinent details about the remote network or device. Subsequent steps involve the creation of VPN connections, wherein routing options, encryption algorithms, and other parameters are specified in accordance with the VPN requirements. Post-configuration, validation of connectivity ensures seamless interaction between network entities. Furthermore, the process extends to subnet configuration, commencing with the delineation of subnet requirements based on the VPC design. Subnets are then created within the VPC, each associated with an appropriate route table to dictate traffic routing. Optionally, network ACLs are configured at the subnet level to furnish an added layer of security. Meanwhile, security groups configuration involves a meticulous assessment of security requisites for diverse resource types within the VPC. Subsequently, security groups are established and endowed with inbound and outbound rules, aligning with the principle of least privilege to curtail unauthorized access. These groups are then linked with corresponding resources to

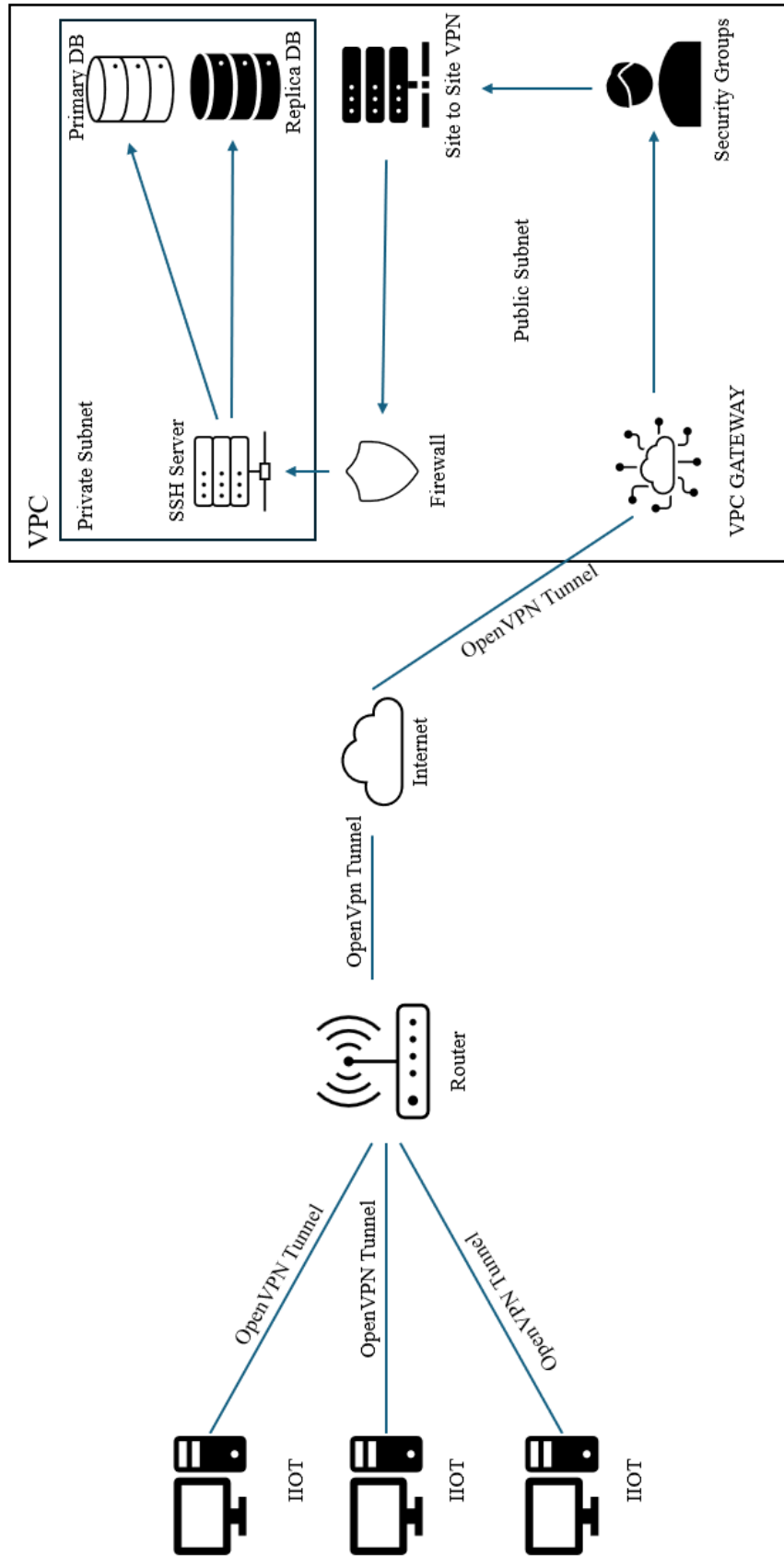


Fig 5.1 Architecture Diagram

enforce the designated security policies. Testing of security group rules follows suit to validate their efficacy. On the user access front, the login module entails the establishment of IAM users, groups, and roles, underpinned by precise delineation of permissions and policies. The implementation of multi-factor authentication (MFA) fortifies security measures, while exhaustive testing ensures seamless user access within the designated parameters. Lastly, the monitoring and log module are instrumental in maintaining operational integrity, involving the activation of AWS CloudWatch for real-time monitoring of AWS resources and applications. CloudWatch alarms are configured to trigger notifications upon the breach of predefined thresholds, complemented by logging setups to capture events and activities for security analysis and compliance auditing. Integration with third-party tools further augments monitoring capabilities, culminating in a robust and secure AWS cloud infrastructure. All these are diagrammatically explained in fig 5.5

MODULES

- 5.1.1 VPC Configuration
- 5.1.2 VPN Configuration
- 5.1.3 Subnet Configuration
- 5.1.4 Security Groups Configuration
- 5.1.5 Login Module
- 5.1.6 Monitoring and Log Module
- 5.1.7 Alert Automation

5.1.1 VPC CONFIGURATION

Step 1: Determine VPC Requirements

Before creating a VPC, it's crucial to understand the requirements. This includes considerations such as IP addressing, subnetting, and availability zones. Determine the size of the VPC, the IP address range (CIDR block), and the number of subnets needed to accommodate different types of resources.

Step 2: Create the VPC

Using the AWS Management Console, AWS CLI, or SDKs, create a VPC in the chosen region with the specified CIDR block. Ensure that the VPC is appropriately named and tagged for easy identification.

Step 3: Configure Internet Gateway (IGW)

If the VPC requires internet access, attach an internet gateway (IGW) to the VPC. This allows instances within the VPC to communicate with the internet and vice versa. Update the route tables to direct internet-bound traffic to the IGW.

Step 4: Set Up Route Tables

Configure route tables to control the routing of traffic within the VPC and to external destinations. Associate the appropriate route tables with the subnets to ensure correct routing.

Step 5: Configure Network ACLs

Optionally, configure network access control lists (ACLs) to control traffic at the subnet level. Network ACLs act as a firewall at the subnet level and allow or deny traffic based on rules defined by the user.

Step 6: Configure VPC Peering or VPN Connections

For interconnecting VPCs or connecting to on-premises networks, configure VPC peering or VPN connections. This enables secure communication between different networks while maintaining isolation and security. Shown in fig 5.2

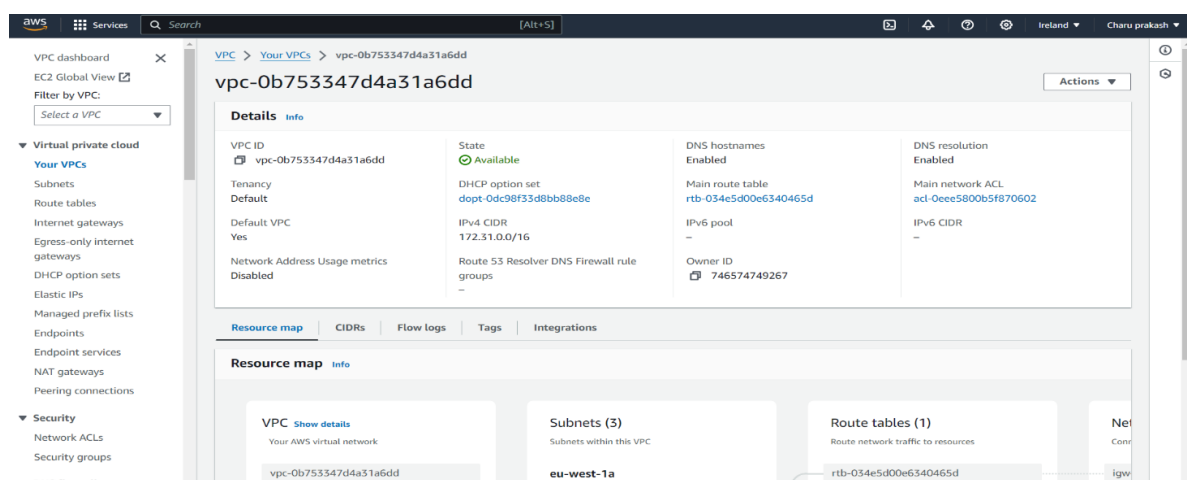


Fig 5.2 VPC configuration

5.1.2 VPN CONFIGURATION MODULE

Step 1: Decide on VPN Type (OpenVpn)

Determine the type of VPN required based on the organization's needs. This could include site-to-site VPN for connecting corporate networks or client-to-site VPN for remote access.

Step 2: Set Up Virtual Private Gateway (VGW)

If not already created, set up a virtual private gateway (VGW) in the VPC. The VGW acts as the endpoint for VPN connections from the customer's network.

Step 3: Configure Customer Gateway (CGW)

Configure the customer gateway (CGW) with the necessary information, such as IP address and authentication details, about the remote network or device.

Step 4: Create VPN Connections

Using the AWS Management Console or CLI, create VPN connections between the VGW and the CGW. Specify the routing options, encryption algorithms, and other parameters according to the VPN requirements.

Step 5: Verify and Establish Connectivity

Once the VPN connections are created, verify connectivity between the networks. Test communication between resources in the VPC and the remote network to ensure the VPN is functioning as expected.

Step 6: Monitoring and Maintenance

Regularly monitor VPN connections for performance and reliability. Implement logging and alerting to detect and troubleshoot any issues that may arise. Perform periodic maintenance to update configurations and ensure security compliance.

A Virtual Private Network (VPN) is a secure, encrypted connection between the IIoT devices and the cloud infrastructure. By setting up a VPN, data transmitted between the devices and the cloud remains private and protected from potential eavesdropping or interception. This module ensures that sensitive information, such as device configurations, operational data, and control commands that are shown in the fig 5.3, are safeguarded during transmission.

VPN Settings

VPN IP Network
Specify the addresses and netmasks for the virtual networks created for VPN clients

Dynamic IP Address Network
When a user does not have a specific VPN IP address configured on the [User Permissions](#) page, the user's VPN client is assigned an address from this network.

Network Address: 172.27.224.0 / # of Netmask bits: 20

Static IP Address Network (Optional)
Any static VPN IP addresses specified for particular users on the [User Permissions](#) page must be within this network

Network Address: / # of Netmask bits: CIDR netmask bits

Group Default IP Address Network (Optional)
When a group does not have a specific Dynamic IP Address pool setting, the dynamic IP address pool for the group will be allocated from this list of subnets.

Network Address: 172.27.240.0/20

Routing
Should VPN clients have access to private subnets (non-public networks on the server side)?

☐ No ☒ Yes, using NAT ☐ Yes, using Routing

Fig 5.3 VPN settings

Routing
Should VPN clients have access to private subnets (non-public networks on the server side)?

☐ No ☒ Yes, using NAT ☐ Yes, using Routing

Specify the private subnets to which all clients should be given access (one per line): 172.31.0.0/16

Should client Internet traffic be routed through the VPN? ☒ Yes

Should clients be allowed to access network services on the VPN gateway IP address? ☒ Yes

DNS Settings
Pushing DNS servers to clients is optional, unless clients' Internet traffic is to be routed through the VPN

Do not alter clients' DNS server settings ☒ Yes

Have clients use the same DNS servers as the Access Server host ☐ No

Have clients use specific DNS servers ☐ No

Fig 5.4 Routing settings

Setting up an OpenVPN server involves installing the OpenVPN software on a designated server and configuring it to establish secure connections for clients. This typically begins with the installation of OpenVPN and the creation of a configuration file specifying parameters like the VPN protocol, port number, and encryption algorithms. Following this, cryptographic certificates and keys are generated for authentication and encryption purposes, including the server certificate, key, and certificate authority (CA) files. Firewall rules must be set up to allow incoming connections on the designated OpenVPN port and, if necessary, network address translation (NAT) rules configured. IP forwarding is then enabled to facilitate the forwarding of VPN traffic between the server and client network interfaces. Once these preparations are complete, the OpenVPN service is started using the configuration file, and server logs are monitored for any potential issues. Finally, client configuration files containing connection settings are distributed to users, allowing them to securely connect to the VPN server for encrypted communication. This comprehensive setup ensures the establishment of a robust and secure OpenVPN server accessible to authorized clients. Like shown in fig 5.5

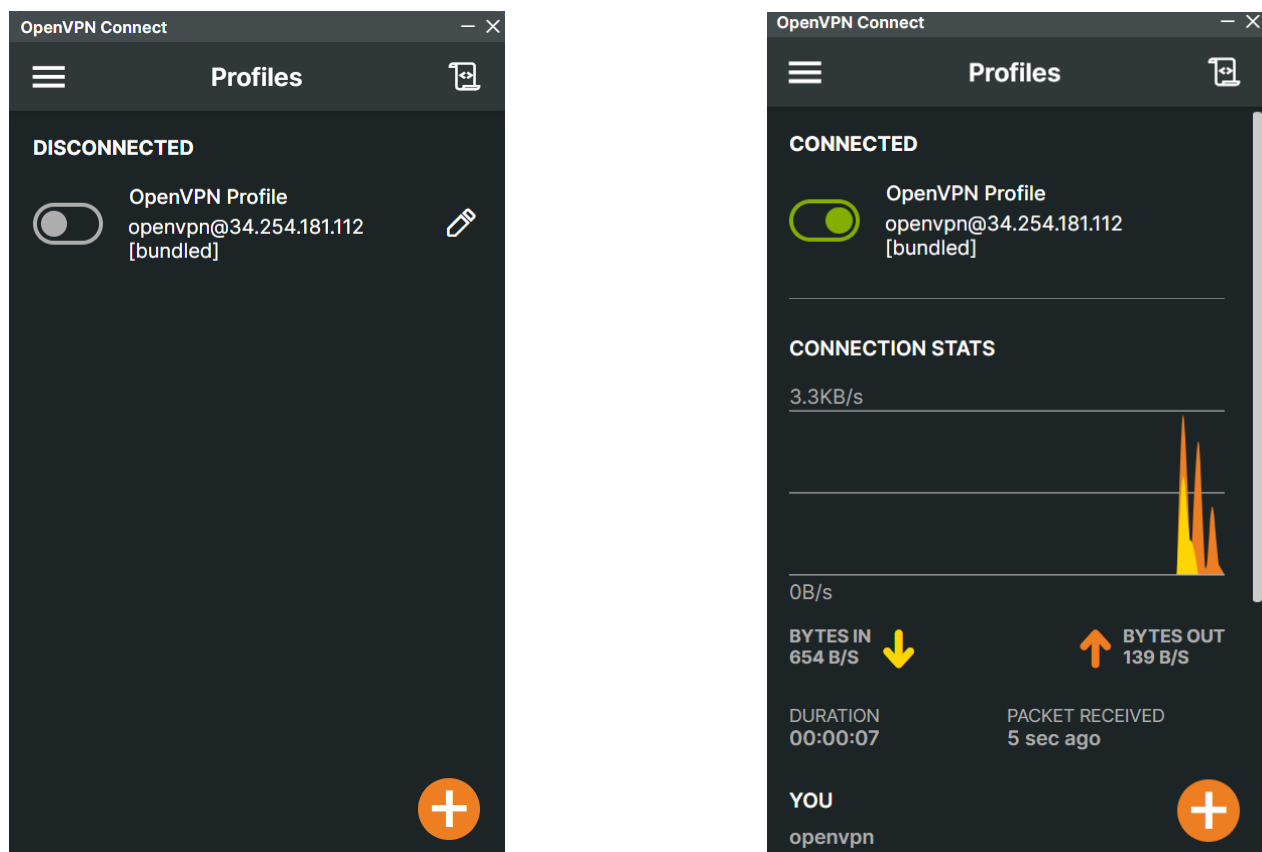


Fig 5.5 Client VPN UI

5.1.3 SUBNET CONFIGURATION MODULE

Step 1: Determine Subnet Requirements

Evaluate the VPC design and requirements to determine the necessary subnets. Consider factors such as public/private access, availability zones, and resource placement.

Step 2: Create Subnets

Using the AWS Management Console or CLI, create subnets within the VPC with appropriate CIDR blocks. Ensure that subnets are appropriately named and tagged for organization and ease of management.

Step 3: Associate Route Tables

Associate each subnet with the appropriate route table to control traffic flow. This ensures that traffic is routed correctly within the VPC and to external destinations.

Step 4: Configure Network ACLs

Optionally, configure network access control lists (ACLs) at the subnet level to provide an additional layer of security. Define rules to allow or deny traffic based on IP addresses, ports, and protocols.

Step 5: Monitoring and Optimization

Regularly monitor subnet usage and performance to identify any potential bottlenecks or issues. Optimize subnet configurations as needed to improve efficiency and resource utilization. Keep track of subnet growth and adjust CIDR blocks accordingly to accommodate future expansion.

In a network, a subnet is a smaller portion of an IP address space. By organizing the network into smaller subnets, this module enhances control and management over the network traffic. In fig 5.6 and 5.7

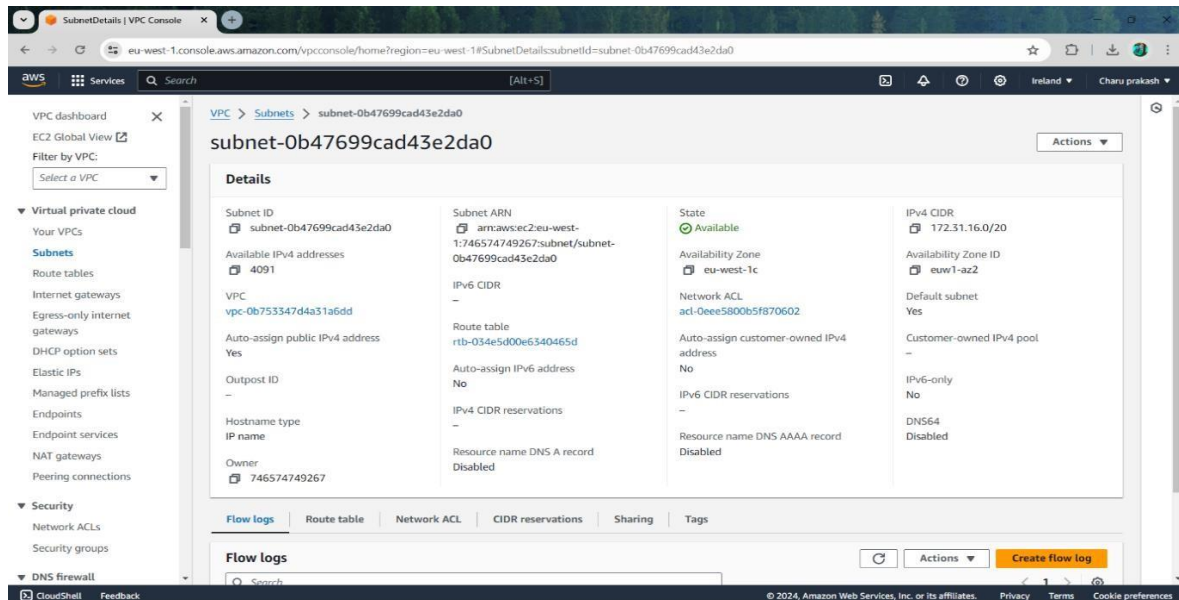


Fig 5.6 Subnet configuration

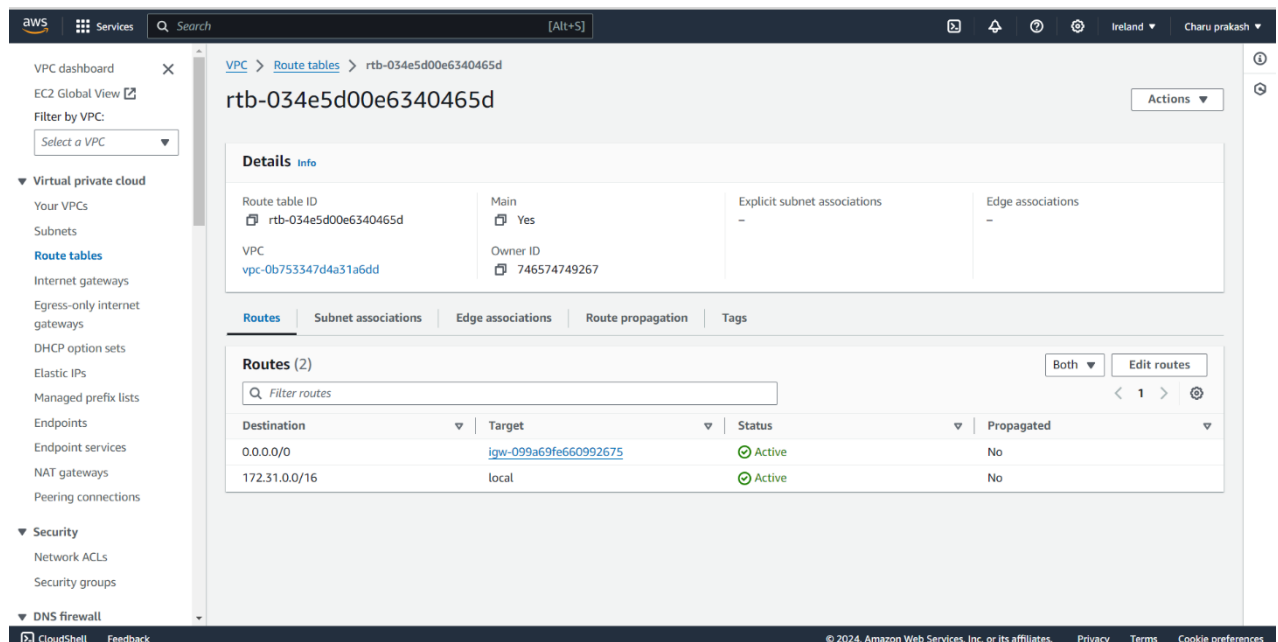


Fig 5.7 Subnet Configuration 2

Subnetting allows for better resource allocation, improved performance, and easier troubleshooting in case of network issues shown in fig5.8, Additionally, it helps in isolating potential security threats, as each subnet can have separate security policies and access controls.

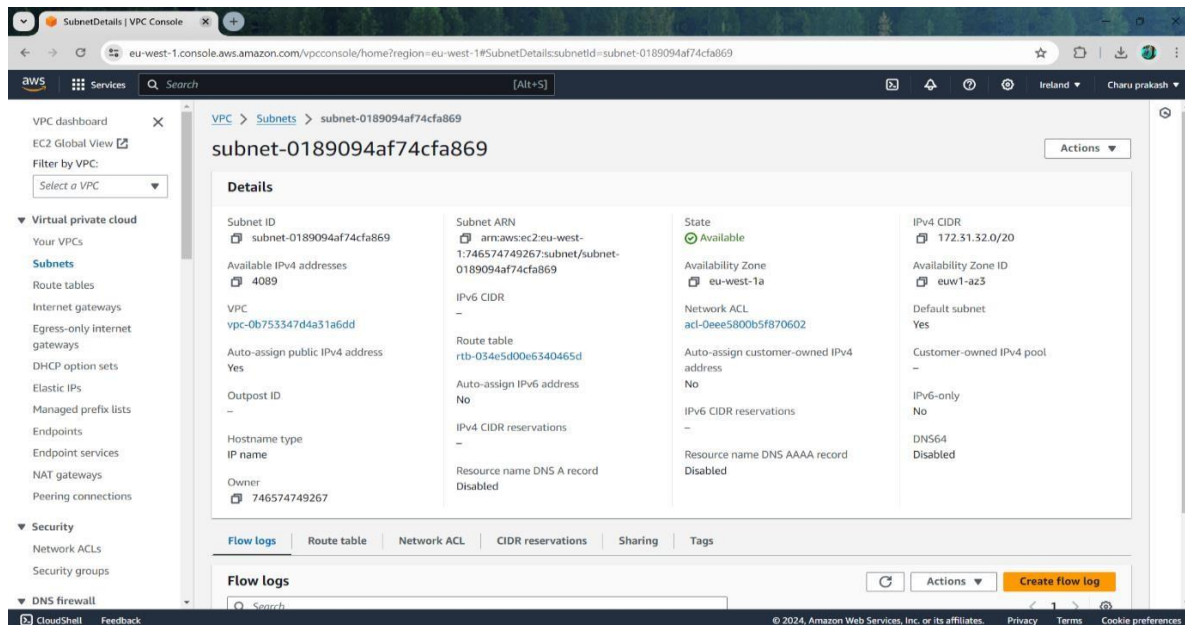


Fig 5.8 Subnet Configuration 3

They are essentially segments of IP address ranges from the VPC CIDR block and are associated with specific availability zones within a region. Subnets enable you to isolate and distribute your resources across different network segments, providing better fault tolerance and scalability. Each subnet can be configured with its own route tables, network access control lists (ACLs), and security groups, allowing you to control traffic flow and security settings at a granular level. By strategically designing and deploying subnets, you can optimize performance, improve security, and build highly available and resilient architectures in AWS.

5.1.4 SECURITY GROUPS CONFIGURATION

Step 1: Identify Security Group Requirements

Begin by identifying the security group requirements for different types of resources within the VPC. Consider factors such as the type of traffic allowed, source/destination IPs, and protocols and ports required.

Step 2: Create Security Groups

Using the AWS Management Console or CLI, create security groups for each resource type. Define inbound and outbound rules to control traffic flow based on the principle of least privilege, allowing only necessary connections.

Step 3: Associate Security Groups

Associate the appropriate security groups with the corresponding resources within the VPC. Ensure that each resource is assigned the necessary security group(s) to enforce the desired security policies.

Step 4: Test Security Group Rules

Thoroughly test the security group rules to ensure they are correctly configured and restrictive enough to prevent unauthorized access. Conduct tests to verify that desired connections are allowed while unauthorized connections are blocked.

Step 5: Monitoring and Compliance

Regularly monitor security group configurations for any changes or anomalies. Implement logging and auditing to track security group changes and ensure compliance with security policies and regulations.

Security groups are a fundamental component of AWS infrastructure, functioning as a virtual firewall that controls inbound and outbound network traffic. In our project, we configure security groups to define and manage access rules for the IIoT devices. These are shown in fig 5.9

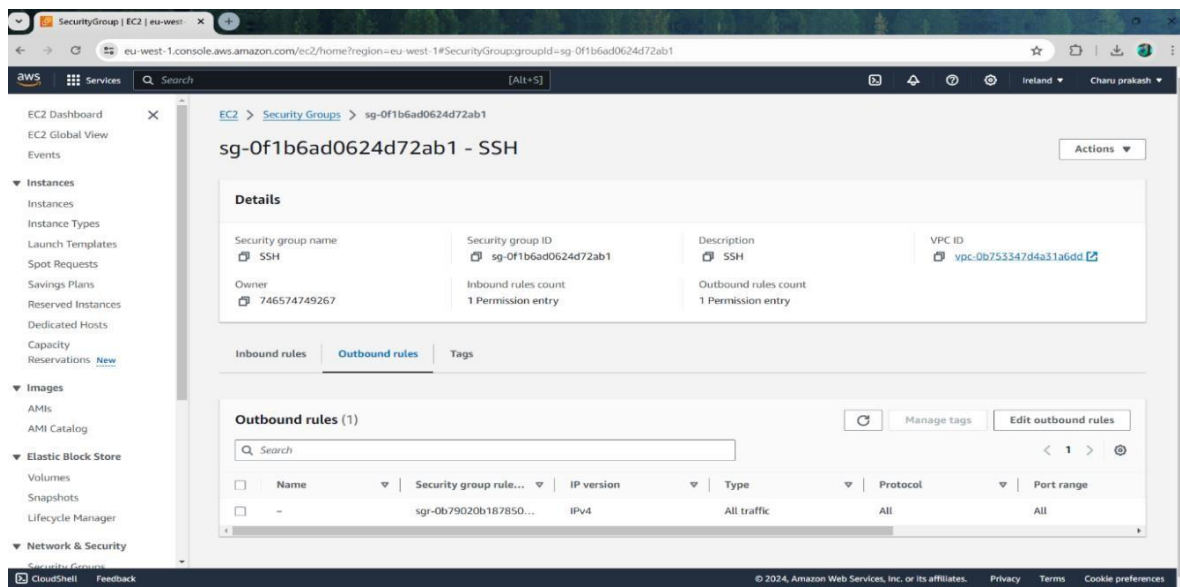


Fig 5.9 SSH outbound rules

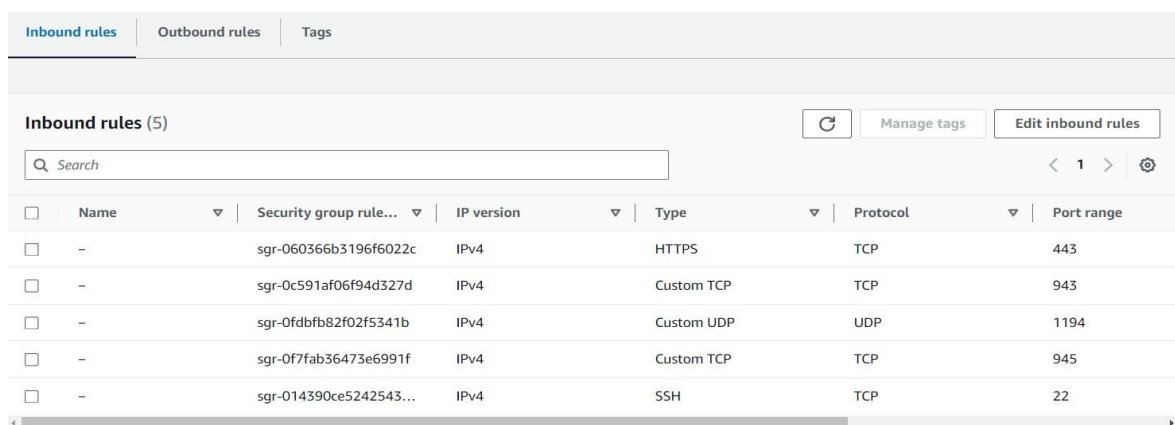


Fig 5.10 SSH inbound rules

Inbound rules Outbound rules Tags							
Outbound rules (1) Manage tags Edit outbound rules							
<input type="text" value="Search"/> < 1 >							
<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	
<input type="checkbox"/>	-	sgr-0a2b1fd24aafd13a7	IPv4	All traffic	All	All	

Fig 5.11 SSH outbound rule 2

By carefully setting up these rules, we can control which traffic is allowed or denied, thus minimizing potential security risks. This module helps in maintaining a secure environment for the IIoT devices and their communication channels. Shown in fig 5.11 and fig 5.12

The screenshot shows the AWS Management Console interface for a security group. The left sidebar contains navigation links for EC2 Dashboard, Instances, Images, Elastic Block Store, and Network & Security. The main content area displays the details for the security group 'sg-0f1b6ad0624d72ab1 - SSH'. The details section includes fields for Security group name, ID, Description, VPC ID, Owner, Inbound rules count, and Outbound rules count. Below the details, there are tabs for Inbound rules, Outbound rules, and Tags. The Inbound rules tab is active, showing a table with one rule.

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-016af46315cc0aea9	IPv4	SSH	TCP	22

Fig 5.12 SSH inbound rule 2

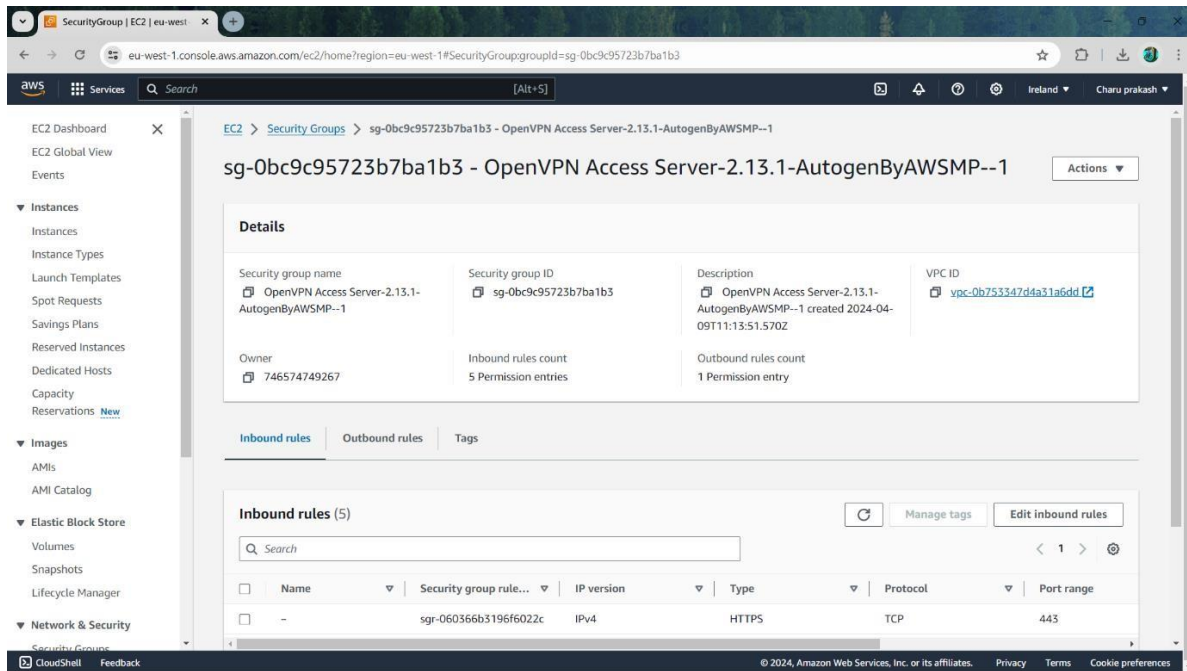


Fig 5.13 Security Groups

These AWS, security groups act like virtual gates around your cloud resources, controlling who can access them and how. Think of them as customizable sets of rules that define what types of traffic are allowed in and out of your Amazon EC2 instances or other resources within your Virtual Private Cloud (VPC). You can specify which IP addresses, ports, and protocols are permitted, effectively creating a barrier against unwanted visitors or potential threats. These security groups shown in fig 5.13 are flexible and easy to manage, allowing you to adjust settings in real-time without disrupting your operations.

By using security groups, you can ensure that your AWS environment remains secure and protected from unauthorized access, while still allowing legitimate traffic to flow smoothly.

5.1.5 LOGIN MODULE

Step 1: Set Up IAM Users, Groups, and Roles

Begin by setting up IAM (Identity and Access Management) users, groups, and roles based on the organization's access policies and requirements. Define the appropriate permissions and policies for each user, group, and role.

Step 2: Define Permissions and Policies

Define IAM policies to control access to AWS resources. Assign permissions to users, groups, and roles based on the principle of least privilege, granting only the permissions necessary for their respective roles.

Step 3: Enable Multi-Factor Authentication (MFA)

Enhance security by enabling multi-factor authentication (MFA) for IAM users. Require users to provide an additional authentication factor, such as a one-time password generated by a hardware or software token, in addition to their username and password.

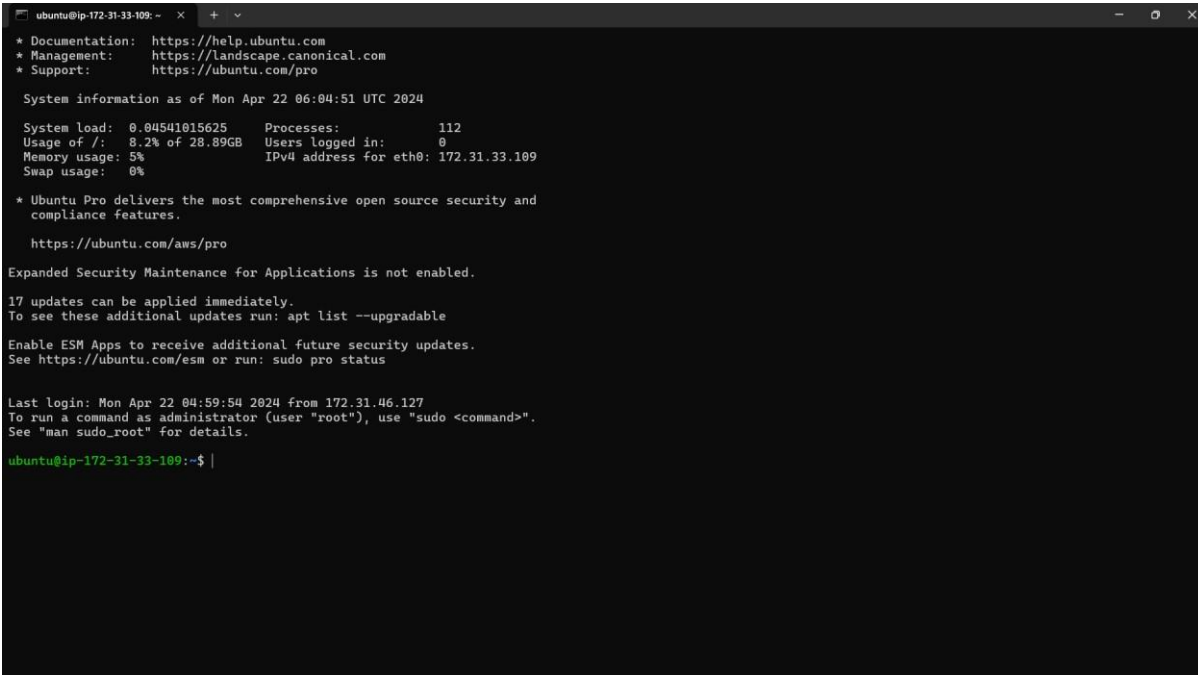
Step 4: Test User Access

Test user access to AWS resources to ensure that IAM permissions are correctly configured. Verify that users can perform their intended actions while being restricted from unauthorized actions.

Step 5: Monitoring and Compliance

Regularly review IAM policies and user permissions to ensure they align with the organization's security policies and compliance requirements. Monitor IAM user activity and access patterns for any suspicious or unauthorized behavior.

A secure authentication and authorization mechanism is vital for ensuring that only authorized users can access the IIoT devices and network resources. In our project, we implement a login module that verifies user identities and grants access based on predefined permissions fig 5.14 shows the client view of login. This module typically involves usernames, passwords, or multi-factor authentication methods to prevent unauthorized access and maintain the integrity of the system.



```
ubuntu@ip-172-31-33-109:~$  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/pro  
  
System information as of Mon Apr 22 06:04:51 UTC 2024  
  
System load:  0.04541015625   Processes:            112  
Usage of /:   8.2% of 28.89GB   Users logged in:      0  
Memory usage: 5%              IPv4 address for eth0: 172.31.33.109  
Swap usage:   0%  
  
* Ubuntu Pro delivers the most comprehensive open source security and  
  compliance features.  
  https://ubuntu.com/aws/pro  
  
Expanded Security Maintenance for Applications is not enabled.  
17 updates can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
Last login: Mon Apr 22 04:59:54 2024 from 172.31.46.127  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-33-109:~$ |
```

Fig 5.14 Client Login Module

In this process of connecting to an Amazon EC2 instance using Windows Terminal and a public-private key pair operates on the principles of asymmetric cryptography and the Secure Shell (SSH) protocol. When initiating an SSH connection, the local machine presents its private key to the remote EC2 instance, which in turn challenges the SSH client to prove its identity by encrypting a random message with the associated public key. The SSH client then decrypts the message using the private key and returns it to the EC2 instance for verification. If the decrypted message matches the original challenge, authentication is successful, granting access to the EC2 instance.

5.1.6 MONITORING AND LOG MODULE:

Step 1: Enable AWS CloudWatch

Enable AWS CloudWatch to monitor AWS resources and applications in real-time. Set up CloudWatch alarms to automatically notify administrators of any abnormal behavior or performance issues.

Step 2: Set Up CloudWatch Alarms

Configure CloudWatch alarms to monitor specific metrics and trigger notifications or automated actions when thresholds are exceeded. Define appropriate thresholds based on performance objectives and service level agreements (SLAs).

Step 3: Configure Logging

Configure logging for AWS services to capture events, API calls, and other activities. Centralize logs in Amazon CloudWatch Logs or export them to Amazon S3 for long-term storage and analysis.

Step 4: Implement AWS CloudTrail

Enable AWS CloudTrail to provide a record of API calls made on the AWS platform. Monitor CloudTrail logs for security analysis, compliance auditing, and troubleshooting purposes.

Step 5: Integration with Third-party Tools

Integrate AWS monitoring and logging data with third-party monitoring tools for comprehensive visibility and analysis. Utilize tools such as Datadog, Splunk, or ELK stack for advanced monitoring, alerting, and log management capabilities.

This module is crucial for maintaining the security and stability of the IIoT system. It continuously monitors the system for any suspicious activities, such as unauthorized access attempts, unusual traffic patterns, or abnormal behaviour. That are shown in the fig 5.15

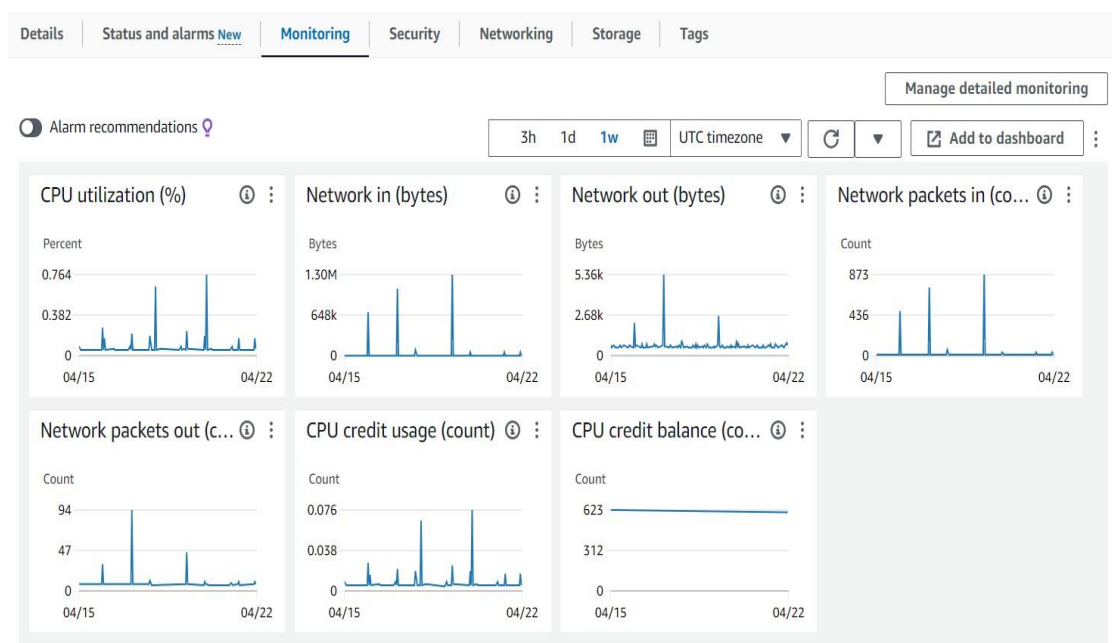


Fig 5.15 EC2 Monitoring

CPU Utilization:

Monitoring the CPU utilization of EC2 instances helps administrators understand the workload on their virtual machines. CloudWatch provides real-time data on CPU utilization, allowing administrators to identify instances that are under heavy load or may require optimization.

Memory Usage:

Tracking memory usage helps administrators ensure that EC2 instances have sufficient memory to handle the applications and services running on them. CloudWatch provides metrics for memory utilization, enabling administrators to optimize instance configurations and allocate resources efficiently.

Disk I/O:

Monitoring disk I/O metrics such as read and write operations per second (IOPS) helps administrators assess storage performance and identify potential bottlenecks.

Network Traffic:

Monitoring network traffic metrics such as incoming and outgoing data transfer rates helps administrators understand the bandwidth requirements of their EC2 instances. CloudWatch provides data on network traffic, enabling administrators to optimize network configurations and allocate bandwidth effectively.

Status Checks:

EC2 instances undergo status checks to ensure that they are running properly and accessible over the network. CloudWatch monitors status checks and provides alerts to administrators if instances fail these checks, enabling proactive troubleshooting and remediation.

Custom Metrics:

Administrators can also monitor custom metrics specific to their applications and workloads using CloudWatch. By defining custom metrics and collecting data from EC2 instances, administrators gain insights into application performance and behavior, allowing them to make informed decisions and optimize resource usage.

Logs:

In addition to metrics, administrators can monitor logs generated by EC2 instances using services like Amazon CloudWatch Logs. By aggregating and analyzing log data, administrators can gain visibility into application behavior, troubleshoot issues, and detect security incidents.

Here we can see CPU utilization (EC2 instance) when we started the instance how much data was passed in and out of the instance from that date , All the details are available here to monitor and scale the instance according to our convenience. This is one of the advantages for using amazon aws there was no charge for this service.

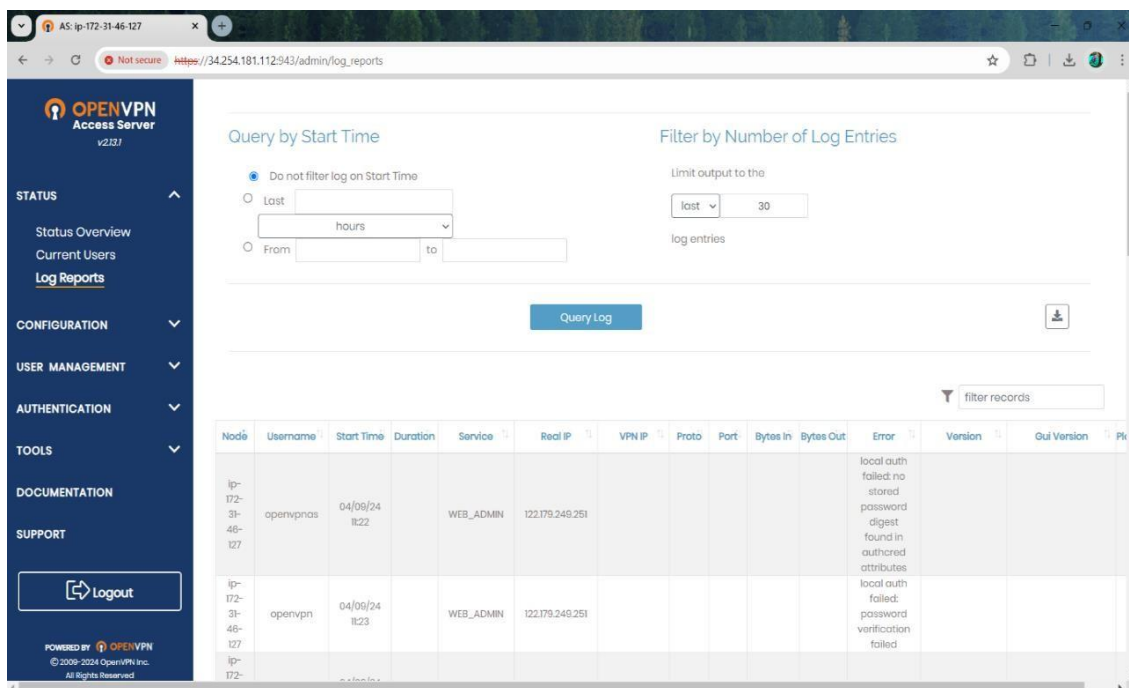


Fig 5.16 VPN User Logs

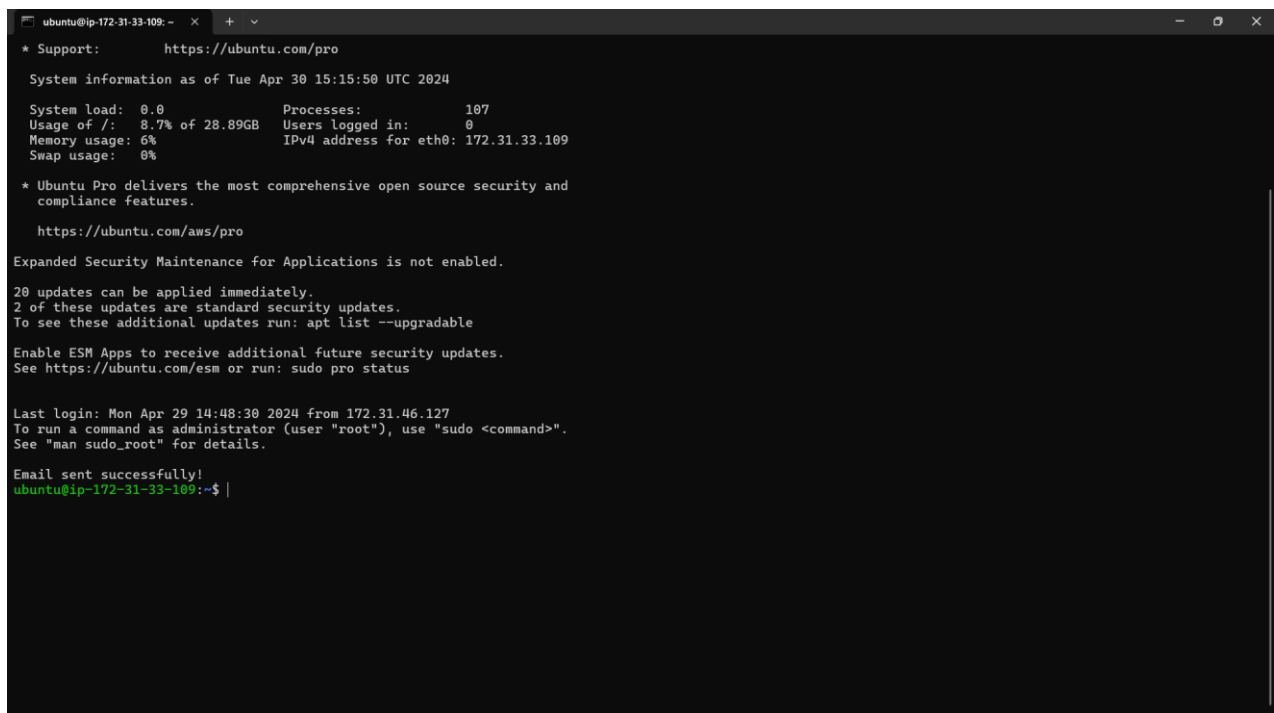
By setting up alerts and notifications, the administrators can promptly respond to potential threats and take necessary actions. Additionally, the log module maintains a record of all activities, allowing for auditing, analysis, and forensic investigations in case of security incidents in fig 5.16 and fig 5.17.

Node	Username	Start Time	Duration	Service	Real IP	VPN IP	Proto	Port	Bytes In	Bytes Out	Error	Version	Gui Version	PK
ip-172-31-46-127	openvpnas	04/09/24 18:22		WEB_ADMIN	122.179.249.251						local auth failed: no stored password digest found in authcred attributes			
ip-172-31-46-127	openvpn	04/09/24 18:23		WEB_ADMIN	122.179.249.251						local auth failed: password verification failed			
ip-172-31-46-127	openvpn	04/09/24 18:23		WEB_ADMIN	122.179.249.251									
ip-172-31-46-127	openvpn	04/09/24 18:27		WEB_CLIENT	122.179.249.251									
ip-172-31-46-127	openvpn	04/09/24 18:35	0d 00:03	VPN	122.179.249.251	172.27.238.2	UDP	1194	937.99 KB	6.64 MB		3.8.2connect3	OCWindows_3.43-3337	
ip-172-31-46-127	openvpn	04/09/24 18:05	0d 00:06	VPN	122.179.249.251	172.27.238.3	UDP	1194	773.61 KB	3.48 MB		3.8.2connect3	OCWindows_3.43-3337	
ip-172-31-46-127	openvpn	04/09/24 18:16	0d 00:05	VPN	122.179.249.251	172.27.232.2	UDP	1194	688.80 KB	1.59 MB		3.8.2connect3	OCWindows_3.43-3337	

Fig 5.17 VPN User Log 2

5.1.7 ALERT AUTOMATION

The Python script for email automation is intricately designed to seamlessly integrate with the SSH server's login process, ensuring that every login event triggers an immediate alert to the system administrator. Upon detection of a login event, the script swiftly generates a comprehensive alert message, containing essential details such as the timestamp, user account, and IP address of the device initiating the login. Leveraging the SMTP functionality in Python, this alert message is promptly dispatched via email to the designated administrator address. To ensure reliability and stability, the script incorporates robust error handling mechanisms, gracefully managing network errors or SMTP server issues. Furthermore, security considerations are paramount, with sensitive information securely stored and stringent access controls enforced. The script's efficiency and scalability allow it to handle high volumes of login events without impacting system performance. It operates autonomously as a background process or service, continuously monitoring SSH login activities across diverse operating systems and server configurations. Through this automation is seen in fig 5.18, administrators can maintain heightened vigilance over server access, bolstering the overall security posture of the system.



```
ubuntu@ip-172-31-33-109: ~  
* Support:      https://ubuntu.com/pro  
System information as of Tue Apr 30 15:15:50 UTC 2024  
System load:  0.0      Processes:           107  
Usage of /:   8.7% of 28.89GB  Users logged in:    0  
Memory usage: 6%      IPv4 address for eth0: 172.31.33.109  
Swap usage:   0%  
* Ubuntu Pro delivers the most comprehensive open source security and compliance features.  
https://ubuntu.com/aws/pro  
Expanded Security Maintenance for Applications is not enabled.  
20 updates can be applied immediately.  
2 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
Last login: Mon Apr 29 14:48:30 2024 from 172.31.46.127  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
Email sent successfully!  
ubuntu@ip-172-31-33-109:~$
```

Fig 5.18 Email Automation

5.2 RESULT AND ANALYSIS

Here we successfully encrypted all the network traffic shown in fig 5.19, So eavesdropping becomes useless, and all the network traffic fortified to industrial security standards and ssh server ensures the security of the storage stored IIot data shown in the fig 5.20 .The data from IIot to ssh server is secured using vpn and stored data is secured using ssh protocol.

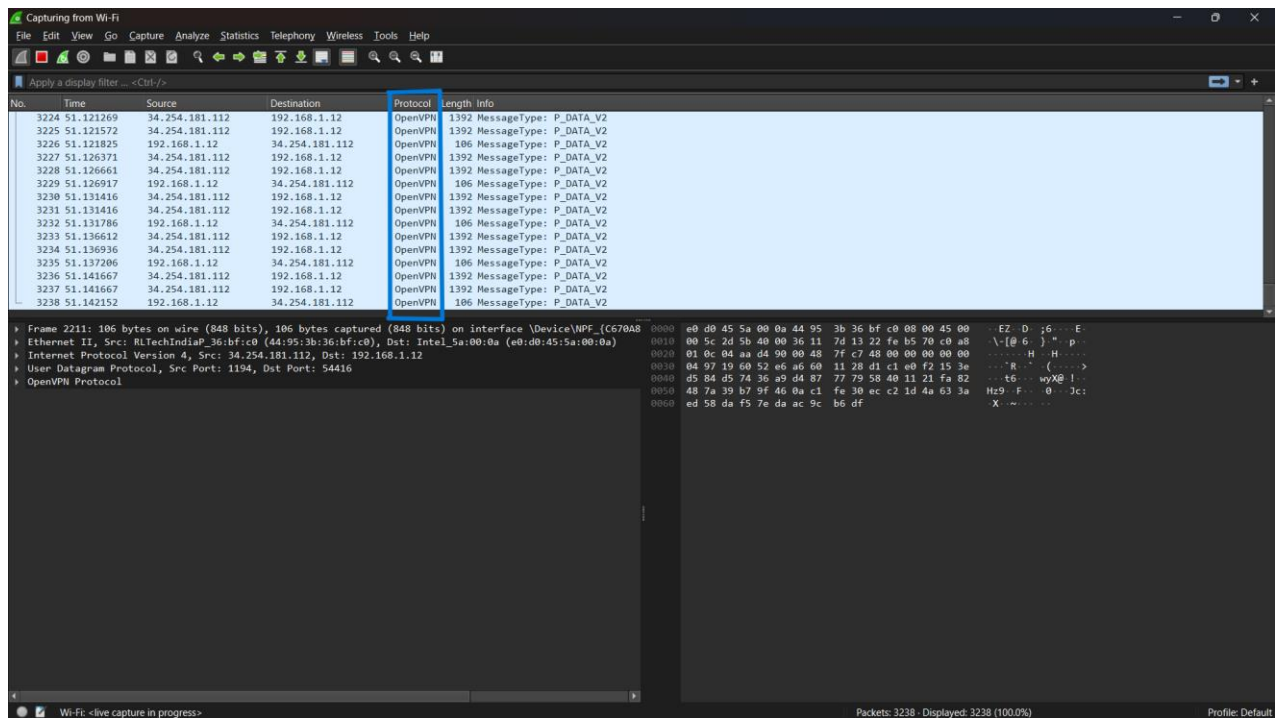


Fig 5.19 Wireshark analysis

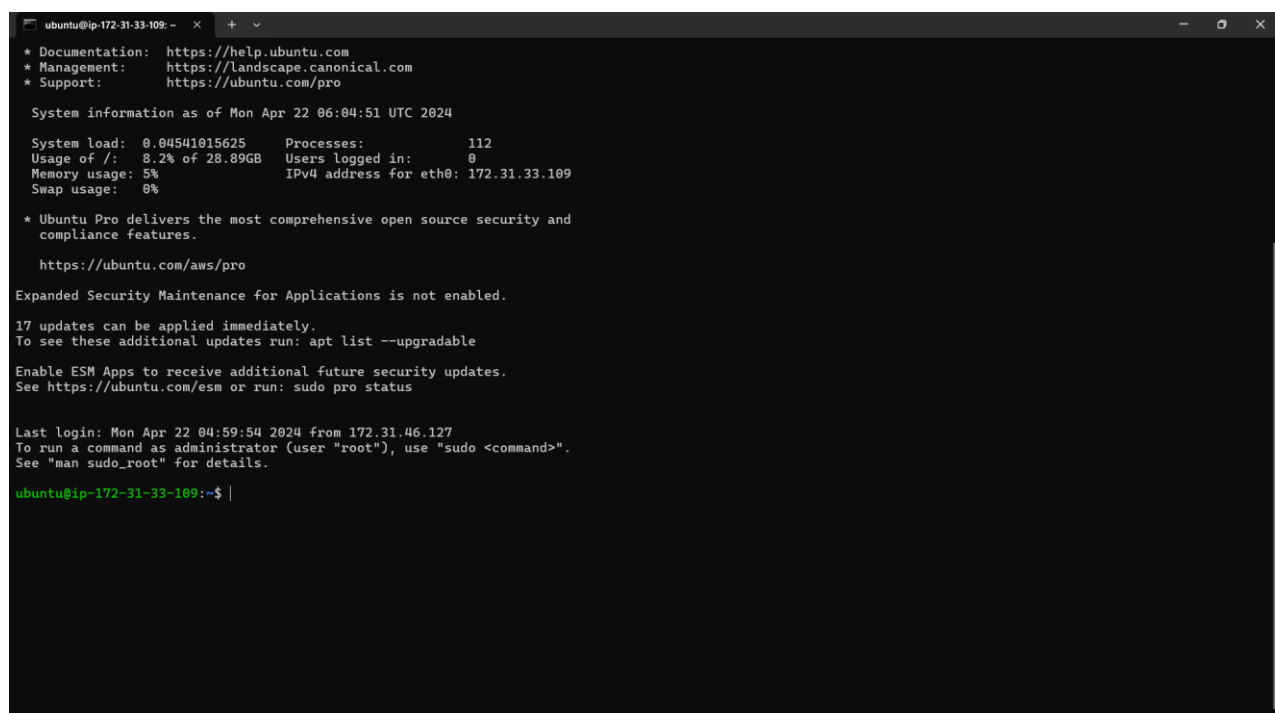


Fig 5.20 SSH server

In the ever-evolving landscape of cyber threats, safeguarding Industrial Internet of Things (IIoT) infrastructure is paramount to ensure uninterrupted operations and data integrity. Our project, fortified by a comprehensive array of security measures, stands as an impregnable fortress against a multitude of malicious attacks. Here's why common cyber threats such as Man-in-the-Middle (MITM), Brute Force, SQL Injection, Denial of Service (DoS), and Distributed Denial of Service (DDoS) attacks are rendered futile against our robust security infrastructure: To ensure the security of our project we conducted the following series of attacks.

- Brute Force attack
- Man in the middle attack
- SQL Injection attack
- Denial of service attack
- Distributed denial of service attack

Man-in-the-Middle (MITM) Attack

Our utilization of public key authentication in SSH, coupled with a VPN network enforced by OpenVPN, establishes a secure channel for communication between IIoT devices and the server. This encrypted tunnel prevents unauthorized interception or manipulation of data, rendering MITM attacks ineffective against our fortified defenses.

Brute Force Attack:

With SSH access restricted solely to devices within our authorized VPN network, the likelihood of successful brute force attacks is significantly mitigated. Additionally, the implementation of public key authentication instead of traditional password-based login methods adds an additional layer of security, making brute force attempts impractical and futile.

SQL Injection:

System incorporates rigorous measures to prevent SQL injection attacks, including parameterized queries and input validation techniques. By sanitizing user inputs and employing secure coding practices, we fortify our databases against exploitation, ensuring that attackers cannot manipulate SQL queries to gain unauthorized access or compromise data integrity.

Denial of Service (DoS) and Distributed Denial of Service (DDoS) Attacks

Through meticulous configuration of security groups, firewall rules, and traffic filtering mechanisms, we proactively thwart DoS and DDoS attacks before they can inflict harm. By limiting access to authorized users and employing AWS IAM for precise control over user privileges, we effectively mitigate the risk of resource exhaustion and service disruption caused by malicious traffic floods.

In essence, our IIoT security infrastructure stands as a formidable barrier against a myriad of cyber threats, ensuring the resilience and reliability of critical industrial operations. By implementing a layered defense approach that encompasses encryption, access controls, and proactive monitoring, we uphold the integrity and security of our systems, safeguarding against potential financial losses, reputational damage, and operational disruptions.

CHAPTER 6

CONCLUSION

The proposed system for bolstering IIoT cyber security through SSH and VPN integration offers a sophisticated and multi-layered defense mechanism against a wide array of cyber threats. By adopting public key authentication in SSH and employing a VPN network with OpenVPN, the system ensures that access to critical infrastructure is tightly controlled, limiting it to authorized users within the VPN network. This approach not only mitigates the risk of unauthorized access but also prevents potential attackers from intercepting sensitive data during transit.

Furthermore, the implementation of robust encryption protocols for IIoT traffic, coupled with stringent security measures such as firewall configurations and security group rules, fortifies the system's resilience against common cyberattacks including SQL injection, man-in-the-middle attacks, brute force attacks, DoS, DDoS, and keylogging. These proactive defense mechanisms not only safeguard the integrity and confidentiality of data but also minimize the likelihood of system compromise or disruption.

The integration of AWS IAM facilitates granular control over user privileges, ensuring that only authorized personnel with specific permissions can access critical resources. Additionally, the deployment of disaster recovery databases and the physical security measures associated with hosting the server in a secure data center in Ireland further bolster the system's resilience, guaranteeing the availability and integrity of data even in the face of unforeseen contingencies.

Comprehensive logging and monitoring mechanisms enable continuous surveillance of user activities and VPN connections, allowing for timely detection and response to potential security incidents. Moreover, the implementation of automated alerting mechanisms ensures that administrators are promptly notified of any unauthorized access attempts, enabling swift intervention to mitigate risks and safeguard the system.

By combining advanced encryption techniques, access controls, and monitoring capabilities, the system effectively mitigates cyber threats and ensures the uninterrupted functionality of IIoT infrastructure, thereby safeguarding against potential financial losses, reputational damage, and operational disruptions.

6.1 FUTURE SCOPE

While the proposed system provides a solid foundation for IIoT cybersecurity, there are several avenues for future exploration and enhancement:

Advanced Threat Detection:

Implement machine learning algorithms and anomaly detection techniques to enhance the system's ability to detect and respond to advanced cyber threats.

Zero Trust Architecture:

Explore the adoption of zero-trust principles to enforce strict access controls and verify the identity of all users and devices accessing the IIoT environment.

APPENDIX

Email Alert Automation Script

```
import smtplib

from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart

sender_email = "charusam3011@gmail.com"
recipient_email = "charusam3011@gmail.com"
smtp_server = "smtp.gmail.com"
smtp_port = 587
username = "charusam3011@gmail.com"
password = "qcxryerknensqlyz"
subject = "Test Email"
body = "Hi i am charu prakash"
message = MIMEMultipart()
message["From"] = sender_email
message["To"] = recipient_email
message["Subject"] = subject
message.attach(MIMEText(body, "plain"))

with smtplib.SMTP(smtp_server, smtp_port) as server:
    server.starttls()
    server.login(username, password)
    server.sendmail(sender_email, recipient_email, message.as_string())

print("Email sent successfully!")
```

REFERENCES

- 1.A Review of Industrial Internet of Things (IIoT) Security: Vulnerabilities, Counter measures, and Future (<https://ieeexplore.ieee.org/document/10100145>) by Mohan Kumar et al. (2018): This paper provides a comprehensive overview of IIoT security challenges and countermeasures.
- 2.Industrial Internet of Things (IIoT) Security: A Roadmap towards Secure Deployment (<https://www.mdpi.com/1424-8220/21/11/3901>) by Maohua Ge et al. (2020): This article outlines a roadmap for securing IIoT deployments, including considerations for authentication, authorization, and encryption.
- 3.Streun, Fabio,Wanner joel, perrig, Adrian” Evaluating Susceptibility of VPN Implementations to DoS Attacks Using Adversarial Testing”2022.
- 4.Jie Cao, Xing-Liang yuan, Jia-Cheng Fan and Chin-Ling Chen”A VPN-Encrypted Traffic Identification Method Based on Ensemble Learning”2022.
- 5.Suh Charles For bacha, Mbuya Josiah, Anyam Agwu ”Design and Implementation of a Secure Virtual Private Network Over an Open Network(Internet)”2023.
- 6.Andrea Pferscher,Benjamin Wunderling,Bernhard K,Aichernig and Edi Muskardin”Mining Digital Twins of a VPN Server”2023.
- 7.Antonio Francesco Gentile,Davide Macri,Floriano De Rango,Mauro Tropea and Emilio Greco”A VPN Performances Analysis of Contrained Hardware Open Source Infrastructure Deploy in IoT Environment”2022.
- 8.Sahas V G, Mujahid Khan, Siddesh G M,Vinay H Virtual Private Network (VPN) Router using Raspberry-PI”2023

9. Vojdan Kjorveziroski, Cristina Bernad, Katja Gilly, Sonja Filiposka "Full-mesh VPN performance evaluation for a secure edge-cloud continuum" 2024.
10. Sultan Almutairi, Yogev Neumann, Khaled Harfoush "Fingerprinting VPNs with Custom Router Firmware: A New Censorship Threat Model" 2024.
11. Amaldeep S, Sriram Sankaran "Cross Protocol Attack on IPSec-based VPN" 2023.
12. Aadil Khan, Ishu Sharma "Using Ensemble Learning for Securing public Area Networks from VPN Malicious Transmissions" 2023.
13. Vladimir Voicu, Dorin Petreus, Emil Cebuc, Radu Etz "Industrial IoT (IIOT) Architecture for Remote Solar Plant Monitoring" 2022.
14. Alin-Bogdan Popa "QGP-VPN: QKD enhanced VPN solution for general-purpose encrypted communications" 2023.
15. Daniele Canavese, Luca Mannella, Leonardo Regano and Cataldo Basile "Security at the Edge for Resource-Limited IoT Devices" 2024.
16. Tobias Pulls, Ethan Witwer "A Framework for Traffic Analysis Defenses" 2023.
17. Maryna Kolisnyk & Oleksandr Piskachov "Analysis and Systematization of Vulnerabilities of Drone Subsystems" 2023.
18. Niu Hu, Xuan Zhou, Bing xu, Hanqing Liu, Xiangjin Xie and Hai-Tao Zheng "Variation on Prompt Tuning for Named-Entity Recognition" 2023.
19. Michael L, Olive, Corey Snipes, Willard R, True, Todd Kilbourne "Cybersecurity Controls for Connected Aircraft Systems that Support Safety Services" 2022.

20. Ming Zhong, Wenbo Xie, Lei Yang "The optimization and Improvement of Campus Network Based on MSTP" 2023.
21. Fausto Ferreira, Dula Nad, Tomislav Jagust, Nikola Miskovic, F "Increasing sea knowledge awareness among high school students using marine robots" 2023.
22. Biplob Paul and Muzaffar Rao "Zero-Trust Model for Smart Manufacturing Industry" 2022.
23. The SSH Protocol for Secure Remote Login (<https://www.oreilly.com/library/view/ssh-the-secure/0596008953/>) by T. J. Mather (2001): This IETF document defines the SSH protocol for secure remote access.
24. Virtual Private Networks (VPNs): A Survey of Technologies (<https://www.sciencedirect.com/topics/engineering/virtual-private-networks>) by D. Medhi and A. Ray (2017): This paper surveys different VPN technologies and their applications.
25. AWS Identity and Access Management (IAM) (<https://docs.aws.amazon.com/iam/>)
26. AWS Identity and Access Management (IAM) (<https://docs.aws.amazon.com/iam/>).
27. Amazon Virtual Private Cloud (VPC) (<https://docs.aws.amazon.com/vpc/>).
28. AWS IoT Core (<https://aws.amazon.com/iot-core/>).
29. AWS Key Management Service (KMS) (<https://aws.amazon.com/kms/>).

30. AWS Security Best Practices

<https://docs.aws.amazon.com/whitepapers/latest/aws-security-best-practices/welcome.html>

31. AWS Well-Architected Framework. This framework provides guidance on designing and running secure, high-performing, and cost-effective AWS workloads.

32. CIS AWS Foundations Benchmark ([foundations-benchmark.html](https://docs.cisecurity.org/docsfiles/1.10.0/CIS-AWS-Foundation-Benchmark-v1.10.0.pdf)) This benchmark outlines best practices for securing AWS environments.

33. National Institute of Standards and Technology (NIST) Special Publication 800-161) This publication provides guidance for supply chain risk management, which is relevant for securing IIoT components.

34. Cloud Security Alliance (CSA) - Internet of Things (IoT) Top Threats (<https://cloudsecurityalliance.org/press-releases/2022/06/07/cloud-security-alliance-s-top-threats-to-cloud-computing-pandemic-11-report-finds-traditional-cloud-security-issues-becoming-less-concerning>) This document outlines the top threats to IoT deployments.

35. www.infrastrutturecritiche.it/wp-content/uploads/2019/06/SmartCity-convertito-unito.pdf

36. www.mdpi.com/1424-8220/22/15/5626