

SUAVNet: Blockchain-based SDN controller recovery and forwarding device migration for UAVs

by

**Afifa Islam Ananna (192014006)
Shanjida Akter (192014024)
Mahedul Hasan Rifat (192014005)
Towshick Intesir Ahmed Rain (193014017)**

*Capstone Project Report (CSE 499) submitted in partial fulfillment of the
requirements for the degree of*

Bachelor of Science in Computer Science and Engineering

Under the supervision of

Dr. Nafees Mansoor, SMIEEE



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY OF LIBERAL ARTS BANGLADESH**

SPRING 2024

© *Afifa Islam Ananna , Shanjida Akter , Mahedul Hasan Rifat and Towshick Intesir*
Ahmed Rain
All rights reserved

DECLARATION

Project Title	SUAVNet: Blockchain-based SDN controller recovery and forwarding device migration for UAVs
Authors	<i>Afifa Islam Ananna , Shanjida Akter , Mahedul Hasan Rifat and Towshick Intesir Ahmed Rain</i>
Student IDs	192014006, 192014024, 192014005, and 193014017
Supervisor	Dr. Nafees Mansoor, SMIEEE

We declare that this capstone project report entitled *SUAVNet: Blockchain-based SDN controller recovery and forwarding device migration for UAVs* is the result of our own work except as cited in the references. The capstone project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Afifa Islam Ananna
192014006

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Shanjida Akter
192014024

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Mahedul Hasan Rifat
192014005

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Towshick Intesir Ahmed Rain
193014017

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

Date: May 30, 2024



Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
688 Beribadh Road, Mohammadpur
Dhaka - 1207

APPROVAL OF THE COMMITTEE

This is to certify that the CSE 499 capstone project report entitled **SUAVNet: Blockchain-based SDN controller recovery and forwarding device migration for UAVs**, submitted by **Afifa Islam Ananna** (192014006), **Shanjida Akter** (192014024), **Mahedul Hasan Rifat** (192014005) and **Towshick Intesir Ahmed Rain** (193014017) are undergraduate students of the **Department of Computer Science and Engineering** has been examined. We hereby accord our approval of it as the presented work is satisfactory.

Nafees Mansoor, PhD, SMIEEE, Associate Professor
Dept. of CSE, University of Liberal Arts Bangladesh

Muhammad Golam Kibria, PhD, SMIEEE
Dept. of CSE, University of Liberal Arts Bangladesh

Shohag Barman, PhD, Assistant Professor
Dept. of CSE, University of Liberal Arts Bangladesh

Suravi Akhter, Lecturer
Dept. of CSE, University of Liberal Arts Bangladesh

Place: Dhaka

Date: May 30, 2024

ACKNOWLEDGEMENTS

Our profound gratitude goes out to Dr. Nafees Mansoor, our research supervisor, for giving us the chance to conduct research and for his essential advice during the whole project. His inspiring presence, unshakable dedication, visionary insight, and dynamic leadership have all been tremendous sources of inspiration. We learned how to do our work and convey it with the highest clarity under his tutelage. Being guided by him has been a great honor and privilege.

We have a great deal of gratitude for our wonderful teachers at the University of Liberal Arts Bangladesh's, Department of Computer Science and Engineering. There is no denying that their instruction and direction helped us grow into the people we are now.

We are incredibly grateful to our parents for their constant love, unfailing prayers, and selfless sacrifices, which have enabled us to receive an education and equipped us for the future. We also want to express our gratitude to our family and friends for their steadfast support and concern.

As a final note, we want to thank everyone who has helped us along the way.

*Afifa Islam Ananna , Shanjida Akter , Mahedul Hasan Rifat
and Towshick Intesir Ahmed Rain*

University of Liberal Arts Bangladesh

Date: May 30, 2024

Dedicated to
Allah and my Parents
– *Ajifa Islam Ananna*

Dedicated to
my loving Parents
– *Shanjida Akter*

Dedicated to
my loving Parents
– *Mahedul Hasan Rifat*

Dedicated to
my loving Parents
– *Towshick Intesir Ahmed Rain*

ABSTRACT

Unmanned Aerial Vehicles or UAVs, have transformed a number of industries in recent years including disaster management, agriculture, and delivery services. Software-defined Networking (SDN) can revolutionize UAV network management for its programmability, flexibility, and agility. In SDN, the segregation of the control plane from the data plane and programmability allows controllers to be independent from forwarding devices, enabling easier policy modification and network troubleshooting without system shutdowns. SDN's centralized controllers are vulnerable to SPOF (Single Points of Failure), risking whole network collapse by controller failure. To lessen the aforementioned issues, this system proposes a controller recovery scheme in SDN by using blockchain technology. The research also introduces a novel flow rule migration system for SDN forwarding devices based on blockchain. Combining controller recovery scheme and migration mechanism of flow rules for forwarding devices ensure uninterrupted UAV communication and mitigate security threats. Overall, the proposed system will be able to provide fault tolerance and security to guarantee QoS (Quality of Service).

Keywords: Software-defined Networking, Unmanned Aerial Vehicle, Single Point of Failure, Blockchain Technology

Contents

1	Introduction	1
1.1	Problem Statement	2
1.2	Aims, Objectives and Motivation	3
1.2.1	Aims	3
1.2.2	Objectives	3
1.2.3	Motivation	3
1.3	Project Specification	4
1.3.1	Functional Requirements	6
1.3.2	Non-Functional Requirements	7
1.4	Outcome	7
1.5	Report Layout	9
1.6	Summary	10
2	Literature Review	11
2.1	Preliminaries	11
2.1.1	Blockchain	11
2.1.2	Private Blockchain	12
2.1.3	Software-Defined Networking (SDN)	12
2.1.4	Hyperledger Besu	13
2.1.5	Ganache-CLI	14
2.1.6	Smart Contract	14
2.2	Related Works	15
2.3	Performance Metrics	23
2.4	Performance Analysis	26

3	Methodology	43
3.1	Introduction	43
3.2	System Development Model	44
3.2.1	Waterfall Model	44
3.3	System Analysis	45
3.3.1	System Design	46
3.3.2	Implementation	46
3.4	Requirement Engineering	46
3.5	Requirement Engineering Process	47
3.5.1	Feasibility Study	47
3.5.2	Requirement gathering	47
3.5.3	Software Requirement Specification	48
3.5.4	Software Requirement Validation	48
3.6	Software Requirement for Development	48
3.6.1	Microsoft Visual Studio Code	49
3.6.2	Remix IDE	49
3.6.3	Truffle	50
3.6.4	Ganache CLI	50
3.6.5	NodeJS	50
3.6.6	Mininet	51
3.6.7	Metamask	51
3.6.8	Hyperledger Besu	52
3.6.9	Docker	52
3.6.10	Web3.py	53
3.6.11	Lucid Chart	53
3.6.12	Mendeley	53
3.6.13	LaTeX	54
3.7	Required Programming Languages	54
3.7.1	Solidity	54
3.7.2	JavaScript	55
3.7.3	Python	55
3.8	Summary	55

4	Design and Development	56
4.1	Project Management and Financial Activities	56
4.1.1	Work Breakdown Structure	56
4.1.2	Estimate Costs	58
4.1.3	Budget Plan	60
4.1.4	Funding Plan	61
4.1.4.1	Government Grants	61
4.1.4.2	Investor	61
4.1.4.3	Crowdfunding	61
4.2	System Design and Description	62
4.2.1	SDN Controller self-recovery scheme	62
4.2.2	Blockchain-based flow rule migration for SDN forwarding devices	65
4.3	Implementation	68
4.3.1	Smart Contract	68
4.3.2	Deployment of Smart Contract in Hyperledger Besu	70
4.3.3	Ganache Blockchain Integration	75
4.3.4	Mininet Integration	76
4.4	UAV Registration User Interface (UI)	84
4.5	Proposed System's strengths and limitations	87
5	Business Model	89
5.1	Value Proposition	89
5.2	Customer Segments	89
5.2.1	Commercial UAV Operators	90
5.2.2	Government and Defense	90
5.2.3	Telecommunication Companies	90
5.2.4	Emergency Services	90
5.2.5	Environmental Monitoring	91
5.3	Channels	91
5.3.1	Direct Sales	91
5.3.2	Online Platform	91
5.3.3	Digital Marketing	92
5.3.4	Reseller and Distributor Networks	92

5.3.5	Customer Support and Success	92
5.4	Customer Relationships	92
5.4.1	Online Support and Self-Service	92
5.4.2	Customer Success Programs	93
5.4.3	Loyalty and Rewards Programs	93
5.5	Revenue Streams	93
5.5.1	Licensing Fees	93
5.5.2	Partner and Reseller Programs	93
5.5.3	Consulting Services	94
5.5.4	Service Contracts	94
5.5.5	Performance-Based Pricing	94
5.6	Key Resources	94
5.6.1	Technology Infrastructure	95
5.6.2	Human Resources	95
5.6.3	Intellectual Property	95
5.7	Key Activities	95
5.7.1	Research and Development	96
5.7.2	System Design and Architecture	96
5.7.3	Implementation and Deployment	96
5.7.4	Operations and Maintenance	96
5.7.5	Customer Support and Service	96
5.7.6	Marketing and Sales	97
5.8	Key Partnership	97
5.8.1	Technology Providers	97
5.8.2	Investors and Venture Capitalists	97
5.8.3	Software and Platform Developers	97
5.8.4	Funding and Grants	97
5.8.5	Industry Associations and Logistics	98
5.9	Cost Structure	98
5.9.1	Research and Development Costs	98
5.9.2	Infrastructure Costs	98
5.9.3	Operational Costs	98
5.9.4	Marketing and Sales Costs	98
5.9.5	Customer Support and Service Costs	99

5.9.6	Partnership and Collaboration Costs	99
5.9.7	Financial and Other Costs	99
6	Result Analysis	100
6.1	Performance Evaluation	100
6.1.1	Configuration	100
6.1.2	Metric	101
6.2	Proposed System's Performance Results	102
6.2.1	Performance Comparison	105
7	Conclusion	106
7.1	Social, Ethical, and Environmental aspects	106
7.1.1	Social	106
7.1.2	Ethical	107
7.1.3	Environmental	108
7.2	Brief Summary	109
7.3	Future works	110
	References	112

List of Figures

1.1	Block Diagram of the Proposed System	4
3.1	Waterfall Model	44
4.1	Work Breakdown Structure Flowchart	57
4.2	Use case diagram of the proposed system	62
4.3	Activity Diagram of Controller Recovery	64
4.4	Sequence diagram for flow rule migration for SDN forwarding devices	66
4.5	ER Diagram of the proposed system	67
4.6	Besu Project Tree Structure	71
4.7	Generated ABI from Smart Contract Compilation	72
4.8	Smart Contract Deployer	73
4.9	Successful Deployment of the developed Smart Contract	73
4.10	Transaction Record After Creating a Rule	74
4.11	Fetch Rule Information	74
4.12	Ganache-CLI running listening to a specific port	75
4.13	Onos service running	76
4.14	Onos service user login	77
4.15	Mininet topology running	78
4.16	Flow rule structure	79
4.17	Flow rule stored in a forwarding device of Mininet	80
4.18	Successful transaction of the flow rule in the blockchain fetched from Mininet	80
4.19	Fetched flow rule information from blockchain	81
4.20	Flow Rules dumped from forwarding devices	82
4.21	Comparison between flow rule from forwarding devices and block data	83

4.22	Homepage	84
4.23	Register page	85
4.24	Login page	86
4.25	Database	87
6.1	Flow Creation Function	102
6.2	Flow Migration Function	103
6.3	Flow Fetch Compare Function	104

List of Tables

2.1	Comparison of Literature Review	41
4.1	Cost Breakdown	60

Chapter 1

Introduction

UAVs are becoming more popular in both commercial and civilian applications [Mansoor et al. \(2023\)](#). In fact they are now more serviceable because of their recent developments in networking, communication, and manufacturing processes [Alladi et al. \(2020\)](#). Since UAVs are inexpensive, they can be quickly deployed in a range of topographical conditions [H. Wang et al. \(2017\)](#). For example, in applications like crowdsensing, healthcare, rescue, and surveillance. However, mobility poses challenges for network management [L. Gupta et al. \(2015\)](#). In this case, network softwarization, also known as SDN, which separates hardware from network control functions, becomes essential. By coordinating SDN with UAVs, a programmable network based on a centralized control paradigm is created, reducing data layer overhead and enhancing UAV network flexibility [Sultana et al. \(2019\)](#).

Network characteristics such as programmability, flexibility, and availability are altered by SDN, which also speeds up the expansion of the internet and networks through its logical controller centralization and global network overview [Schehlmann et al. \(2014\)](#). SDN aims to address the limitations of current network infrastructure. The separation of the control and data planes simplifies policy enforcement, network reconfiguration, and network evolution by converting network switches into basic forwarding devices and implementing the network control logic in a centralized controller. However, centralized control can lead to a single point of failure (SPOF), causing network standstill if the controller malfunctions or is compromised [M. A. Khan et al. \(2022\)](#).

A multitude of security and privacy concerns, such as denial-of-service attacks, software flaws, unauthorized access to SDN controllers, UAV hijacking etc are brought up by the growing use and deployment of softwarized UAVs [H. Liu et al.](#)

(2018). Due to SPOF in traditional SDN systems which mentioned previously, UAV operations are vulnerable to interruptions. This research examines a blockchain-based SDN controller recovery solution designed especially for UAVs as a response to these issues.

Blockchain, a modern technology has rapidly gained popularity as a novel and promising approach to securely store, exchange, and manage data [T. Rahman et al. \(2023\)](#). It is an incredible technology that accomplishes decentralization and security with anonymity. It is built on distributed ledgers, asymmetric encryption, consensus mechanisms, smart contracts and other fundamental technologies that are independent of any third party [Qiao et al. \(2022\)](#). The decentralization and immutable nature of blockchain technology ensures security, reliability, and transparency which will help to manage UAV network operations. For example, in the event of a controller failure, the blockchain can facilitate secure recovery of the controller state and ensure continuous operation of UAVs. This is why the research combines blockchain with SDN to enhance security in UAV networks and address SPOF recovery. Additionally, it also introduces a Blockchain-based flow rule migration scheme for SDN forwarding devices. As a result, the proposed system bridges the existing gaps and offers a secure and effective solution.

1.1 Problem Statement

The dependence on centralized controllers in conventional SDN models makes critical vulnerabilities, primarily due to the single point of failure (SPOF) issue. This can lead to disastrous network disruptions, compromising the reliability and security of the entire UAV network. Current systems lack enough studies that address vulnerabilities in centralized controllers, leaving UAV networks susceptible to attacks and operational downtimes. To ensure the robustness of SDN deployments, there is an urgent need for a self-recoverable and flexible SDN controller scheme. Introducing such solution is essential to tackle SPOF risks and improve overall network reliability.

1.2 Aims, Objectives and Motivation

1.2.1 Aims

The proposed scheme aims to design and implement controller recovery and migration mechanism of flow rules using blockchain and SDN. The system will enable uninterrupted UAV communication and mitigate security threats. The use of blockchain technology will ensure the immutability and transparency of the flow rules stored in the network. Additionally, the system will be able to provide fault tolerance and security to ensure QoS.

1.2.2 Objectives

- Develop a Blockchain-based self-recoverable SDN controller scheme for UAVs.
- Store the flow rules as immutable blocks in the blockchain.
- Develop a flow rule migration system for SDN forwarding devices to ensure uninterrupted communication of UAVs.

1.2.3 Motivation

As networks and their requirements grow, managing hardware switches and manually setting up network software switches for businesses with highly virtual systems has become increasingly complex and time-consuming [Lotlikar & Shah \(2019\)](#). Traditional networks are complex and challenging to operate, as configuring them according to predefined procedures and adjusting for load changes and faults through reconfiguration is difficult [Haji et al. \(2021\)](#). Traditional networks statically manage switches, routers, and other components which become inflexible and inadequate for today's application demands [Bwalya & Zimba \(2021\)](#). SDN is a technology that offers software-enabled dynamic network configuration and management by programmability to improve network monitoring and performance [Kumari et al. \(2020\)](#).

SDN reduces these challenges by simplifying network management, breaking through the limitations of traditional networks, and facilitating by introducing new networking concepts and developments. Thus this system is proposed in this research.

1.3 Project Specification

The Prototype of a Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs is described in this section. The system presents an efficient scheme for recovering UAV controllers in Software-Defined Networking (SDN) using blockchain technology. Additionally, a novel solution for migrating flow rules from one forwarding device to another using blockchain has been introduced.

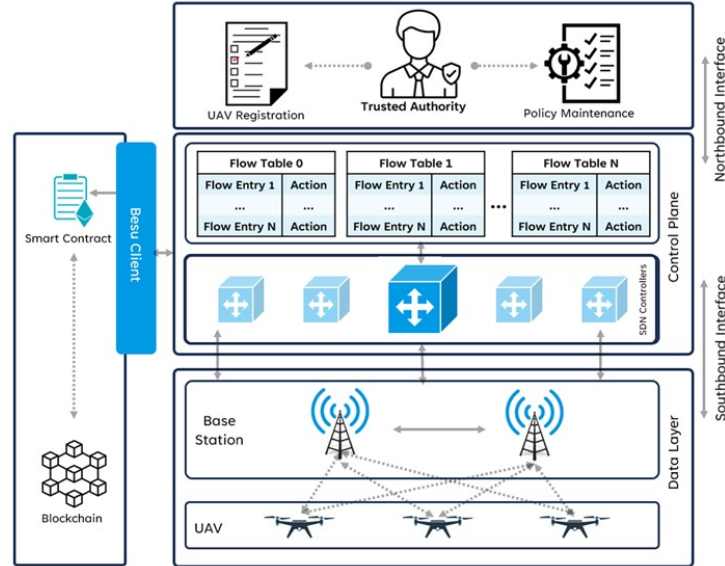


Figure 1.1: Block Diagram of the Proposed System

Figure 1 shows the block diagram of the proposed system. Regarding UAV communication, the system assumes the following:

- UAVs will utilize secure communication channels to interact with the forwarding devices within the SDN. Public key cryptography may be employed to establish this secure communication.
- A trusted authority (TA) will oversee UAV registration in the network. The TA will receive UAV id, model and owner email address to generate communication keys. Each UAV will be assigned its private and public key by the TA, along with public keys of forwarding devices for packet encryption.

Forwarding devices will decrypt incoming data from UAVs using their private keys.

A. SDN controller self-recovery scheme: In the proposed scheme, the SDN controller gains the ability to recover itself from a single point of failure by leveraging its forwarding devices and blockchain technology. This ensures the SDN-enabled UAV communication system's fault tolerance and uninterrupted service delivery. Whenever the controller generates a new flow rule, it captures the unique fields from the rule and securely stores this data in the blockchain. In the event of a single point of failure, the controller can restore itself to its previous state by consulting the blockchain to retrieve flow rule information stored in blocks, comparing it with the existing flow rules in the forwarding devices. In scenarios where the SDN topology incorporates multiple controllers, each controller serves as a node in the blockchain network. Whenever a controller initiates a new flow rule, other controllers in the network rely on consensus mechanisms to validate and append each rule as a block in the chain. This approach ensures transparency and traceability within the system. In the context of an SDN system utilizing an ONOS controller in Mininet, specific fields of a flow rule, such as flow ID, device ID, MAC source and destination, flow priority, and protocol type, remain immutable throughout the lifetime of the rule and are stored as blocks in the blockchain for each rule.

In the event of a single point of failure, where the controller loses memory of previous records, it initiates the recovery process. Assuming the failed controller commenced its service at time T_0 and the failure occurred at T_f , the controller retrieves all block data from the chain between T_f and T_0 . It then cross-references each rule from the chain with the stored rules in the forwarding devices. If a match is found, the data is reintegrated into the controller's operation.

B. Flow rule migration for SDN forwarding Devices: Forwarding devices in SDN are hardware components with a limited lifespan. Over time, they may require maintenance or replacement. Thus, the proposed system introduces a mechanism where the flow rules of a forwarding device can be migrated to a new device, and the updates will be reflected in the blockchain network.

For the migration, a trusted authority will be empowered to execute this functionality. The system will retrieve the flow rules from the old forwarding device that

needs flow rule migration and will receive the new forwarding device ID from the trusted authority. Subsequently, it will update the device ID field for all matching rules and create new blocks in the blockchain network. Also, all the flow rules stored in a specific forwarding device can be fetched from the blockchain using the deployed smart contract in the network.

1.3.1 Functional Requirements

- **UAV Registration:** The system should have a verification mechanism for UAVs to be able to retrieve and share communication keys with forwarding devices of SDN.
- **Data Storage:** The system has to be able to store all registration and generated communication key information in a local storage.
- **Recovery Mechanism:** The SDN controller should have a mechanism to recover its state using information from the blockchain.
- **Flow-Rule dump mechanism:** The SDN controller should be able to extract unique fields from each new flow rule it creates.
- **Flow Rule Verification:** The SDN controller should be able to validate that the flow rules are consistent with stored block data in the blockchain to ensure no discrepancies exist.
- **Blockchain Node Operation:** Each SDN controller should operate as a node and take part in the consensus process in the blockchain network.
- **Blockchain Storage:** The SDN controller should be able to store the extracted flow rule information in the blockchain.
- **Flow-Rule Migration:** The system should be able to allow flow rule migration from one forwarding device to another and update the previous block data in the blockchain accordingly.

1.3.2 Non-Functional Requirements

- **Security:** The system should be able to determine flow rule authenticity stored within the forwarding devices while recovering itself and mark them as malicious if any discrepancy is found.
- **Immutability:** The system should be able to deliver data immutability as it stores the flow rules as immutable blocks in the blockchain.
- **Decentralization:** The controllers get to participate in the consensus mechanism before storing any flow rule in the chain, which ensures equity between the controllers and removes the need of a middle authority.
- **Fault tolerance:** In the system, an SDN controller should be able to recover itself from SPOF autonomously and continue its expected functions, which solves the centralization problem.
- **Reliability:** The system should be reliable enough to recover the controller from SPOF and ensure smooth migration of rules between forwarding devices.
- **Integrity:** As the registration process of the devices participating in the network is handled by a trusted authority, the system should be able to provide data integrity to the controller for accurate forwarding decisions.
- **Flexibility:** The system should be able to maintain balance, ensure continuity in unexpected events, and cope with uncertainty.
- **Programmability:** The system should be able to provide programmability of packet forwarding behavior within the SDN network.

1.4 Outcome

The deployment of the system, Blockchain-based controller recovery and SDN forwarding device rule migration for UAVs has shown encouraging results in addressing the traditional network issues. And those outcomes are given below:

- **Dynamic and Efficient Network Management:** SDN enables real-time programmability of devices and networks, resulting in improved performance

and adaptability to changing conditions. And smart contracts within the blockchain automated the handover process between controllers, reduced downtime and ensured seamless operation.

- **Enhanced Fault Tolerance and Network Resilience:** The integration of blockchain technology with SDN, a decentralized mechanism for controller recovery which significantly reduced the risk of SPOF. This ensures that the UAV network remains operational even in the event of a controller failure. Blockchain's distributed ledger stores multiple copies of data across various nodes, providing a robust backup system that can quickly restore network functions if a failure occurs.
- **Improved Security and Trust:** Blockchain's immutable ledger ensures that flow rules are securely recorded and cannot be altered or tampered with, which enhances the trustworthiness and integrity of the UAV network. By using blockchain, flow rules are stored securely which ensures better network management.
- **Flexible Policy Management:** Network policies can be easily modified and enforced through the SDN controller, providing the flexibility needed to adapt to new operational requirements or threats. As all actions and changes in the network are transparently recorded on the blockchain, allowing for easy auditing and monitoring the whole network.
- **Improved Coordination and Collaboration:** Blockchain facilitates better coordination between different SDN controllers, providing a transparent platform for sharing state information and ensuring consistent network policies across the UAV network. Also the decentralized nature of blockchain enables collaborative decision-making processes among multiple controllers and nodes, enhancing the overall network efficiency and decision accuracy.
- **Quality of Service Assurance:** SDN can dynamically allocate network resources based on real-time data, ensuring that UAVs receive the necessary information required for their operations. The proposed system can maintain high levels of service even during network disruptions or controller failure.

The proposed Blockchain-based SDN controller recovery and rule migration system for UAVs offers a comprehensive solution to enhance the fault tolerance, security, and efficiency of UAV networks. By using both blockchain and SDN technologies, the system ensures uninterrupted communication, robust security measures, and high-quality service.

1.5 Report Layout

Chapter 1 outlines the problem statement, aims, objectives, motivation, project specification, functional- nonfunctional requirement, outcome and summary of the report.

Chapter 2 focuses on the preliminaries, related works, comparative analysis, performance analysis and comparative analysis table of the proposed system. Basically it presents literature reviews.

Chapter 3 explores the specifics of requirements analysis, use case modeling, and implementation requirements. It specifies the actions and factors that must be taken into account for the project to be implemented properly.

Chapter 4 analyzes the project's interface design, user experience, and front-end and back-end design specs. This chapter offers details on the system's intended aesthetic and functional layout.

Chapter 5 covers the project's budget phase. It reviews the findings and discusses their importance of creating a budget. It describes every phase of the budget broadly with proper justification.

Chapter 6 shows the result and analysis of the proposed system. It compares the results with both public and private blockchain and which one performs better.

Chapter 7 investigates the project's larger effects on the community and the environment. It discusses moral questions, environmental concerns, and the project's long-term viability. At the chapter's conclusion, the project's overall activities are summarized, along with future plans and prospective advancements.

1.6 Summary

In this chapter, the problem statement, aim, motivation, functional and non-functional requirements, importance and short overview of the proposed system are described.

Chapter 2

Literature Review

2.1 Preliminaries

In this section, the following topics have been covered in relation to developing Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs.

2.1.1 Blockchain

Blockchain technology acts as a distributed ledger, providing a secure, decentralized, and transparent platform for managing data [Kumutha & Jayalakshmi \(2022\)](#). Due to its immutable and tamper-proof nature, blockchain provides an excellent solution for managing controller recovery and flow rule migration in SDN for UAVs. This system is able to store the flow rules records in the blockchain, from which the controller can verify and compare the authenticity of the rules during the recovery process and if a forwarding device fails, blockchain technology can enable the seamless migration of its flow rules to another device, ensuring network resilience and integrity. Blockchain technology provides a reliable and efficient method for managing SDN controller recovery and flow rule migration for SDN forwarding devices. Because of blockchain's decentralized architecture, data security and integrity are maintained since no single entity can alter the data [Angelis & Da Silva \(2019\)](#). Furthermore, a secure and trustworthy method of maintaining UAV network configurations is guaranteed by the intrinsic resistance of blockchain technology to malicious attacks. This robust security framework prevents unauthorized access

and modifications, making it an optimal choice for critical applications in UAV networks.

2.1.2 Private Blockchain

These types of blockchains are not accessible to the general public and are restricted to a specific group of individuals or organizations. The ledger is shared exclusively among its participating members [Sarmah \(2018\)](#). Private blockchain is used for Blockchain-based controller recovery and SDN forwarding device rule migration systems for UAVs due to its inherent characteristics of security, permissioned access, and control. A private blockchain makes sure that only people with permission, such network administrators, UAV operators, and reliable authorities, may view and approve transactions on the network. This permissioned model enhances data privacy and integrity, crucial for maintaining the reliability and trustworthiness of the SDN infrastructure supporting UAV communications. Regarding the private blockchain, there isn't any mining, proof of work, or no remuneration [Guegan \(2017\)](#). Through the utilization of private blockchain's security and control functionalities, the system can efficiently reduce single points of failure, guarantee open rule management, and enable smooth device migration, all of which contribute to the increased resilience and effectiveness of SDN-enabled UAV communication systems.

2.1.3 Software-Defined Networking (SDN)

The acronym "SDN" stands for "Software-Defined Networking" representing a centralized networking paradigm where network control is logically decoupled from the physical infrastructure and managed through software applications. This separation offers various advantages, such as streamlining network management and control processes [Amin et al. \(2018\)](#). SDN architecture has three main components: the application layer, the controller, and the data plane. Applications interact with the controller to set network rules and manage traffic. The SDN controller serves as the focal point of the network, providing instructions to the data plane regarding flow forwarding or modification. Additionally, it handles the conversion of commands from applications into the lower-level communication protocol utilized by the data plane devices [Cabaj et al. \(2014\)](#). The controller is

empowered to generate new flow rules and effect modifications to flow tables as necessary. The data plane handles packet forwarding based on predefined flow rules stored in flow tables. While SDN provides flexibility and programmability, its centralized controllers risk network collapse due to Single Points of Failure (SPOF). By integrating blockchain into SDN architecture, blockchain decentralizes the control plane, preventing single-point failures while maintaining network-wide visibility, authenticating application flows, securing configurations, and enhancing resource protection and communication within SDN [Turner et al. \(2023\)](#). The SDN controller self-recovery scheme stores flow rules in blockchain, enabling rapid recovery. Additionally, Blockchain-based flow rule migration facilitates seamless updates to forwarding devices, ensuring uninterrupted communication and enhancing network resilience. This integration guarantees Quality of Service (QoS) while providing transparent and traceable network management.

2.1.4 Hyperledger Besu

The Hyperledger Besu is an open-source and permissioned blockchain technology designed for commercial application. It offers robust security and identification features, runs user-defined smart contracts, and has a modular architecture with pluggable consensus methods [Cachin et al. \(2016\)](#). Hyperledger improves security and access control by guaranteeing that only authorized parties, like reputable authorities and network administrators, are able to participate in the SDN, controllers record flow rules on the Hyperledger blockchain, allowing for recovery by comparison of stored rules with forwarding devices. As a blockchain node, every controller adds and verifies rules by consensus to ensure transparency. Rules are moved to new devices via flow rule migration, and changes are safely stored in the blockchain. Smart contract deployed in Hyperledger besu private blockchain is used to automate this procedure, ensuring accurate and consistent rule updates. The system ensures uninterrupted UAV connectivity, minimizing security concerns and guaranteeing quality of service.

2.1.5 Ganache-CLI

The ethereum blockchain emulator Ganache is used for developing and testing decentralized apps and smart contracts. Ganache CLI is a command-line interface tool for Ganache and it launches the local test blockchain, allowing the bash console to be used for intermediate smart contract testing [Moldamurat et al. \(2024\)](#). It provides a local blockchain environment that simulates the Ethereum network, allowing developers to deploy and interact with smart contracts without needing a live testnet. It is crucial to the Blockchain-based SDN controller recovery and forwarding device migration system for UAVs because it makes possible to create and thoroughly test the smart contracts that are the foundation of the system's operations. Ganache CLI makes sure these smart contracts are dependable and operate as intended before deploying them on a live blockchain, increasing the system's resilience and fault tolerance by simulating a real blockchain environment with adjustable block creation times and difficulty levels.

2.1.6 Smart Contract

Smart contracts are executable codes on the blockchain that automate and enforce agreements between parties without a trusted third party [S. N. Khan et al. \(2021\)](#). They manage the storage, retrieval, and validation of flow rules on the blockchain, ensuring that only valid and authorized updates are applied to the network. The risk of fraudulent activity is decreased when smart contracts are used since they do not require intermediaries or central authorities [S. Wang et al. \(2019\)](#). Smart contracts can automate the recovery procedure in the event of a controller failure by reapplying the last known good state to a repaired or replacement controller after collecting flow rules from the blockchain. For flow rule migration, smart contracts facilitate the transfer of rules between forwarding devices, ensuring that these updates are securely and accurately reflected on the blockchain. UAV communication and network performance are guaranteed by this automation, which also improves fault tolerance and security.

2.2 Related Works

In [Kumar et al. \(2022\)](#), the proposed system comprises two models: the Network Model and the Threat Model. In the Network Model, six entities are involved: Trusted Party (Tp), SDN Controllers (SDNc), OpenFlow Switches (Ofs), Unmanned Aerial Vehicles (UAV), Flow Analyzer (FA), and Blockchain (BC). Tp registers UAVs and Ofs, Incoming packets from UAV to SDN are first sent to the FA then FA examines the packet's legitimacy, if valid then sent to the SDN, otherwise the packet will be rejected and the forwarding OFs will be blacklisted. The SDN controller uses the packet to create a transaction in BC which acts as a distributed ledger and other controllers verify the transaction using PoA consensus. The proposed framework is built on a permissioned blockchain. In the Threat Model, Tp and SDNc are trusted, while UAVs and OFs are deemed untrusted, prompting the use of the 'Dolev-Yao' threat model to detect malicious behaviour. Additionally, 'Canetti and Krawczyk's Adversary model has been used to detect an adversary who can steal secret credentials and hijack session key or state on an ongoing session between two network participants.

In [R. Gupta et al. \(2021\)](#), The system consists of four different phases: registration, data, UAV and data controller phase. Devices, UAVs, and switches need to register themselves with the registration authority. Kerberos authentication protocol was used by the registration authority to establish secure communication between different entities of the system. IoT devices receive a communication key to initiate communication with UAVs. UAVs have a secure code from RA to check the identity of devices and other UAVs. Also, they can also check for eavesdroppers using K-means unsupervised clustering algorithm. Received data from devices are passed to the controller after checking the authenticity of UAVs and devices and if found valid, then the data gets added to a private blockchain, otherwise, discarded.

The authors in [N. Hu et al. \(2021\)](#), aimed to establish an efficient communication framework for UAVs through a multi-layered architecture comprising a data plane, control plane, and application plane. The data plane consists of a large number of UAVs and they communicate with the control plane in two ways: Satellite network and terrestrial Base station. For dynamic configuration, network upgrade and compatibility of the data plane, the P4 programming language has been proposed. The control plane, operating from ground stations, is segregated from the Data

plane through a standardized Control Data Plane Interface (CDPI). The control plane consists of multiple ground control stations and all stations form a consortium blockchain and use PBFT consensus to synchronize control data. Blockchain stores data plane and application plane data, routing schedules, and flow table calculation logic as smart contracts. For specific UAV operations, a special task id can be bound with the control information (Flow table) so that a UAV receives flow table information related to only its task from the controller. Moreover, the application plane consists of end-user business applications that consume UAV network resources and APIs like RESTFUL, and OSGi provided by the control plane are used to realize specific applications, such as routing policy calculation, QoS, VNF, central policy management, resource allocation and mobility management.

In the article [Liao et al. \(2021\)](#), the authors aimed to enable multi-drone services in smart cities by proposing an SDN-based architecture coupled with blockchain for enhanced cooperation and interoperability between drone controllers. The architecture comprises three planes: Application, Control, and Data. Application plane allows SD-IoD users to deploy drone applications that require multi-drone collaboration without worrying about the complexity, collaboration and interoperability between different service providers. The Control Plane houses SDN controllers responsible for managing drones owned by service providers, with a subset forming a consortium blockchain to achieve consensus. A novel consensus mechanism called Proof of Security Guarantee (PoSG) is introduced, incentivizing validators with rewards and ensuring reliable block validation. Furthermore, a digital currency called "Cooperation coin" is proposed to facilitate payment for drone services requiring collaboration between multiple service providers. Lastly, the Data Plane contains networking devices such as switches and drones, enabling communication between controllers and drones via the southbound interface for efficient mission execution.

In [Aldaej et al. \(2023\)](#), SDN-enabled UAV-IoT networks are examined to route secure traffic across different internet domains. To address network privacy problems, the utilization of abstract topology is done. Specifically, a smart contract is used to upload the abstract topology from the SDN controllers to the blockchain. Abstract topology stores topological details of the UAVs at a specific time, including information on the connected devices, IP addresses, hops and device stats. Virtual inter-domain and intra-domain links are essential components of Software-Defined

Networking (SDN) that enable dynamic resource allocation, traffic routing, and efficient network management. Moreover, the incorporation of validation procedures and a voting mechanism into the abstract topology ensures accuracy and reputation management in the current application domain. Reputation management is based on 2 factors: a local and global reputation. Using controllers to authenticate abstract topology, the smart contract generates a local reputation, which is forwarded to the controllers. When the intelligent contract adds up all of these local reputations, it comes up with the overall global reputation.

In [Derhab et al. \(2021\)](#), the network consists of N number of SDN controllers and domains made of data layer elements. Each controller acts as a master for one domain and is redundant for more than one domain, a redundant controller can replace the master controller in case of failure. All the controllers act as nodes in the blockchain. The master controller creates a block using the flow, and topology information of the network and the redundant controllers participate in the consensus process. If consensus is reached then the block gets added to the chain. Otherwise, the block is rejected and the master and redundant controllers that validated the block get their reputation value deducted.

The research article [Misra et al. \(2020\)](#), a blockchain-based solution addresses SDN controller Single Point of Failure (SPOF) by enabling recovery. When SPOF occurs, the controller requests flow rules from switches. The switches provide flow rules, and the controller performs a union operation to build the table. To authenticate received rules, the controller consults the blockchain for metadata, comparing and verifying against stored information. During block creation, the controller signs metadata and sends both rule and signed metadata to the intended switch, while other switches only receive signed metadata. The intended switch signs the metadata and shares it with other entities. Switches compare received metadata against controller-signed metadata, using proof-of-authentication for consensus. If consensus is achieved, the metadata is added to the blockchain as a block, ensuring a secure and resilient SDN environment.

In the article [Boukria et al. \(2019\)](#), a private blockchain is formed by an SDN controller and trusted nodes, where the controller can write new blocks, and trusted nodes can only read information. The authors propose two security approaches: attack detection and prevention. In attack detection, Vswitch requests a new rule,

the controller processes it, stores it in the blockchain, and VMFw (Firewall) compares rules for anomalies, notifying the administrator if found. In attack prevention, Vswitch must await VMFw approval before executing newly created rules to mitigate potential attacks. This ensures a secure SDN environment with proactive and reactive measures for network protection.

In [Zeng et al. \(2022\)](#), A blockchain-based solution for sharing local topology information by SDN controllers to form a global network view that can be used for cross-domain path calculation. To not expose the whole network topology, the SDN controllers only share the abstract view of the local topology and other controllers of the network verify the information by sending ICMP packets to detect the inter-domain links. As determining intra-domain links is not possible in this approach, if at least half the controllers reach consensus, then the topology information gets added to the blockchain and the controller that shared correct information receives an increase in its reputation. Local reputation values are stored in the controllers while global reputation values go to the blockchain and are used to determine the optimal routing path for packet forwarding through the network.

In the article [A. Rahman et al. \(2021\)](#), the authors proposed a three-layer SDN architecture tailored for IoT, comprising perception, edge, and cloud layers. Within the perception layer, IoT device clusters autonomously select cluster heads for efficient packet forwarding, considering proximity and energy levels to optimize power consumption. Blockchain keeps a record of the flow rules and makes them distributed and consistent among all the controllers. The authors developed an application to detect a compromised switch. This application dumps the stored rules of the switches periodically, hashes them and compares them with each other and also against the rules stored in the blockchain. If anomalies are detected, a controller creates a new rule and broadcasts it to block the compromised switch communication and isolate it from the network.

In the article [Mendiboure et al. \(2020\)](#), the author's focus was on enhancing authentication, access control, and scalability in software-defined vehicular networking. To manage the vehicles/SDN controllers/RSUs the system is broken into three types of blockchain nodes: local, inter-domain and global. Local nodes manage devices within a sub-network, while inter-domain nodes authenticate vehicles crossing domains, storing certificates in relevant sub-networks and Global nodes help local sub-networks to retrieve global status information. X.509 certificates

are utilized for device registration within sub-networks, stored in local blockchain networks, with TLS handshake employed for local device authentication. Inter-domain nodes validate certificate authenticity during cross-domain travel, updating certificates in new sub-networks if deemed valid. For access control, RSUs are attached to geographical areas and SDN controllers operate in specified areas, vehicle communication checks for geographical correspondence with them, updating if needed in the blockchain. SDN controller connections are denied if the vehicle's current area is unauthorized.

In [Khalid et al. \(2023\)](#), the author provides a scalable solution and access control policies for IoT devices. The main purpose of this system is to guarantee confidentiality, only approved users can get access to mitigate existing predicaments of IoT-related access control policy management and security. The system uses access control contracts for peer node access management, Judges contracts for misconduct assessment and penalty determination, and registers contracts as a lookup table for all executed methods. Their primary contribution is to provide a scalable solution for automated access control by storing the access control policies in the blockchain and managing the network traffic efficiency through SDN. Solidity is used as the language for creating smart contracts. A javascript server is used as a go-between to fix any problems that come up when the blockchain and SD-IoT interact. Lastly, they evaluated in terms of throughput and resource access delay between the blockchain component and the controller. Results show that an increase in nodes significantly impacts an increase in resource access delay while having an inconsequential impact on throughput.

In [Lokesh & Rajagopalan \(2020\)](#), the proposed security model for SDN controllers employs blockchain technology, specifically utilizing a cluster model with two independent blockchain networks: Application Blockchain (ABC) and Device Blockchain (DBC). Each cluster has its own set of ABC and DBC, while a common Controllers Blockchain (CBC) is shared among all clusters. Each SDN controller instance possesses a unique public key and private key for secure communication. OpenFlow-enabled devices establish a control channel by obtaining a session key from their respective controller instances. Devices encrypt packets using the session key before transmitting them to the controller, which subsequently decrypts and records the information in the operational data store. Applications validate their operations among controllers using ABC, operating on configuration data stores

and aiding in the generation of new flow rules. Controllers, after validating themselves, add digital signatures to a common CBC and encrypt transactions using device-specific session keys, generating a Merkle hash tree for all transactions in the DBC.

In the article [Faizullah et al. \(2020\)](#), the authors presented a blockchain-based security and access control mechanism for IoT networks by using SDN. In this paper, there were different servers (controllers), each managing a group of IoT thin client devices (forwarding devices). To maintain the same state without any central entity, blockchain technology is used for its decentralized and immutable nature. Blockchain will store information about the access policy and security mechanism. For security purposes, the system will check if the request comes from a valid user or not by matching the policy of request with the stored policy in the blockchain so that no malicious request or security attacks can interrupt the IoT network. In the access control mechanism, blockchain checks if the user or requester in the forwarding device has access to the system and whether the task which is requested is valid or not. This verification process by blockchain will ensure that even a valid user is blocked from performing a task that is not allowed to be done. Additionally, blockchain must ensure that no device should operate within its maximum capacity to ensure load balancing.

In the article [Faizullah et al. \(2020\)](#), BS-IOT integrates blockchain into SDN for secure and cooperative network management. The network is divided into sub-domains, each with controllers, switches, and smart devices, utilizing Blockchain Sharding. Leader controllers in each domain create blocks for sub-chains after validation, collectively forming the main blockchain for cooperative management. To address data-sharing concerns, the network employs sub-domains and sub-chains. The scheme comprises four components: Inter-Domain Network Management by using consortium blockchain with PBFT (Byzantine Fault Tolerant) for consensus to handle the network management including data sharing, routing, and resource allocation within a single domain, Intra-Domain Network Management to facilitate cross-domain communication, Node Management which addressing compromised or broken controllers and an Extensive Service Platform allowing flexible deployment of applications like intrusion detection and secure data sharing.

The article [L. Liu et al. \(2020\)](#), proposes BlockSDN, a blockchain-as-a-service framework for SDN, to address security issues. It adopts a permissioned blockchain

architecture and addresses two attack scenarios: a malware-compromised switch at the data plane and a distributed denial of service (DDoS) attack at the control plane. In response to these threats, the system employs a process that identifies and handles malicious incoming requests in a SDN environment. If the requests are determined to be from a single device (DoS attack), the originator node is traced, and if from multiple devices (DDoS), the controlling machine is identified. The malicious entities are added to a blacklist log, relieving other switches. Legitimate requests are sent to a consensus node for further processing and added to a synchronized block in a permissioned blockchain, ensuring all switches have access to the verified flow rules.

The authors of [Abou El Houda et al. \(2019\)](#), aimed to enhance the efficiency and security of Industrial IoT (IIoT) applications by integrating decentralized blockchain technology with Software Defined Networking (SDN). They proposed a comprehensive architecture consisting of three layers: perception, end, and cloud, to control the flow of data from IoT devices to cloud service providers (CSPs). To optimize energy consumption and cluster-head selection, a cluster-head selection (CHS) algorithm was implemented in the perception layer. Using an OpenFlow routing mechanism, a FloodLight-based controller was used in the SDN architecture to ensure valid data transmission to the control layer. Blockchain integration facilitated six types of transactions, including release, decrypt, consume, update, and revoke, enabling efficient data management while enhancing security and privacy. SDN controllers ensure data validity before securely storing it in the distributed ledger. Upon completion of data transactions through blockchain, cloud data was delivered to IIoT devices, creating a reliable and secure data flow management solution for Industrial IoT applications.

The article [J. Hu et al. \(2020\)](#), proposed a security solution for SDN supported by edge-blockchain, which interconnects IoT networks. Their approach introduced an Edge Computing-based Blockchain-as-a-Service (BaaS) solution, emphasizing efficient, edge-distributed blockchain for flow verification. By offering blockchain services to multiple users in a network with IoT devices, they merged SDN with edge computing to ensure secure communication. Entities like IoT devices, SDN switches, blockchain agents (BCAs), and edge nodes were part of the architecture. BCAs, situated at edge servers, verified flows using smart contracts on a permission-based consortium blockchain like Hyperledger. They devised reward schemes based

on contract theory to mitigate moral hazards from BCAs hosted by third-party edge providers, determining optimal block size, latency, and reward values.

In [A. Rahman et al. \(2020\)](#), Block-SDoTCloud integrated SDN and Blockchain for enhanced cloud storage security. This architecture comprises three layers: the Perception Layer, Infrastructure Layer/IoT Networks, and SDN-based Security on the Cloud. The Infrastructure Layer controls IoT forwarding devices and makes use of SDN gateways for safe data transmission, the Perception Layer collects data from actual sensors and devices in the real world. The SDN-based Security on the Cloud layer leverages SDN's capabilities in managing networking applications and employs the Block-SDoTCloud model to ensure data security. The OpenFlow protocol and TLS connectors are highlighted as being important tools for secure communication inside the SDN architecture.

In conclusion, there has been a lot of interest lately in the fusion of blockchain with Software-Defined Networking (SDN) technology for a range of network management and security applications. The reviewed research articles highlight a wide range of inventive methods and fixes aimed at improving data security, integrity, and effective administration across several domains, such as smart cities, IoT, industrial IoT, and UAV networks. Most of the Blockchain-based SDN UAV systems focus on data sharing security, collaboration between UAVs, topology information sharing, prospects of using 5G, for securing SDN networks from false flow rule injection, AI and deep learning-based schemes. When paired with the dynamic and programmable aspects of SDN, the intrinsic immutability and decentralized nature of blockchain provide strong foundations for safe communication, effective resource allocation, and resilient network management. But one inherent problem with the SDN architecture is Single point of failure (SPOF) of the controller and none of the proposed SDN UAV systems consider the SPOF issue of the SDN architecture. Thus Our proposed system introduces a Blockchain-based SDN controller recovery and forwarding device migration for fault tolerant SDN UAV communication. Blockchain technology can address these issues with its immutability and distributed nature to help the SDN controller recover itself from a SPOF.

2.3 Performance Metrics

Performance metrics are critical for analyzing a system's effectiveness and efficiency, making them an integral part of any study report. Measuring system performance allows researchers to identify strengths and weaknesses, bottlenecks, and areas for improvement. They provide a mechanism to evaluate the system's performance to that of similar systems. This study's comparative analysis includes the following parameters:

Security: The confidence that a system meets its requirements and is resilient against vulnerabilities and failures [Shukla et al. \(2022\)](#).

Interoperability: A measure of the ability to perform inter-operation between entities (software, processes, systems, business units) and facilitate communication, cooperation and coordination among these processes and units [Belchior et al. \(2021\)](#).

Immutability: Immutability is defined as immutable or also in the form of permanence. A value or a variable is immutable if it can not be changed in program execution. Blockchain is immutable because of the usage of cryptographic hash functions like SHA-256. In the blockchain, the output value, known as a hash, is used as a unique identifier for a block of data. The hash of each resulting block matches the hash of the previous block. The block hash is dependent on the data contained in the block, which means any changes to the data will require a change to the block data. Therefore, each block created on the hash is based on the data contained in that block and the hash of the previous block. These hash markers play an important role in ensuring the security and immutability of the blockchain [Rahardja et al. \(2021\)](#).

Decentralization: A network is decentralized if each transaction in the network is done only by two nodes at a time and does not need third-party validation. For blockchain, decentralization allows the blockchain to be non-reliant on a central authority. This enables nodes to essentially have equal voting rights within the network, which is then utilized with the consensus algorithm to dictate the blockchain [Zarrin et al. \(2021\)](#).

Scalability: Scalability is a system's elasticity or ability to grow. Poor scalability can result in poor system performance [Garcia et al. \(2008\)](#).

Redundancy: Redundancy is the intentional duplication of critical components

or functions of a system with the goal of increasing the reliability of the system [Randles et al. \(2011\)](#).

Fault Tolerance: Fault tolerance is the power of a system to preserve the execution of its expected functions despite any failure. It is associated with dependability, operational success and failure. A fault tolerance-based system should investigate the defects in individual software or hardware components and ensure the system will provide services in one or many failures of the infrastructural components [Bharany et al. \(2022\)](#).

Reliability: Reliability aims at analyzing and evaluating the ability of products and services to perform the functions that they are intended to provide by design [Coit & Zio \(2019\)](#).

Integrity: Integrity in the context of a system refers to the integrity of data. Data integrity is the maintenance and assurance of the accuracy, and consistency of data over its entire life cycle [B. Liu et al. \(2017\)](#).

Flexibility: Flexibility is the ability to maintain balance in the system, ensure continuity in unexpected situations and cope with uncertainty [Girela-López et al. \(2020\)](#).

Programmability: The capability of a device or a network to accept a new set of instructions that may alter the device or network's behaviour [A. Rahman et al. \(2020\)](#).

Accuracy: It means correctness. Accuracy is often evaluated based on how well the network adheres to its defined policies, efficiently manages resources, and responds to dynamic changes in network conditions [O'Neill \(2020\)](#).

Consistency: Consistency means that all network devices receive the same instructions and have the same understanding of the network's current state. If one device has a different view than another, it could lead to network misconfigurations, disruptions, or inefficient use of resources [Hosen et al. \(2020\)](#).

Robustness: If a blockchain is resilient, it can sustain the network and its nodes, running smoothly. All nodes should be running effectively and doing transactions without interruption in order to maintain the overall blockchain's health [Saleh et al. \(2020\)](#).

Authenticity: The capacity to verify requesters' identities before granting access to critical information. With blockchain authentication, the block's hash containing personal data must confirm a person's identity, such as a unique identification number [Shi et al. \(2020\)](#).

Availability: Enables blockchain nodes to check whether data for a proposed block is available without downloading the entire block. In other words, it is the assurance that a block applicant released all transaction data for a block and that the data is accessible to other network participants [Hepp et al. \(n.d.\)](#).

Off-chain: Off-chain transactions are those verified away from the central blockchain system, typically resulting in a cheaper and faster transaction for the user [Wan et al. \(2019\)](#).

Transaction latency: A blockchain performance metric evaluates the time frame of completing the transaction in a blockchain network and is measured in milliseconds (ms) or microseconds (μ s). The total time of adding a transaction in a block and blockchain by mining is Transaction Latency [Manuskin et al. \(2020\)](#).

Transaction throughput: A critical blockchain performance metric to measure the total successful transaction within a period of time that measures in transactions per second (TPS) or transactions per minute (TPM) unit. Other factors such as block size, block time, consensus mechanism, network latency, and scalability solution influence transaction throughput for a better outcome [Aoki et al. \(2019\)](#).

Block creation time: the average time of creating a block for a transaction in a blockchain is referred to as the block creation time. The block generation time may differ for its consensus mechanism and design choice. The shortest period of time is the best for the best performance. The unit of it is measured in s [Wan et al. \(2019\)](#).

2.4 Performance Analysis

The study of this section provides a clear sense of the advantages and disadvantages of existing work having specific parameters.

Considering the term of security, article [Kumar et al. \(2022\)](#), integrates Blockchain-enabled registration and verification for secure data sharing among UAV-SDN network entities and a deep learning-based flow analyzer for intrusion detection in the network. To ensure immutability, blockchain is used for storing UAV data. Controllers form blockchain as decentralized storage to reduce the dependency on third parties. The system uses a registration system, an additional layer of security which is a deep learning-based flow analyzer to detect the packet's legitimacy, making the system quite fault tolerant. It uses a registration system to validate connected UAVs and blockchain technology to store UAV information to ensure reliability and integrity. The system uses a packet legitimacy detection mechanism to provide accuracy of stored information in the blockchain. Considering consistency, UAVs need to be registered and their data goes through a flow analyzer before executing which increases the system's efficiency and robustness. The authenticity of the UAVs is achieved by using registration and packet filtering schemes and the addition of Blockchain technology increases the system security. To evaluate the system efficiency, Transaction latency, Transaction throughput, and Block creation time are measured. With these advantages, the system can also face some challenges for not considering some parameters like interoperability, scalability, redundancy, flexibility, programmability, availability and Off-chain. How the internal and external communication happens in the network is unclear as the system does not regard interoperability. There is no way to find out the scalability of the system for a large network. For not considering redundancy, there is no backup controller if any controller failure happens which also proves the system is not flexible enough. Also for not considering programmability, there is no way to understand the flexibility of the system. The way is not clear enough if the data is available for valid users or not. There is no way in the system to store additional information on UAVs in the off-chain.

In article [R. Gupta et al. \(2021\)](#), the network is secure from security attacks such as controller hijacking, and man-in-the-middle. Considering security, the system can detect eavesdroppers to protect the controller which ensures the consistency of the network. As 5G has been proposed, the number of transactions in the blockchain is greater than in traditional communication systems. All the stored UAV data are immutable as blockchain is used here. The data of the controllers is stored in blockchain, a decentralized database which reduces the use of third parties to control the network. K-means unsupervised clustering algorithm has been used to detect packet's legitimacy, making the system quite fault tolerant which creates a secure communication channel between UAVs and SDN controllers. The system uses a Kerberos-based registration system to validate connected UAVs and IoT devices and also has an eavesdropper detection mechanism to detect the packet's legitimacy to ensure reliability which makes the proposed network more secure than other existing networks. To maintain data integrity, before storing information in the blockchain, the UAVs need to be registered to the system and data packet information goes through a K-mean unsupervised clustering algorithm which creates a robust and secure network. To provide data accuracy, the proposed network uses blockchain technology which ensures the reliability of the UAV's data or information. The system is capable of providing consistency, as it can detect and protect the network from different types of network attacks. A registration system based on the Kerberos authentication protocol is used to ensure authenticity which protects the system from unauthentic or malicious UAV entries. On the other hand, for not considering interoperability, it is unclear how to communicate with UAVs among different networks. For not considering redundancy, the system is enabled to use less amount of storage. The system isn't flexible enough if a controller fails or SPOF occurs. The system is also not flexible enough to store additional data in the off-chain. As transaction latency, transaction throughput and block creation time are not considered as parameters there is no way to measure the efficiency or the performance of the network.

In the article [N. Hu et al. \(2021\)](#), considering programmability, The system takes advantage of hardware programming technologies in the data plane, which allows support for network upgrades and compatibility. Multiple ground stations form a consortium blockchain and provide consistent data view and schedule updates

for UAVs. Therefore, regardless of which controller the UAV binds, the same result will be produced. A public key encrypted communication is used between UAVs and controllers which ensures a secure communication. Blockchain provides immutability which protects the stored data from external or suspicious modification. The system is quite reliable as it is programmable from the application layer, and controllers are decentralized because of blockchain technology. The UAV information goes to the blockchain and they use a secure communication channel to communicate with the controllers, which provides data integrity. To ensure data accuracy, the system uses blockchain technology which makes the data immutable and reliable. For authenticity, the system utilizes public key cryptography to secure the communication between controllers and UAVs and avoid malicious control nodes. As the system doesn't have any kind of redundancy, it uses less storage. Although the system uses public key cryptography to secure the UAV communication channel with a controller, what will happen if a registered UAV starts acting maliciously while sending data packets? So, whether or not the system will be able to provide consistency is questionable which also proves the system is not fault-tolerant and flexible enough if a controller or RSU (Roadside Unit) fails or SPOF occurs. As the system does not consider interoperability and scalability, there is no clarification on how a network will behave with another network and large network. As the system does not consider block creation time, transaction throughput and transaction latency, there is no way to measure the performance of the system.

As the research article [Liao et al. \(2021\)](#), considers interoperability, the controller of the system can share domain service among them in a trustless environment. The multi-drone service sharing information gets stored in the blockchain, which is immutable and reliable. The controllers of the system create a consortium blockchain that is decentralized. They use a consensus mechanism called Proof of Security (PoSG) to validate the drone service sharing information in the blockchain without any need for a centralized authority. Because of programmability, the end users can obtain drone service from the application layer of the system without worrying about the underlying complexity of the system. This proposed system considers transaction latency, throughput and block creation time by which the system is

able to measure the performance of the network compared with the existing proposed systems. Although the system provides secure cross-domain service sharing between controllers, it doesn't propose any mechanism or registration system to secure the UAV communication with the controllers. So, whether the system is entirely secure, reliable and robust or not is debatable. As the article [Liao et al. \(2021\)](#), doesn't consider scalability, it is difficult to measure how the system will perform in a large network. The system uses less storage in the sense of redundancy. As the stored information in the blockchain goes through a rigorous consensus process, injecting false information is quite impossible. So the accuracy of the stored data in the chain is undeniable. The controllers utilize a Proof of Security Guarantee (PoSG) to validate the drone service sharing information, which provides sufficient security of stored data in the blockchain and ensures integrity. As it doesn't have any mechanism to prevent SPOF, it is debatable whether or not the system is flexible and fault-tolerant. The system will not be able to provide full consistency, as it is prone to network attacks from the lack of a registration system for the UAVs to establish a secure connection. At the same time, it does provide the consistency of drone service sharing information as it goes through a consensus process before storing it in the chain and all the controllers have a consistent view of the stored information. There is no way in the system to store additional information as it does not consider off-chain.

Considering security in article [Aldaej et al. \(2023\)](#), the system guarantees the secure exchange of data among various components of the UAV-SDN network through the implementation of Blockchain technology for registration and verification. Additionally, a flow analyzer based on deep learning is incorporated to identify any potential intrusions. Because of using blockchain technology, there is no need to depend on any centralized or third-party authority. The simulation results demonstrate that the proposed technique can successfully develop trust between numerous controllers and enable safe routing across different domains as the system considers integrity. The result of having robust security features in the system is high reliability and high performance to ensure efficient and secure network management. The proposed model achieved enhanced measures of accuracy ensuring routing dependability and managing global reputation in the blockchain.

The proposed model has achieved enhanced precision measures of in terms of accuracy. The results of the simulation show that the suggested method can effectively establish trust among multiple controllers and facilitate secure routing as proof of consistency. As the system does not consider authenticity and availability, there is a gap to provide a secure communication channel in the network. Comparing or measuring the performance of proposed systems with existing systems is quite difficult as there is no consideration of block creation time, transaction latency and throughput. Again there is no scope to store important additional information in the off-chain.

To ensure security in article [Derhab et al. \(2021\)](#), Blockchain-based communication is proposed between controllers to avoid network attacks and reputation value is maintained in the blockchain to determine a controller's reliability. By using Blockchain and reputation systems, controllers can reach consensus in the system and can keep it functional even if a controller fails. As the data stored in the blockchain, it is immutable and free from any kind of external modification. Considering decentralization, the controllers and the trusted nodes form a private blockchain. As a result, the system isn't controlled by a single entity. Each domain of data layer devices has more than one redundant controller as a backup in the case of a single-point error. As the controllers act as each other's redundant replacement, it solves the SPOF issue of SDN architecture. At the same time, the system also provides other features to make it more reliable. The system does provide data integrity as the data stored in the blockchain goes through the consensus of the blockchain nodes before storing the flow and topology information of the network. Also to prevent wrong information insertion, the controllers rely on the reputation values of each other to reach a consensus. Considering flexibility, the system can detect and prevent false or wrong information insertion also it can prevent SPOF. As the system can prevent wrong flow and topology information insertion, all the data layer devices such as switches have a consistent view of the flow rules and topology information, which helps to make accurate forwarding decisions. As a result of providing accuracy, the system can deal with modification based on conditions which also ensure the robustness of the system. To ensure authenticity the system stores data in the blockchain which makes the storage more secure. To analyze the performance of the proposed system transaction latency, throughput

and block creation time are considered. The system doesn't have interoperability as the system uses only one blockchain. It doesn't require interoperability because there is no need for cross-chain exchange of data thus this system is not appropriate for large networks and there is no consideration of scalability. In the system, there is no clear discussion regarding the programmability of the controller. As there is no consideration of off-chain, the system is not dependent on any external database.

In [Misra et al. \(2020\)](#), the system stores metadata of flow rules instead of storing the whole flow rule data in blockchain, which saves about 75 percent of space and helps the network to perform more efficiently and effectively. By considering fault tolerance, the controllers can recover themselves to a previous state by receiving flow information from switches for uninterrupted performance. As the metadata is stored in the blockchain, there is no change or external modification of the stored data. The system is flexible enough to recover itself from SPOF. Consider decentralization, the data layer devices such as switches act as blockchain nodes to reach consensus and store the metadata in the blockchain which helps the system not to depend on any third party or centralized authority. Authenticity of the valid flow rules while recovering is ensured by the usage of Blockchain technology for its immutable nature. On the other hand, although the system relies on blockchain nodes' consensus mechanism to store metadata in the chain, it doesn't consider how to avoid false flow rule injection or modification in the data layer. This means whether the system is entirely secure is questionable. There is no explanation in the system, of how the system will perform in the different networks. The system does address the SPOF issue. Still, it lacks any security measures to prevent false flow rule injection or modification in the data layer devices which proves that this system is not reliable enough. However the system does provide data integrity as the metadata stored in the blockchain goes through consensus of the blockchain nodes before storing, and the controller relies on these stored metadata to only insert authenticated flow rules in the flow table while recovering from SPOF. The system doesn't have any kind of redundancy as there is no backup storage in the system. The system hasn't considered consistency which makes it vulnerable to network attacks. To measure the performance of the system, there is no consideration of transaction latency, block creation time and throughput. As there is no

consideration of off-chain, the system is unable to store additional data.

The article [Boukria et al. \(2019\)](#), to ensure security, the system detects and prevents flow rule injection from malicious devices in SDN data layer elements. Blockchain provides controller authentication and integrity of the traffic flow circulated between the controllers and other network elements. Considering consistency, the system can prevent false flow rule injection; all the data layer devices such as switches have a consistent view of the flow rules, which helps to make accurate forwarding decisions. As the data stored in the blockchain, it is immutable and free from any kind of external modification. The controllers and the trusted nodes form a private blockchain. Here, the system isn't controlled by a single entity. Which means, the system is decentralized. Interoperability hasn't been considered as the system stores all the data in one blockchain and there isn't any need for cross-chain data exchange for this system. The system does provide data integrity as the data stored in the blockchain goes through consensus of the blockchain nodes before storing, and also the nodes check the authenticity of the flow rules stored rules in the chain against the rules in a switch before executing it. The system does provide accuracy, as it can deal with modification in the system's condition (prevent flow rule injection) which also ensures robustness. As the flow rules need to wait for the approval of the VMFm node before executing them, the system can ensure authenticity. The system provides block creation time which helps to measure the performance of the system. There is no idea regarding how the system will perform in a large network or with lots of devices as there is no consideration of scalability. Although the system can detect and prevent false flow rule injection, it isn't fault tolerant. If a controller somehow fails, it will affect the data layer devices under it. Although the system addresses some of the underlying issues of SDN architecture, it still isn't completely reliable. The flexibility of the system is debatable as the system can detect and prevent false flow rule injection, but there isn't any mention of what will happen if a single point of error occurs in the controller. The article didn't mention anything about the programmability of their system thus there is no way to get clear information the controller controls the whole system. As there is no consideration regarding the availability, it is difficult to ensure that valid users are able to get information.

In the article [Zeng et al. \(2022\)](#), the controllers and the trusted nodes form a private blockchain. Here, the system isn't controlled by a single entity. As the abstract topology information stored in the blockchain goes through the consensus of the blockchain nodes, the system does provide data integrity. To secure from any kind of modification in the system, the topology information is stored in Blockchain. To provide routing reliability, the controllers maintain global and local reputation values to reach consensus. The system can handle cross-domain packet forwarding which means the system ensures interoperability. It can also validate newly added nodes and maintain the reputation of each controller to provide shared data accuracy between them. Authenticity of the sharing of abstract topology information is ensured by the usage of Blockchain technology for its immutable nature. Although the system relies on blockchain nodes' consensus mechanism to store abstract topology information and reputation values in the chain, it doesn't consider how to avoid false flow rule injection or modification in the data layer. Which means the system isn't entirely secure. As there is no consideration regarding scalability, there is no idea how the system will perform in large networks. As there is no redundancy, the system uses less storage. The system is not fault-tolerant and flexible, as the authors didn't provide any mechanism that would allow the controller to retrieve itself from SPOF. The system is not consistent as it hasn't considered data layer security which makes it vulnerable to network attacks. In the system, there is no need to communicate with other networks thus interoperability is not considered. As there is no consideration regarding transaction latency, throughput and block creation time, measuring the performance of the system is quite difficult.

Considering security in [A. Rahman et al. \(2021\)](#), the system is able to dump flow rules stored in switches periodically, block the communication and isolate the rouge switches from the network if an anomaly is found. The controllers act as nodes and form a blockchain. Here, the system isn't controlled by a single entity. As the flow rules are stored in the chain, any controller can recover to its previous state by fetching information from the chain, which means, the controllers can withstand SPOF which makes it fault tolerant. As the controllers of the system store their flow rules in the blockchain, thus the stored information is safe from any external modification. Because of programmability, the end users can modify the controller policies from the application layer if needed. Considering reliability, the system can withstand

SPOF and the controllers can detect compromised switches also. The system does provide data integrity as the flow rule information stored in the blockchain goes through the consensus of the blockchain nodes. By ensuring flexibility, the system can prevent network attack and modification and copy with SPOF. The system does provide data accuracy as it uses blockchain technology and also it has a mechanism to detect compromised switches. To ensure consistency of rules among controllers, all the controllers maintain the flow rules in a Blockchain. Authenticity of the flow rules has been considered by introducing Blockchain-enabled data layers which secure the stored information. As transaction latency, block creation time and transaction throughput are not considered it is difficult to compare the performance of the proposed system with existing systems.

To ensure security in [Mendiboure et al. \(2020\)](#), authentication and access control have been implemented which ensure the interaction between different SDN network devices in a secure manner. To solve the scalability issue, the system is broken down into several local sub-systems to form a global Blockchain network which enables communication among the domains as well as ensures interoperability. Because of interoperability, the system consists of multiple chains that exchange information with each other. Vehicle certificate, controller and RSU information go to the blockchain, which makes the stored information immutable to any kind of modification. The system is broken down into different local blockchain networks and together they form a global network. No centralized authority is required in it. Considering reliability, the system is scalable and can avoid network attacks. The system does provide data integrity as the certificates, controller and RSU information goes to the blockchain, making it immutable, decentralized and accessible to every node of the network. To ensure data accuracy, it uses blockchain technology and also has authentication and access control mechanisms which makes the whole system quite secure and ensures the availability of data to valid users. As controllers have a consistent data view of the whole network which helps to handle the network more efficiently. The authenticity of the devices has been ensured by storing the X.509 certificates in the Blockchain as immutable blocks. Measuring the performance system is easier as it considers transaction latency, throughput and block creation time as its parameters. As there is no redundancy in the system, the system is robust. But if a controller or RSU fails or SPOF occurs, the system isn't

fault tolerant. Although the system can handle vehicles crossing domains, it doesn't mention what will happen if SPOF occurs in one of the controllers, thus there is a lack in terms of system flexibility. As there is no consideration regarding off-line, thus the system is not dependent on external databases to manage their data.

The paper [Khalid et al. \(2023\)](#) presents a robust solution for managing access control policies for IoT devices, addressing several key parameters. In terms of security, the system ensures authentication and access control, guaranteeing that only authorized users can access IoT devices, thereby mitigating security risks. The use of blockchain-based smart contracts establishes immutability, ensuring that access control policies are tamper-proof and reliable over time. Additionally, the system focuses on scalability, essential for automated IoT authentication systems, allowing the solution to accommodate a growing number of devices and platform heterogeneity efficiently. Furthermore, by leveraging SDN for network programmability, the system optimizes network traffic efficiency, enhancing overall performance. The evaluation of transaction throughput provides insights into the system's ability to process transactions efficiently, crucial for real-time IoT applications. However, several parameters are not considered in the paper. Interoperability and flexibility are essential for seamless integration and adaptation, while reliability and availability ensure consistent and accessible operation. While Decentralization and fault tolerance enhance resilience. Authenticity and integrity safeguard data trustworthiness. this approach may lead to a limited perspective, incomplete analysis, and reduced system robustness.

The article [Lokesh & Rajagopalan \(2020\)](#) that network principles are programmed as controller applications. The programmable control plane allows enforcing a sophisticated security policy that can be easily customized for securing the network. The system focuses on the security of controllers. To ensure consistency, the authors proposed a security model based on Blockchain technology among instances of the SDN controller. The system ensures the integrity of the data by encryption when the data packet passes from the forwarding device to the controller which creates secure communication between the control and data plane. To ensure system reliability, every controller in the network will have to validate their state among themselves before they commit any transaction. The applications at a controller instance will

validate their operations among themselves. In a cluster, devices have to validate their transactions among themselves. Controllers will have to digitally sign their corresponding CBLOCK to establish authenticity before trying to add the block to the CBC(Controller Blockchain) which ensures secure reliability of data. As there is no consideration regarding interoperability, thus there is no proof of how it will communicate in different networks. The authors do not describe how the system will perform on a large-scale network. In the system, there is no backup controller or device. Thus redundancy is not considered which ensures the system uses less space. As the proposed system is not considered fault tolerant, if any device or controller faces failure, there is no way to overcome the failure which means the system is not robust and flexible enough to overcome all types of faults. It does not consider transaction latency, transaction throughput and block creation time, thus it is difficult to measure the performance of the system. There is no need for additional storage to store the data thus off-chain is not considered here.

In article [Faizullah et al. \(2020\)](#), to ensure security, every transaction is added to the blockchain after verifying a request which protects the network from unauthorized entry. Only permissioned users, devices and operations can perform in the network which will protect the network from unauthorized access to ensure reliability. Blockchain technology provides a working proof of decentralized trust in the system which secures the system from external or malicious modification. In order to achieve scalability authors choose OpenFlow software switch and OpenVSwitch. They compare their result with more systems and the result shows that their system is more scalable than others. The system provides the comparative bandwidth utilization of permissions, public and OpenFlow SDNs. The bandwidth is almost half when the packet arrival rate reaches 1000 packets per second. When authors triple the number of packets/s, the whole network starts coming down and malfunctioning But when using the proposed architecture, the bandwidth starts at 2.1 Gb/s and stays stable even when traffic is tripled from 1000 packets/s. Thus prove the proposed system works more accurately or efficiently. The integrity of the flow table is ensured by checking flow tables with other controllers and switches which ensures the reliability of the flow rules. Software-defined networking (SDN) is revolutionizing the network infrastructure with a new paradigm. SDN splits the control/routing logic from the data transfer/forwarding enabling comprehensive

network programmability and improving load management. Different levels and each type of user are authorized to access/request/change specific tasks/policies. To ensure the access granularity for each device, each task will be first validated from the blockchain. As there is no consideration of interoperability, thus this system is appropriate for diverse networks. There is no clear explanation in the system if the single point of controller occurs to overcome the situation thus the system is not completely fault-tolerant, flexible and robust. The performance of the system can be compared with existing systems as it does not consider block creation time, transaction throughput and latency as their parameters.

The article [L. Liu et al. \(2020\)](#), uses SDN for large-scale networks for its feature of programmability. Three techniques are used to secure the network by managing the node and one of them is able to detect abnormal behavior of the network. Sharding is used to ensure the scalability of the network. Sharding divides the whole network into smaller, more manageable segments. This sharding technique makes the system more efficient by increasing the performance of the network. Controllers of the network build blockchain which ensures the decentralized management of the network and reduces the use of external databases or dependency on third parties. To prevent system failure, a backup controller is used if a leader controller fails or is compromised in every domain to make the system fault tolerant but it increases the redundancy of the system and this a one kind of lack of the system. By considering interoperability, controllers in a domain communicate with each other. and leader controllers communicate in a cross-domain network. To ensure reliability, before adding any controller to the network verification process is conducted which makes the system robust. To consider consistency, a backup controller is used which increases the redundancy. Blockchain sharding technique is used to overcome the poor throughput of blockchain which automatically decreases transaction latency and block creation time and thus the performance of the system will increase. As blockchain is used thus storage data is safe from external modification. At the same time, there is no way to store additional data as off-chain is not considered here.

The article [Abou El Houda et al. \(2019\)](#) provides significant advantages in enhancing the security, reliability, flexibility, and robustness of the Cochain-SC system. The implementation of an OpenFlow-based firewall ensures the security of

blockchain nodes by effectively monitoring and controlling network traffic, thus safeguarding the integrity of the blockchain network. Decentralization facilitates secure collaboration among multiple SDN-based domains, enabling the efficient transfer of attack information to mitigate DDoS attacks. Reliability is ensured through the use of blockchain technology, which guarantees the accuracy and immutability of recorded data. Additionally, the integration of an intra-domain DDoS mitigation strategy using machine learning enhances the accuracy of attack detection models. The system's flexibility is enhanced by providing mechanisms to easily add or remove collaborators, allowing organizations to join or leave the system seamlessly. However, lacking authenticity considerations, there's a risk of unauthorized access or manipulation of data, compromising the trustworthiness of the system. The absence of my reduced overall system efficiency. Without optimizing these parameters, the system may struggle to handle a large number of transactions simultaneously, leading to bottleneck and delays.

In article [Asaithambi et al. \(2022\)](#), ensures security through a unique controller, validating data before securely storing it in the blockchain ledger, thus safeguarding against unauthorized access. Scalability is achieved by designing the SDN framework with multiple homogeneous controllers, enhancing the adaptability of the IIoT environment. Confidentiality is maintained by integrating blockchain with a distributed ledger, enabling efficient data management through various transaction types. Immutability ensures that validated data or transactions remain unchanged once stored in the blockchain ledger. Decentralization is enabled as controllers collectively form the blockchain, facilitating decentralized network management. Reliability is bolstered by a robust transaction validation system, ensuring system integrity. Accuracy is upheld through blockchain technology and meticulous data verification processes. Furthermore, blockchain preserves the authenticity of IoT device identities, while the distributed ledger guarantees data consistency across the SDN controller network, ensuring reliable and precise operation. On the other hand, The system's lack of attention to interoperability leaves the method of internal and external communication within the network unclear. Additionally, the system's inflexibility could limit its ability to adapt to changing requirements or environments. Without redundancy strategies, the risk of data loss or system failure due to component malfunctions is increased. The system's vulnerability to disruptions or

failures is heightened due to the lack of focus on robustness. Neglecting off-chain capabilities may limit options for transaction management and blockchain overhead reduction. Failure to address transaction latency, throughput, and block creation time can impact system performance and responsiveness. The absence of fault tolerance measures increases the system's susceptibility to single points of failure or controller malfunctions. Lastly, the lack of information on programmability restricts the system's adaptability and extensibility to meet various application requirements or configurations.

In [Abou El Houda et al. \(2019\)](#), Utilizing Blockchain Contract Agents (BCAs) ensures authorized data access, enhancing security. Multiple controllers are employed for scalability without altering the architecture, offering flexibility and growth potential. The system leverages blockchain for immutability, recording flow rules to prevent tampering. Decentralization is achieved through a permission-based consortium blockchain, enhancing network resilience. Data integrity is maintained through robust flow rule verification processes. Byzantine consensus ensures reliability, fostering trust among nodes for dependable transactions. Blockchain technology guarantees authenticity during recovery, preventing unauthorized alterations. While not explicitly stated, the system demonstrates robustness, complementing its overall effectiveness. Its absence of discussion on interoperability raises concerns regarding its ability to seamlessly integrate with external systems and networks, potentially hampering collaboration and usability. Furthermore, the system's lack of flexibility restricts its ability to efficiently store additional data off-chain. The system's lack of redundancy planning exposes it to disruptions from component failures, risking downtime and data loss without proper backups. Moreover, its apparent absence of fault tolerance discussions casts doubt on its resilience, especially concerning single points of failure. The failure to address transaction latency, throughput, and block creation time may result in inefficient transaction processing, congestion, and delays during peak usage periods, affecting user experience. Additionally, the lack of mention regarding programmability indicates a missed opportunity to highlight the system's capabilities for developers to create and deploy custom applications or smart contracts. Finally, the absence of discussions on consistency raises concerns about the system's ability to maintain data integrity and synchronization across distributed nodes, undermining trust in its reliability.

and accuracy.

In article [A. Rahman et al. \(2020\)](#), considering security, controllers are able to effectively handle the security risk, new threats, and different kinds of attacks. As data is stored in the blockchain, it is secure from any external modification. As the controllers form the blockchain, there is no need to depend on any third party or centralised authority. This system ensures data integrity by flow rule verification process which ensures only authorized data access in the network which also ensures the accessibility of the system. This system is able to avoid network threats or attacks which makes the system reliable and robust. Because of programmability, this system offers better management of the network compared with traditional networking features. As systems use transaction throughput as a parameter, it will be helpful to measure the performance of the system. The system does provide data accuracy as it uses blockchain technology. In the Blockchain environment, each block can be accurately composed of multiple transactions. There are also some lacking in the system. For example, it is difficult to measure the performance of the system for large and diverse networks as it does not consider scalability and interoperability as its parameters. There is no clear description regarding how the system authenticates the devices. In the system, it is not possible to store additional information as it is not considered off-chain. The system isn't fault tolerant because the authors didn't talk about what happens if a controller fails or if there's a Single Point of Failure (SPOF) which means this system is not completely flexible enough to face failure. But it does not consider redundancy, it means it uses less storage.

Table 2.1: Comparison of Literature Review

	Security	Immutability	Interoperability	Decentralization	Scalability	Redundancy	Fault Tolerance	Reliability	Integrity	Flexibility	Programmability	Accuracy	Consistency	Robustness	Authenticity	Availability	Off-chain	Transaction latency	Transaction throughput	Block creation time
Boukria et al. (2019)	Y	Y	N	Y	N	N	N	N	Y	N	Y	Y	Y	Y	Y	N	N	N	Y	Y
Derhab et al. (2021)	Y	Y	N	Y	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	N	Y	Y
Misra et al. (2020)	N	Y	N	Y	Y	N	Y	N	N	Y	N	N	N	Y	Y	N	N	N	N	N
Zeng et al. (2022)	N	Y	N	Y	N	N	N	Y	Y	N	N	Y	N	Y	Y	N	N	N	N	N
A. Rahman et al. (2021)	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	N	N	Y	Y	Y
Mendiboure et al. (2020)	Y	Y	Y	Y	Y	N	N	Y	Y	N	N	Y	Y	Y	Y	N	N	Y	Y	Y
Kumar et al. (2022)	Y	Y	N	Y	N	N	Y	Y	Y	N	N	Y	Y	Y	Y	N	N	Y	Y	Y
R. Gupta et al. (2021)	Y	Y	N	Y	N	N	N	Y	Y	N	N	Y	Y	Y	Y	N	N	N	N	N
N. Hu et al. (2021)	Y	Y	N	Y	N	N	N	Y	Y	N	Y	Y	N	Y	Y	Y	N	N	N	N
Liao et al. (2021)	N	Y	Y	Y	N	N	N	N	N	N	Y	N	N	Y	N	Y	N	Y	Y	Y
Faizullah et al. (2020)	Y	N	N	Y	Y	Y	N	Y	Y	N	Y	Y	N	N	Y	N	N	N	N	N
L. Liu et al. (2020)	Y	N	N	Y	Y	Y	N	Y	N	N	Y	N	Y	N	Y	N	N	Y	Y	Y
Lokesh & Rajagopalan (2020)	Y	Y	N	Y	N	N	N	Y	Y	N	Y	N	Y	N	Y	N	N	N	N	N
Asaithambi et al. (2022)	Y	Y	N	Y	Y	N	N	Y	N	N	N	Y	Y	N	Y	N	N	N	N	N
A. Rahman et al. (2020)	Y	Y	N	Y	Y	N	N	Y	Y	N	Y	N	N	Y	Y	N	N	N	N	N
J. Hu et al. (2020)	Y	Y	N	Y	N	N	N	Y	Y	N	Y	Y	N	Y	N	N	N	N	Y	N
Aldaej et al. (2023)	Y	Y	N	Y	N	N	N	Y	Y	N	N	Y	Y	Y	N	N	N	N	N	N
Abou El Houda et al. (2019)	Y	N	N	Y	N	N	N	Y	N	N	N	Y	N	Y	N	N	N	N	N	Y
Khalid et al. (2023)	Y	Y	N	Y	Y	N	N	Y	Y	N	Y	Y	N	Y	N	N	N	N	Y	N

From the above **Table 2.1**, it is found that existing systems discussed several blockchain-based security solutions for network controllers that efficiently handle security threats, new vulnerabilities and different kinds of attacks. By process like flow rule verification, the systems guarantee data integrity and authorised access. Nonetheless, the systems are not without limits, including difficulties with device authentication, scalability, compatibility, fault tolerance and redundancy which are raised questions about how flexible and effective these systems would be in vast, heterogeneous networks.

Chapter 3

Methodology

Methodology involves the systematic and theoretical examination of strategies and approaches within a particular field of study. It scrutinizes the set of strategies and guidelines that guide how knowledge is acquired and applied. A methodology does not aim to produce specific outcomes. Instead, it offers theoretical underpinning for knowing which systems, styles, or practices can be used in a particular situation to achieve a desired conclusion. In software analysis and design, methodology plays a crucial role in bridging the gap between requirement specifications and implementation. It serves as an intermediate stage where human-readable requirements are transformed into executable code. Systems analysis can encompass tasks such as modifying computer code to fulfill a specific function, fixing a malfunctioning air conditioning system, or optimizing daily routines to prevent errors. For this system, we are utilizing a waterfall methodology that follows a linear and sequential approach.

3.1 Introduction

The methodology for implementing the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs integrates blockchain technology with Software-Defined Networking (SDN) to enhance fault tolerance and security. The SDN controller self-recovery scheme stores unique flow rule fields immutably on the blockchain, allowing for rule retrieval and comparison in case of controller failure. Blockchain nodes are used by multi-controller topologies to verify and append flow rules, guaranteeing transparency. For forwarding device rule migration, a trusted authority updates device IDs in the blockchain, facilitating

seamless transitions during maintenance or replacement. Smart contracts enable the retrieval of flow rules, ensuring uninterrupted UAV communication and improved Quality of Service (QoS).

3.2 System Development Model

3.2.1 Waterfall Model

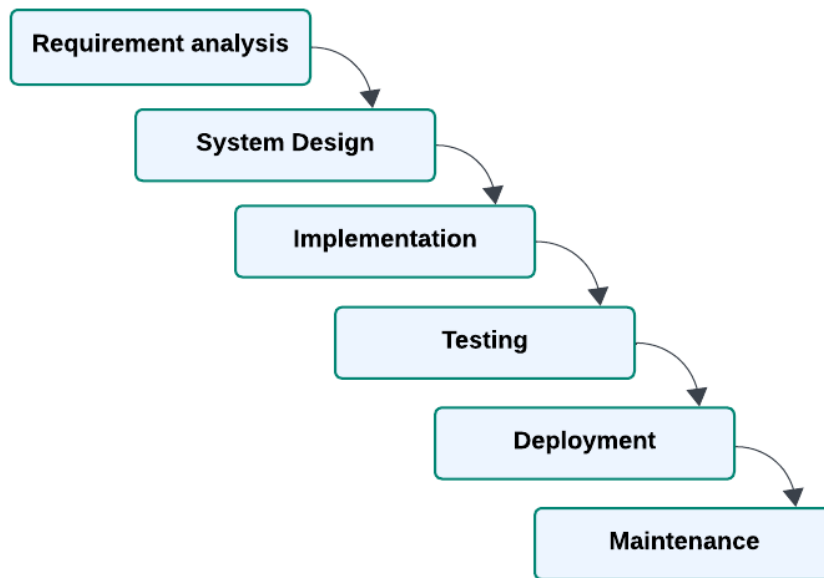


Figure 3.1: Waterfall Model

The waterfall methodology, a linear and sequential approach, is applied to develop the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs. This methodology involves distinct phases, each completed before moving to the next, ensuring a structured and thorough development process. In the requirement analysis phase, system needs for SDN controller recovery and flow rule migration are gathered and documented, including blockchain integration for fault tolerance and security. Based on these requirements, a comprehensive system design is created. This includes the architecture for blockchain integration with SDN, how unique flow rule fields will be stored and retrieved, and

the process for migrating flow rules between forwarding devices. During implementation, the system design is translated into code. Smart contracts are developed to manage flow rule storage and retrieval on the blockchain, the ONOS controller is configured in Mininet to store flow rule fields, and the flow rule migration mechanism is implemented with oversight by a trusted authority. The integrated components are thoroughly tested to ensure functionality and compliance with requirements. The controller recovery mechanism and flow rule migration system are tested for failure recovery and seamless rule transfer, respectively. Upon successful testing, the system is deployed in the UAV communication network, including setting up blockchain nodes, deploying smart contracts, and configuring SDN controllers and forwarding devices. In the maintenance phase, the system is monitored and updated as needed, addressing any issues and ensuring continuous operation and security. By following the waterfall methodology, the development of the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs is conducted in a structured and sequential manner, ensuring each phase is completed and validated before proceeding to the next. This approach provides a clear roadmap for the development process, ensuring thorough documentation, design, implementation, and testing to achieve a robust and reliable system.

3.3 System Analysis

System analysis is an analytical technique used to describe a system's workings and details to stakeholders or customers. It includes a description of the system's objectives, features, deployment procedure, and any planned modifications or improvements down the road. In order to facilitate effective communication and decision-making during the development and maintenance phases, system analysis aims to provide a comprehensive understanding of the features, processes, and capacities of the system.

3.3.1 System Design

System design is the process of planning and arranging how a system will operate. It comprises answering any questions and allaying any concerns users may have about the appearance and functionality of the system. Prior to a system being deployed, system design is a crucial stage in the system development process. It involves the initial work of outlining the system's components, their relationships, and its guiding principles of functioning. System design has a major influence on the entire development process since it lays the foundation for an effective system implementation.

3.3.2 Implementation

The research paper's implementation phase focuses on implementing the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs. This section covers the development techniques, technologies, and technical details of the system. It looks at the key components of smart contract implementation, blockchain integration with SDN and user interface development, among others and describes the steps used to turn the requirements into a workable solution. Throughout the implementation phase, the frameworks, tools, and construction processes of the system are highlighted.

3.4 Requirement Engineering

Requirement engineering for the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs entails a thorough process of gathering, analyzing, documenting, and validating requirements. Functional and non-functional requirements, covering aspects such as fault tolerance, security, scalability, and performance, are meticulously recorded. Throughout development, rigorous methods manage requirements to ensure alignment with evolving stakeholder needs and system objectives, addressing challenges like centralized controller vulnerabilities and malicious UAVs, ultimately ensuring uninterrupted communication and security within the SDN environment.

3.5 Requirement Engineering Process

The requirement engineering process overview outlines the systematic steps involved in identifying, documenting, and validating the requirements for the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs system. It discusses essential phases like feasibility study, requirement collection, software requirement specification, and validation. The passage delves into the strategies employed to ensure the accuracy and comprehensiveness of the requirements, highlighting challenges and considerations encountered at each stage.

3.5.1 Feasibility Study

Evaluating the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs system's practicality and potential success is the main goal of the feasibility study. It involves assessing technical aspects like whether blockchain can integrate smoothly with SDN and UAV systems. The study that was carried out to evaluate the technical, operational, and financial viability is included in this subsection. technical feasibility, where the capabilities of implementing blockchain technology within SDN environments and its compatibility with UAV communication systems are evaluated. Economically, it examines the costs versus the benefits to ensure financial viability. Operationally, it checks if the proposed solutions align with existing infrastructure and operational needs without disruption. The goal of this study is to offer a thorough understanding of the project's viability, assisting in the decision-making process for its implementation.

3.5.2 Requirement gathering

The process of gathering requirements for the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs entails identifying stakeholders and learning about their needs and objectives. This is achieved through interviews, surveys, and workshops to capture functional and non-functional requirements, including system features, performance criteria, security measures, and regulatory compliance. The collected requirements serve the foundation for

the design and development of the system, guaranteeing the cooperation of all stakeholders and a successful implementation.

3.5.3 Software Requirement Specification

The Software Requirement Specification (SRS) for the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs outlines both functional and non-functional requirements crucial for system development. The behavior of the system, including features like controller recovery, flow rule migration, and blockchain integration, is described in depth by the functional requirements. Non-functional requirements include features like security, performance, and others that guarantee the scalability and reliability of the system. The usage of use case diagrams, entity-relationship diagrams, and user stories, among other standard documentation techniques, to effectively documenting requirements are covered in this article.

3.5.4 Software Requirement Validation

Validating software requirements means confirming that the documented specifications are accurate, consistent, and comprehensive. This topic covers the techniques used to review, validate, and confirm the requirements. To identify and fix any issues or gaps in the specifications, it emphasizes the importance of involving stakeholders and experts.

3.6 Software Requirement for Development

The software requirements listed below are necessary for developing the proposed Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs. First and foremost, it is essential to have a robust blockchain platform that supports smart contracts and offers safe, decentralized ledger features. To maintain integrity and transparency when adding new flow rules, the system should work with a variety of consensus techniques. The software must also make it easier for SDN controllers to design a self-recovery mechanism that will allow them to safely store and retrieve flow rule data failure. Moreover, software components for authorization and authentication are required in order to coordinate

flow rule migration between forwarding devices via a trusted authority system. Comprehensive testing tools and frameworks should be employed to validate the system's performance, security, and resilience under various scenarios, ensuring its effectiveness in providing fault tolerance and uninterrupted communication for UAVs.

3.6.1 Microsoft Visual Studio Code

Microsoft Visual Studio Code (VS Code) is a robust and adaptable source-code editor that is used for developing Blockchain-based controller recovery and SDN forwarding device rule migration systems for UAVs. It provides customisable features and productivity-boosting extensions, and it supports a multitude of programming languages. VS Code guarantees accessibility and flexibility with cross-platform compatibility [Dongre et al. \(2020\)](#). Because of its cross-platform interoperability and lightweight design, Visual Studio Code is a preferred option for developers working with many operating systems. Microsoft Visual Studio Code provides developers with the tools they need to write, debug, and deploy code effectively, regardless of the complexity of the project from basic scripts to intricate applications.

3.6.2 Remix IDE

Remix IDE is a Strong and open-source integrated development environment (IDE) that is made especially for ethereum smart contract development with solidity language. It is crucial for developing blockchain-based controller recovery and SDN forwarding device rule migration systems for UAVs. One of the most popular tools for creating, evaluating, and debugging smart contracts is Remix IDE [Narváez et al. \(2022\)](#). A code editor, a console for interaction with the ethereum blockchain, and an extensive collection of plugins that increase its functionality, such as integrated testing environments and static analysis tools—are all part of its intuitive interface. In the Ethereum development community, Remix IDE is essential for promoting quick prototyping and creativity since it facilitates both local and remote development, making it accessible for developers at various levels of expertise.

3.6.3 Truffle

Truffle is a widely-used Ethereum development framework that aids in constructing decentralized applications (dApps), including blockchain-based controller recovery and SDN forwarding device rule migration systems for UAVs. It makes development, deployment, and compilation of smart contracts easier. Moreover, it has a built-in testing framework that facilitates automated testing and allows for comprehensive smart contract validation. With its powerful capabilities and Ethereum integration, truffle speeds up the creation of dApps [Natarajan et al. \(2017\)](#). Interaction with deployed smart contracts is facilitated by the JavaScript-based API. Developers can obtain useful information from Truffle’s documentation and community assistance.

3.6.4 Ganache CLI

For testing and development purposes, Ganache CLI provides a fast and customizable blockchain simulation, making it a vital tool for Ethereum developers that is also beneficial for our proposed system. Ganache is a Command Line Interface (CLI) tool that is used to create a local blockchain that has all the functionality of the main Ethereum network [Mann et al. \(2020\)](#). For our system, we created a personal blockchain on our PC using Ganache CLI. Ganache CLI offers features such as quick mining of transactions, robust logging, and the ability to modify blockchain parameters, which helps in debugging and optimizing smart contracts before deployment to a live network. It is a vital part of the Truffle Suite, enabling productive and successful development workflows for decentralized apps thanks to its intuitive interface and adaptability.

3.6.5 NodeJS

The proposed Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs requires Node.js. An open-source server-side JavaScript runtime environment is called Node.js. Because of its significant performance and scalability, it can handle multiple tasks at once and analyze massive amounts of data. As we use Truffle and Ganache for our system, they depend on Node.js for their operation because they are JavaScript-based applications. Integrating Truffle and Ganache into the system necessitates Node.js, further emphasizing

its pivotal role in facilitating the development and deployment of innovative solutions like the one proposed. Utilizing the power of JavaScript throughout the application stack, Node.js is a stable and effective platform with a vibrant community and a wealth of resources [Gourisetti et al. \(2019\)](#).

3.6.6 Mininet

Mininet is indispensable for the proposed blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs as it provides a crucial platform for simulating and testing network environments. Mininet, an open-source network emulator, allows the development of software-defined networks and applications to be tested, learned, and deployed as sophisticated virtual networks on a single Linux kernel [Dholakiya et al. \(2021\)](#). Its capability to emulate various network topologies, including SDN controllers, forwarding devices, and UAVs, allows researchers to assess the effectiveness of the proposed schemes in a controlled setting before real-world deployment. Leveraging its Python-based API, users can dynamically create, modify, and analyze network configurations, making it a flexible platform for investigating various networking scenarios. Overall, Mininet serves as a vital tool for validating and refining the proposed system's functionality, ensuring its readiness for practical implementation in UAV networks.

3.6.7 Metamask

MetaMask is a browser extension that functions as a digital wallet for cryptocurrency management and communication with decentralized apps (dApps) created on blockchain platforms like Ethereum. It is crucial for our proposed system due to its secure authentication management, transaction authorization, and smart contract interaction capabilities. Metamask provides a simple user interface and facilitates the maintenance of Ethereum accounts. The smooth interaction with web applications is made possible via its JavaScript library. Metamask is a useful tool for utilizing blockchain technology because of its documentation and developer community [Hasan et al. \(2020\)](#).

3.6.8 Hyperledger Besu

Hyperledger Besu is essential for the proposed system's success as it offers a private blockchain network environment, ensuring only authorized entities participate, thereby safeguarding the integrity of the SDN controller recovery and flow rule migration processes for UAVs. Leveraging smart contracts, Besu minimizes mistake risk and improves operational efficiency by enabling the automation of tasks related to controller recovery and rule migration. Its features include support for privacy, confidentiality, and customizable consensus mechanisms, enabling organizations to implement secure and efficient blockchain applications. Enterprises increasingly prefer Hyperledger Besu for its capacity to construct networks supporting private transaction processing, integration with public blockchains such as ethereum, and maintaining architectural flexibility with high transaction throughput [Uddin et al. \(2021\)](#).

3.6.9 Docker

Docker is a platform that enables developers to develop, distribute, and run applications in lightweight, portable containers. It is beneficial for developing blockchain-based controller recovery and SDN forwarding device rule migration systems for UAVs. Developers can create, deploy, and maintain applications with ease when using Docker since containers operate consistently and reliably on any machine that has Docker installed. Docker is used in our system to run Besu. By simplifying the software development and deployment process, this method improves team productivity on a variety of projects by optimizing workflows. Docker can expedite the testing and production deployment of code [Preeth et al. \(2015\)](#).

3.6.10 Web3.py

Web3.py is essential to this system as it provides the necessary tools for interacting with the Ethereum blockchain, which serves as the foundation for implementing the proposed solutions. Web3.py is a python library that enables developers to integrate blockchain functionality into their web applications, allowing users to interact with smart contracts, read/write data to the blockchain, and safely manage digital assets. The web3 library facilitates interaction with smart contracts on the ethereum blockchain by providing programmatic access through the given ABI [PrasadRao et al. \(n.d.\)](#). Web3.py is now the de facto standard for ethereum development, enabling programmers to create reliable and intuitive decentralized applications. It does this by providing extensive documentation and vibrant community support.

3.6.11 Lucid Chart

Lucidchart is a powerful web-based diagramming tool that is used for the development of Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs. It enables users to create, share, and work together on a variety of visual representations, including professional diagrams, organizational charts, flowcharts, and network diagrams. Customizable templates and simple drag-and-drop capability are features of Lucidchart. The construction of diagrams is made easier with Lucidchart [Kosba et al. \(2016\)](#). The tool may be accessed from any device due to its cloud-based nature, and its multiple export options make it easy to integrate it into project materials.

3.6.12 Mendeley

Mendeley is an extensive reference management and academic social networking tool that is beneficial for our system. It is designed to help researchers finding ,sharing , and organizing research papers. Mendeley makes access to academic resources, collaboration, and reference management easier [Foytik et al. \(2020\)](#). Mendeley correctly incorporates references and citations. It has a cross-platform interface that is easy to use. Using mendeley, users may organize their research process by building a personal library of research materials that is available from any device and taking advantage of its robust search and labeling features.

3.6.13 LaTeX

LaTeX is a tool used for writing documents that look professional, particularly in technical and scientific writing, that is beneficial for our system. LaTeX ensures consistent and polished output by letting users concentrate on content while the system takes care of the layout. LaTeX improves project documentation's uniformity and professionalism [Hellman \(2002\)](#). Mathematical formulas are handled using its typesetting skills. Packages for citations, references, and bibliographies are available in LaTeX. Collaboration is facilitated by integration with version control. Documents with cross-referencing characteristics are well-structured in LaTeX.

3.7 Required Programming Languages

To develop the blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs requires expertise in Solidity, JavaScript, Python. On the ethereum blockchain, solidity is required for the implementation of smart contracts. JavaScript is essential for deploying smart contracts via Truffle in this system and It's used to write scripts that compile Solidity contracts, migrate them to the blockchain, and interact with deployed contracts. Python is necessary for Mininet, as it configures network topologies, sets up virtual network elements, and automates network behavior for testing and simulation.

3.7.1 Solidity

Solidity is essential for the Blockchain-based controller recovery and SDN forwarding device rule migration system for UAVs. Solidity is a high-level, statically typed programming language supporting inheritance, polymorphism, libraries, and complex user-defined types, making it an effective tool for developing sophisticated smart contracts [Wohrer & Zdun \(2018\)](#). Solidity enables developers to design self-executing contracts with the rules of the agreement encoded directly into code, making it easier to create decentralized applications (dApps). It enables the development of smart contracts to securely manage and automate the storage, validation, and migration of SDN flow rules within the blockchain and developers can use it to design transparent, tamper-proof smart contracts for our proposed system. With features like inheritance and interfaces, Solidity makes contracts safe and effective.

3.7.2 JavaScript

JavaScript is crucial for our proposed system. JavaScript is a high-level, interpreted language renowned for its dynamic and versatile capabilities. It is used for various purposes for example to write scripts that compile Solidity contracts, move them on the blockchain, and interact with deployed contracts. It is also necessary for the deployment of smart contracts via Truffle in our system. JS offers dynamic interfaces, easy integration, and access to tools and libraries with wide resources and community support [Kosba et al. \(2016\)](#). JavaScript enables efficient backend communication through asynchronous programming. Its extensive ecosystem provides libraries and modules designed for specific functionalities.

3.7.3 Python

Python, a dynamic and versatile programming language used for many different tasks. For the proposed system mininet requires python since it automates network behavior for testing and simulation, configures network topologies, and builds up virtual network pieces. The backend of Mininet, based on Python coding, enables the creation of custom network topologies, nodes such as switches, controllers, and hosts, by executing commands and displaying necessary results through a Python script [Bholebawa et al. \(2016\)](#). Python is significant because of its readability and simplicity, which enable both beginners and experienced programmers to utilize it effectively. Its active community continually adds new features and improvements, making Python a reliable and powerful tool for solving a wide range of problems.

3.8 Summary

In this chapter, we have discussed the project's development approach and also specified the tools and language integral to the project's implementation.

Chapter 4

Design and Development

4.1 Project Management and Financial Activities

Project managers can supervise the development, deployment, and maintenance of the system by effectively managing resources, stakeholders, and timelines of the project. At the same time, the financial activities of the project involve budgeting, funding and financial monitoring to ensure longer term viability of the project. This section deals with the intersection of project management and financial activities within a blockchain UAV communication system using SDN, looking at their significance, challenges as well as possible solutions to ensure that such initiatives are successful.

4.1.1 Work Breakdown Structure

A Work Breakdown Structure (WBS) is a project management technique that divides a project into smaller, more manageable parts. The overall scope of the project is organized and defined by these parts, which are usually represented in a hierarchical form. An increasingly detailed definition of the project work is given at each sub level of the WBS. By using WBS, project managers can effectively plan, organize and manage projects by determining the necessary activities, dependencies and resources required at each level of this structure.

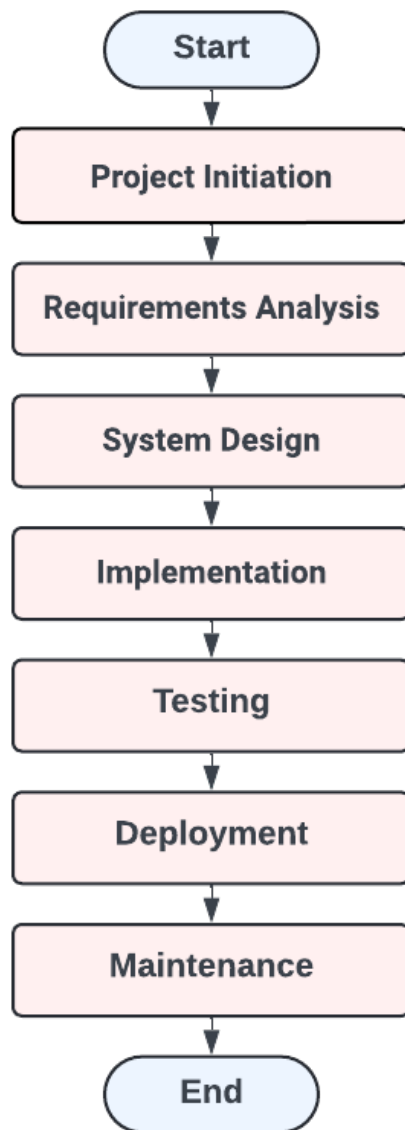


Figure 4.1: Work Breakdown Structure Flowchart

Figure 4.1 presents the WBS of our proposed scheme. It plays an important role to provide a roadmap for project implementation, management and enabling effective coordination of activities.

4.1.2 Estimate Costs

To calculate the costs related to each of the work packages within the WBS, both direct and indirect costs need to be considered. An example of estimated costs for each work package taking into account different cost factors is given below:

Project Planning and Initiation:

- Define Project Scope: No direct costs.
- Identify Stakeholders: No direct costs.
- Create Project Charter: No direct costs.

Requirements Analysis

- Identify Requirements: No direct costs.
- Requirements Documentation: No direct costs.
- Conduct Market Research: Cost of market research reports or consulting fees.
- Review Requirements with Stakeholders: Consulting fees.
- Requirements Specification: No direct costs.

System Design:

- Develop System Architecture: Cost of design tools and UI/UX resources.
- Create Detailed Design Specifications: Cost of design tools.
- Review Design with Stakeholders: Consulting fees.
- Finalize Design Documents: No direct cost.

Implementation or Coding:

- Set Up Development Environment: Cost of hardware, software licenses, and cloud services.
- Develop Smart Contract: Developer salaries or consulting fees.
- Conduct Code Reviews: No direct cost.

Testing:

- Prepare Test Plan: No direct cost.
- Unit Testing: Testing resources' salaries or consulting fees.
- Integration Testing: Testing resources' salaries or consulting fees.
- System Testing: Testing resources' salaries or consulting fees.
- User Acceptance Testing: Testing resources' salaries or consulting fees.

Deployment:

- Create Deployment Plan: No direct cost.
- Deploy System to Production: Deployment resources' salaries or consulting fees.
- Conduct Post-Deployment Verification: Deployment resources' salaries or consulting fees.

Maintenance:

- Monitor System Performance: Maintenance resources' salaries or consulting fees.
- Provide Technical Support: Support resources' salaries or consulting fees.
- Implement Bug Fixes and Patches: Development resources' salaries or consulting fees.
- Perform System Enhancements: Development resources' salaries or consulting fees.

4.1.3 Budget Plan

It is essential to create a project budget that includes all necessary costs and materials for the execution of projects, together with contingency funding. Any organization can improve the management of their financial resources, plan for additional needs and achieve more effectively their objectives through the development of an integrated budget.

Table 4.1: Cost Breakdown

Cost Type	Description	Cost (in taka.)
Hardware	Networking devices	2,000
	Computer 1	82,000
	Computer 2	91,000
	Computer 3	62,000
	Total cost	2,37,000
Developer	Salary of 2 developers in 4 month	2,72,000
Software	Working tools	10,000
Maintenance	Maintenance cost(per month)	5,000
	Internet bill(per month)	800
Field Testing		40,000
Logistic Supports	Stationary	50,000
Total Project Budget		6,14,800

Based on our analysis, **Table 4.1** provides an estimate of the costs for making this proposed system. The cost of making the project may vary depending on other factors.

4.1.4 Funding Plan

A funding plan, ensuring that the project has the necessary resources to achieve its objectives while maintaining financial health and sustainability, is essential for the effective management of the financial aspects of the project. The funds will be raised through a combination of government grants, angel investing, and crowdfunding.

4.1.4.1 Government Grants

The availability of government grants or funding sources to support technology firms may vary depending on the nation in which the firm is located. The group would have to fulfill certain requirements, such as having a solid business plan and a clear route to revenue, in order to be eligible for government grant.

4.1.4.2 Investor

Securing funding from venture capitalists and angel investors who are interested in technology and UAV developments. Partnerships with companies in the UAV, blockchain and SDN sector that are willing to invest in this project for potential future benefits can also be a way of obtaining funding.

4.1.4.3 Crowdfunding

Funding can be increased through the launch of a crowdfunding campaign targeted at tech enthusiasts, UAV hobbyists and other members of the public who are interested in supporting innovative projects. The team would have to develop an interesting video and presentation, set real financial targets as well as attractive rewards for backers to launch a successful crowdfunding campaign.

4.2 System Design and Description

The proposed system is an efficient scheme for SDN controllers to recover from SPOF using blockchain technology. Additionally, a novel mechanism to migrate flow rules to new forwarding devices has been introduced.

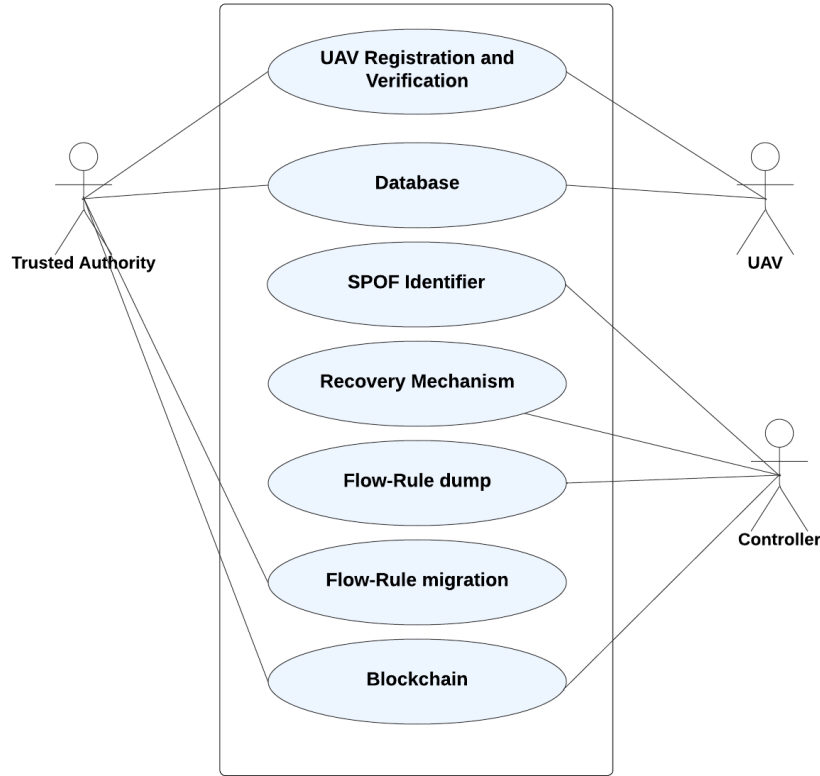


Figure 4.2: Use case diagram of the proposed system

Figure 4.2 shows a use case representation of the proposed system. The Controller, UAVs, and the trusted authority are the main actors. The trusted authority is responsible for managing UAV registration and verification, database management, SPOF identification and blockchain operations. The UAV interacts with the UAV registration and verification process, as well as the database.

4.2.1 SDN Controller self-recovery scheme

In the proposed scheme, the SDN controller leverages its forwarding devices and blockchain technology to recover from a single point of failure (SPOF), thereby

ensuring fault tolerance and uninterrupted UAV communication service. The controller identifies unique fields from each new flow rule and securely records this data on the blockchain. In case of a failure, the controller retrieves the flow rule information from the blockchain and compares it with the existing rules in the forwarding devices to restore its previous state.

Here are some of the fields that each flow rule may contain:

flowId: A unique identifier for the flow rule.

deviceId: Specifies the identifier of the forwarding devices to which the rule is applied.

ETH_TYPE: Specifies the Ethernet type criterion.

IPV4_SRC: IPv4 source address of a host.

IPV4_DST: IPv4 destination address of a host.

ETH_SRC: Ethernet source address criterion, matches packets based on source MAC address at the data link layer.

ETH_DST: Ethernet destination address criterion, matches packets based on destination MAC address at the data link layer.

Priority: Priority of the flow rule, higher value indicates higher priority.

IP_PROTO: Defines the protocol type: TCP or UDP.

Whenever a new flow rule is generated, the controller extracts the unique fields that remain unchanged during the lifecycle of each flow rule, and sends them to create blocks in the blockchain. Controllers in SDN use OpenFlow protocol to communicate with its forwarding devices and application layer. To check whether all the forwarding devices within the SDN are in operation, controllers ping their forwarding devices periodically to ensure reachability.

Algorithm 1 Controller SPOF Detection

Input: Controller C at stable S_t , set of forwarding devices S .

Output: Controller C recovered from S_f to S_t , set of forwarding devices S .

```
for  $S_i \in S$  do
    controller sends periodic ICMP packets to ping forwarding devices
    if  $ping.response == Normal$  then
        |  $S'_t = S_t$  create flow rule blocks in Blockchain
    else
        |  $S'_t = S_f$  Initiate Blockchain-based recovery repeat
        | until  $S'_t = S_t$ ;
    end
    controller  $C$  returns to stable state  $S_t$ 
end
```

The process of SPOF detection has been given in **Algorithm 1** . If SPOF has been detected, then the recovery process gets triggered autonomously.

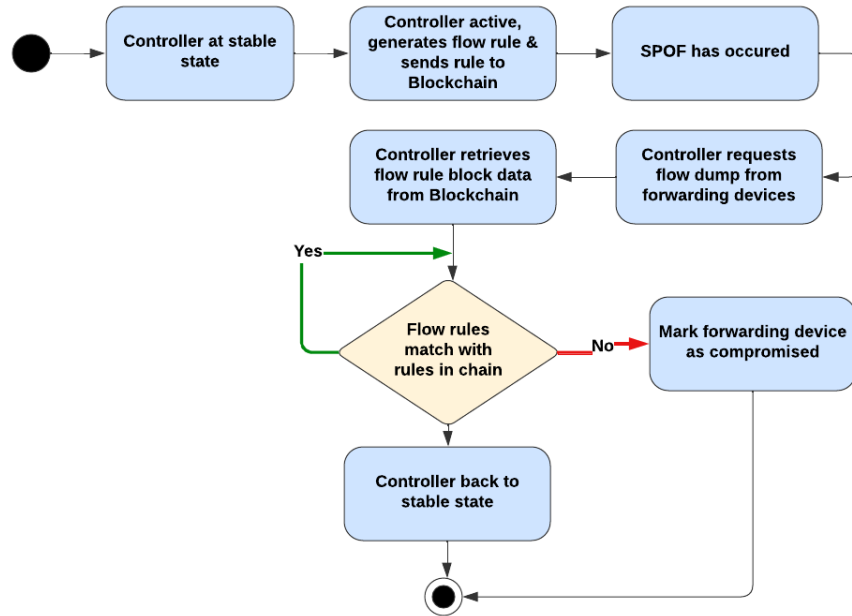


Figure 4.3: Activity Diagram of Controller Recovery

Figure 4.3 shows the activity diagram of the recovery process. While the controller at stable state, it generates blocks from flow rules and sends them to the

blockchain network. In the event of SPOF, the controller requests for flow dumps from all its adjacent forwarding devices and also fetches the block data. It then cross-references each rule from the chain with the stored rules in the forwarding devices. If a match is found, the data is reintegrated into the controller's operation. After the comparison, the controller goes back to its stable state.

4.2.2 Blockchain-based flow rule migration for SDN forwarding devices

Forwarding devices are the hardware components in SDN, and store flow rules created by a controller. These flow rules tell the forwarding devices how to route data packets within the network. As SDN breaks the vertical integration, the hardware and software components are separated from each other. So, the forwarding devices can't decide the routing behavior, instead, this responsibility goes to the controller. When a new data packet arrives, the controller creates a new rule for it and sends the rule to be stored in a forwarding device. Next time, whenever a similar packet arrives, the forwarding device will recognize it and forward the packet immediately based on the predefined rule generated by the controller. Any hardware is prone to failure and over time may need replacement or servicing. As the proposed system stores the generated flow rules in blockchain, forwarding device's limited lifespan raises a problem. The flow rules stored in any particular forwarding device contains the device id of the forwarding device, which is a unique field of a flow rule. As the proposed system stores this information in blocks, if a forwarding device containing those flow rules were changed will no longer remain comparable since the previous device id stored in blockchain will not match with the new device id. To solve this problem, the proposed system has a mechanism to handle flow rule migration from the previous forwarding device to a new forwarding device.

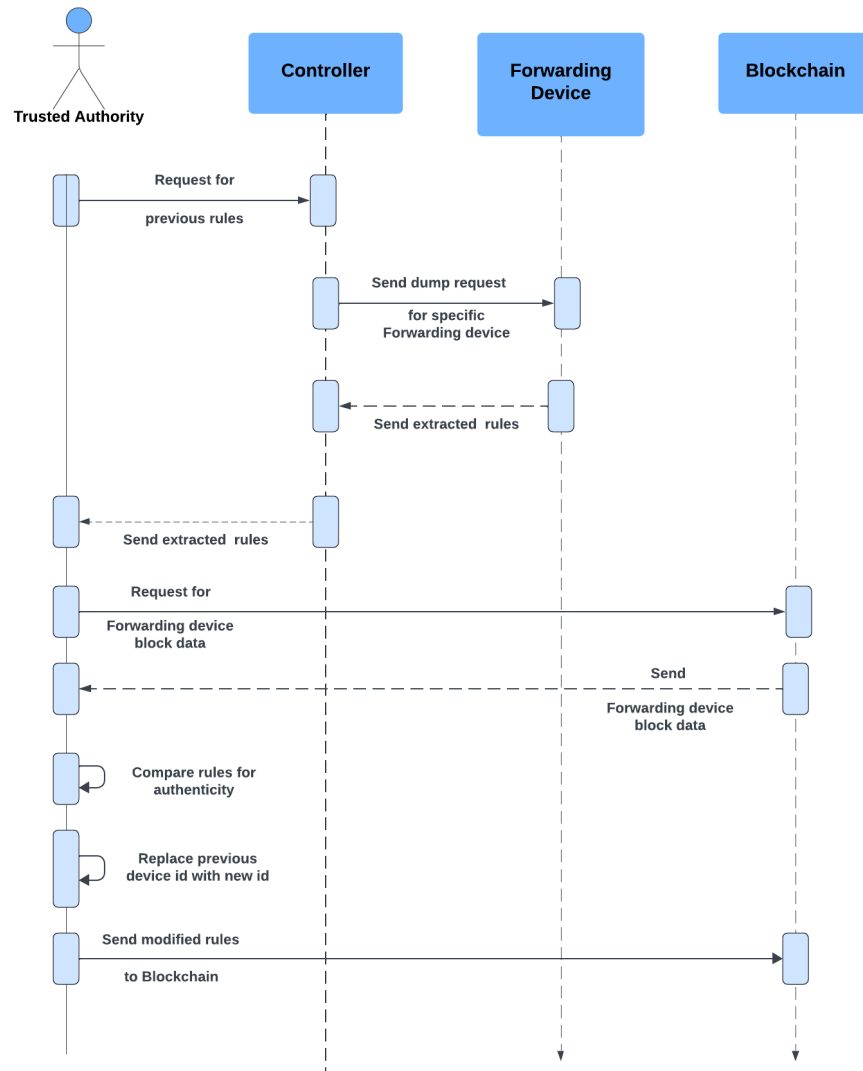


Figure 4.4: Sequence diagram for flow rule migration for SDN forwarding devices

Figure 4.4 is a sequence diagram representation of the flow rule migration mechanism for forwarding devices. Only the trusted authority will handle the migration process, and have the access to execute the migration function written in smart contract. First, the trusted authority will send a request to the SDN controller for the flow rule dumps from the forwarding device that needs flow rule migration. The controller will dump the rules from that particular forwarding device and send the extracted rules' information back to the trusted authority. The trusted authority will next perform an authentication process before updating the flow rule information. A request for flow rule block information for the forwarding device will be sent to blockchain. Then a comparison will be performed to ensure all the rules' authenticity against block data. If the rules pass the authentication step, all the flow rules' device id field will be updated with the new forwarding device id. After successful replacement, the migration process will be complete and the modified flow rules will go to blockchain to reflect the changes made.

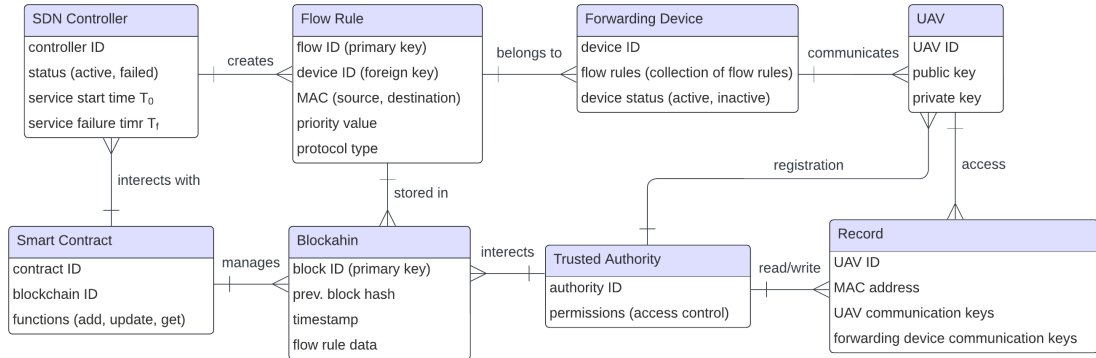


Figure 4.5: ER Diagram of the proposed system

Figure 4.5 is the the ER diagram that illustrates a system combining SDN, blockchain, and UAVs. The SDN Controller creates flow rules for Forwarding Devices, which then communicate with UAVs. Flow Rules are stored in the blockchain, managed by Smart Contracts. UAVs register with a Trusted Authority that controls access to Records, which contain UAV ids, model, owner address and communication keys. The blockchain interacts with both the Trusted Authority and the controller, ensuring secure and efficient data management. This setup enhances communication and control within the network, leveraging blockchain's security and transparency.

4.3 Implementation

This section goes through the implementation process of the proposed system. It delves into the deployment of the smart contract onto two different networks and the ensuing interaction with them. Hyperledger Besu was selected as the private network and Ganache to simulate public blockchain-like performance.

4.3.1 Smart Contract

Solidity was selected as the smart contract programming language to ensure that the code remains the same in both public and private blockchain after deployment. The smart contract uses Solidity version 0.8.0. SDN controllers and the trusted authority can use the deployed smart contract to interact and change the state of the blockchain.

Algorithm 2 SUAVNet Smart Contract

Input: Flow rule fields

Output: Flow rule information

Function:

addFlowRule(flow rule fields)

Create a new flow rule object with the provided details;

Store the flow rule object in the “blockData” mapping using the content identifier as the key;

Function:

getBlockData(Content Identifier)

end

Retrieve the flow rule object from the “blockData” mapping using the content identifier;
return the details of the flow rule object;

Algorithm 2 shows the function to create a new rule block from a flow rule. Whenever an SDN controller creates a new rule, it extracts the unique fields and creates blocks using them.

Algorithm 3 Migrate Data From Previous Device

Input: Device ID

Output: null

Function:

updateDeviceForPreviousDevice;

takes **previous Device ID** & **new Device ID** as parameters

if *stored device ID* == *device ID(parameter)* **then**

 | assign New Device value to Previous Device value

else

 | return false;

end

event emit message "Device Updated!";

Function:

getFlowRulesByDeviceID;

takes **Device ID** as parameters

if *stored device ID* == *device ID(parameter)* **then**

 | populate **result** array;

else

 | return NULL;

end

return result;

Algorithm 3 shows the function responsible for flow rule migration among forwarding devices. Only the trusted authority has the access and rights to execute this function. Whenever flow migration is necessary, the trusted authority will call the `updateDeviceForPreviousDevice()` function and pass the previous forwarding device ID and new forwarding device id as arguments. The smart contract will then fetch the rule block data for the previous forwarding device and update them with new forwarding device id. The block data can be fetched using the `getFlowRulesByDeviceId()` function. It takes the forwarding device id as parameter and returns the corresponding flow rule information from blockchain.

4.3.2 Deployment of Smart Contract in Hyperledger Besu

Hyperledger is an Ethereum client tailored to suit the needs of enterprise. It can cater to both public and private permissioned network scenarios, featuring an adaptable EVM implementation. It is versatile, capable of running on test networks like Sepolia and Görli. Moreover, it boasts a range of consensus algorithms, including Proof of Stake, Proof of Work, and Proof of Authority (IBFT 2.0, QBFT, and Clique). Its elaborate permissioning schemes are meticulously crafted for deployment within consortium environments.

The selected consensus for the private Besu network is QBFT (Quorum Byzantine Fault Tolerance), which is an enterprise-grade Ethereum client developed under the Linux Foundation's Hyperledger project. It is designed to ensure consensus among a network of nodes, even in the presence of Byzantine faults, where nodes may act maliciously or fail unpredictably.

In QBFT, network participants (nodes) take turns proposing blocks of transactions, and other nodes vote on the validity of these proposed blocks. Consensus is reached through a process of repeated rounds of proposing and voting. This mechanism allows QBFT to tolerate up to one-third of the network's participants acting in a Byzantine manner while still maintaining agreement on the state of the blockchain.

The private network consists of four validator nodes. Only the validators participate in the consensus process.

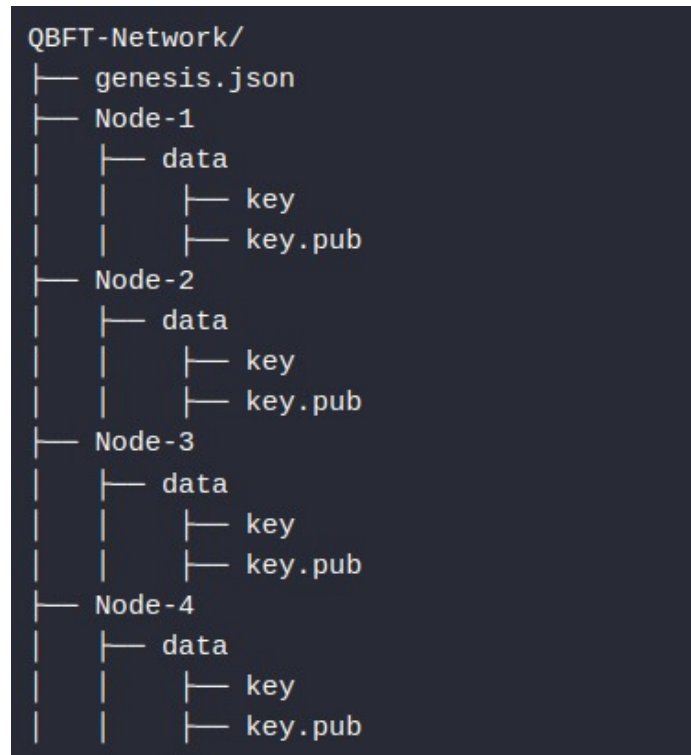


Figure 4.6: Besu Project Tree Structure

Figure 4.6 shows the folder tree view of the project. Each validator node has its own key. The genesis.json file contains information about all the validators, how much time the consensus should take, gas limit, difficulty etc.

```
[
  {
    "contractName": "SDNRecovery",
    "abi": [
      {
        "inputs": [],
        "stateMutability": "nonpayable",
        "type": "constructor"
      },
      {
        "anonymous": false,
        "inputs": [
          {
            "indexed": true,
            "internalType": "uint256",
            "name": "flowId",
            "type": "uint256"
          },
          {
            "indexed": false,
            "internalType": "string",
            "name": "newDevice_Id",
            "type": "string"
          }
        ],
        "name": "DeviceUpdated",
        "type": "event"
      },
      {
        "inputs": [],
        "name": "flowRuleCount",
        "outputs": [
          {
            "internalType": "uint256",
            "name": "",
            "type": "uint256"
          }
        ]
      }
    ]
  }
]
```

Figure 4.7: Generated ABI from Smart Contract Compilation

After setting up the private network, a truffle project containing the smart contract is placed in the root folder. The solidity compiler compiles the code into bytecode and generates ABI, which will be deployed to the blockchain network. **Figure 4.7** shows the generated ABI for the smart contract.

```
const SDNRecovery = artifacts.require("SDNRecovery");

module.exports = function (deployer) {
  deployer.deploy(SDNRecovery);
}
```

Figure 4.8: Smart Contract Deployer

Figure 4.8 shows the migration script. It takes the smart contract artifacts as input. A function is exported, accepting the truffle 'deployer' object as its argument. This function defines the deployment steps, typically utilizing the 'deployer.deploy()' method to deploy contracts along with any required constructor arguments or initialization parameters. Additionally, the migration script can manage contract dependencies, ensuring that dependencies are deployed prior to dependent contracts.

```
Compiling your contracts...
=====
> Everything is up to date, there is nothing to compile.

Starting migrations...
=====
> Network name: 'besu'
> Network id: 1337
> Block gas limit: 4700000 (0x47b760)

1_deploy_contracts.js
=====

Replacing 'SDNRecovery'
-----
> transaction hash: 0xf350d6b8ec4fc5802dbdc31f2a705bb4f59c3459ba02bbe8d1048d34b3004ccb
> Blocks: 0
> contract address: 0xCaB7530B10318dEcc2915aca9e22E3C85987e5B2
> block number: 5546
> block timestamp: 1716921195
> account: 0x627306090abaB3A6e1400e9345bC60c78a8BEf57
> balance: 89999.999999531444516
> gas used: 1345569 (0x148821)
> gas price: 0.000001 gwei
> value sent: 0 ETH
> total cost: 0.00000001345569 ETH

> Saving artifacts
-----
> Total cost: 0.00000001345569 ETH

Summary
=====
> Total deployments: 1
> Final cost: 0.00000001345569 ETH
```

Figure 4.9: Successful Deployment of the developed Smart Contract

After compiling the smart contract, the `truffle-config.js` file is set accordingly to connect to the Besu network. It takes network specifications for Besu, such as network ID, host, port etc. The contract is then deployed to the Besu network. **Figure 4.9** shows the successful deployment of the smart contract.

[illegible]

Figure 4.10: Transaction Record After Creating a Rule

Figure 4.10 shows a successful transaction while adding a new flow rule using the truffle console. The appropriate parameters are passed to the `addFlowRule()` function and a new block gets appended to the chain.

```
truffle(besu)> SDN.getFlowRulesByDeviceId('switch1')
[
  [
    '1',
    'switch1',
    '00-B0-D0-63-C2-26',
    '00-B0-D0-63-C2-26',
    '100',
    '6',
    flowId: '1',
    deviceId: 'switch1',
    MACSrc: '00-B0-D0-63-C2-26',
    MAC4Dst: '00-B0-D0-63-C2-26',
    priority: '100',
    ipProto: '6'
  ]
]
```

Figure 4.11: Fetch Rule Information

Figure 4.11 shows the `getFlowRulesByDeviceId()` function fetching the previously created rule's block information by passing SDN forwarding device ID.

4.3.3 Ganache Blockchain Integration

The smart contract was also deployed in Ganache to simulate testnet-like performance. To achieve similar performance of a testnet, difficulty and block creation time has been set accordingly. The truffle project setup for Ganache is similar to Besu network's setup.

```
(7) 0x2cfc7d69f8d555fc4bb11b83a6ae231d1ce65f032802915f2739e127949f9ef4
(8) 0x4bbb88e61f8563fdf434406b461b07053537461a2d636e08b944b48a661c1f09
(9) 0x0c70107889c2c854f079ec2f4ebd7ed2c370ea2cc9763b78e6162625d62f0034

HD Wallet
=====
Mnemonic:      recall amused spatial tragic expire obtain net sick busy steel ag
ree person
Base HD Path:  m/44'/60'/0'/0/{account_index}

Gas Price
=====
20000000000

Gas Limit
=====
6721975

Call Gas Limit
=====
9007199254740991

Listening on 127.0.0.1:8545
█
```

Figure 4.12: Ganache-CLI running listening to a specific port

Figure 4.12 shows the ganache-cli running and listening to a specific port. After launching the Ganache-CLI, again, similar to Besu, network specifications such as network ID, host, port etc are given in truffle-config.js to successfully connect and deploy smart contracts on Ganache.

4.3.4 Mininet Integration

This section delves into the integration of Mininet with blockchain. Mininet simulates the SDN environment and can create virtual network components like: hosts, switches, SDN controllers etc.

```
cse@cse-Z490-UD-AC:~$ /opt/onos/bin/onos -l onos
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!

      _____
     /         \
    /   ONOS   \
   /             \
  /_____/_____ \
 /               \
/                 \
\                 /
 \_____/_____ /
  \           /
   \         /
    \       /
     \_____/

Documentation: wiki.onosproject.org
Tutorials:    tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.
```

onos@root >

Figure 4.13: Onos service running

For the proposed system, Onos has been selected as the controller for Mininet. **Figure 4.13** shows the Onos service up and running.



Figure 4.14: Onos service user login

Figure 4.14 shows the UI of the ONOS service for a user to login.

```
cse@cse-Z490-UD-AC:~/truffle_public$ sudo python3 mininet_script.py
[sudo] password for cse:
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> 
```

Figure 4.15: Mininet topology running

After login, a script written in Python is executed to create a Mininet network topology instance that has the Onos as the controller, one forwarding device, and two hosts. After generating the topology, a flow rule is created to demonstrate the setup is working as intended. **Figure 4.15** shows the successful launch of the Mininet terminal.

```

flow_rule = {
  "flowId": "51791398821646033",
  "deviceId": "of:0000000000000001",
  "priority": 40000,
  "isPermanent": True,
  "selector": {
    "criteria": [
      {"type": "ETH_TYPE", "ethType": "0x0800"},
      {"type": "IPV4_SRC", "ip": "10.0.0.1/32"},
      {"type": "IPV4_DST", "ip": "10.0.0.2/32"},
      {"type": "ETH_SRC", "mac": "00:00:00:00:00:01"},
      {"type": "ETH_DST", "mac": "00:00:00:00:00:02"},
      {"type": "IP_PROTO", "protocol": "6"}
    ]
  },
  "treatment": {
    "instructions": [
      {"type": "OUTPUT", "port": "2"}
    ]
  }
}

```

Figure 4.16: Flow rule structure

Figure 4.16 shows the flow rule that was installed.


```
truffle(development)> const t = await SDNRecovery.deployed()
undefined
truffle(development)> t.flowRules("51791398821646033")
Result {
  '0': '51791398821646033',
  '1': 'of:0000000000000001',
  '2': '00:00:00:00:00:01',
  '3': '00:00:00:00:00:02',
  '4': '40000',
  '5': '6',
  flowId: '51791398821646033',
  deviceId: 'of:0000000000000001',
  MACSrc: '00:00:00:00:00:01',
  MACDst: '00:00:00:00:00:02',
  priority: '40000',
  ipProto: '6'
}
truffle(development)> 
```

Figure 4.19: Fetched flow rule information from blockchain

Figure 4.19 shows the information of the flow rule fetched from the blockchain.

```

{
  "flowId": 1,
  "deviceId": "switch1",
  "ETH_TYPE": "IPv4",
  "IPV4_SRC": "192.168.0.1",
  "IPV4_DST": "192.168.0.2",
  "ETH_SRC": "A:BB:CC:DD:EE:FF",
  "ETH_DST": "FF:EE:DD:CC:BB:AA",
  "Priority": 100,
  "IP_PROTO": 6
},
{
  "flowId": 2,
  "deviceId": "switch1",
  "ETH_TYPE": "IPv4",
  "IPV4_SRC": "192.168.1.1",
  "IPV4_DST": "192.168.1.2",
  "ETH_SRC": "11:22:33:44:55:66",
  "ETH_DST": "66:55:44:33:22:11",
  "Priority": 100,
  "IP_PROTO": 6
},
{
  "flowId": 3,
  "deviceId": "switch1",
  "ETH_TYPE": "IPv4",
  "IPV4_SRC": "192.168.2.1",
  "IPV4_DST": "192.168.2.2",
  "ETH_SRC": "AA:AA:AA:AA:AA:AA",
  "ETH_DST": "BB:BB:BB:BB:BB:BB",
  "Priority": 100,
  "IP_PROTO": 6
},
{
  "flowId": 4,

```

Figure 4.20: Flow Rules dumped from forwarding devices

Suppose a single point of failure has occurred, the controller will dump flow rules from all the forwarding devices. **Figure 4.20** shows the flow rules as JSON format.

```
cse@cse-Z490-UD-AC:~/compare$ node blockDataCompare.js
Rule 1 is not authentic.
switch1 is compromised.

Rule 6 is not authentic.
switch2 is compromised.

Rule 9 is not authentic.
switch2 is compromised.

All rules checked.
cse@cse-Z490-UD-AC:~/compare$
```

Figure 4.21: Comparison between flow rule from forwarding devices and block data

A script has been written in JavaScript to iterate through all the rules to detect anomalies. If found, it prints the compromised rule and device in the console. **Figure 4.21** shows the output after the script has completed its operation. The compromised forwarding devices will be then left for further inspection by the trusted authority.

4.4 UAV Registration User Interface (UI)



Figure 4.22: Homepage

Figure 4.22 demonstrates the homepage for the system. This page gives a brief description to the viewer into what the system has to offer. Furthermore, it redirects the UAV user to the Login and Register page.

Register Your UAV
Home Login Register Devices Logs

UAV Registration Form

UAV ID:

Model:

Owner Email:

Register

Your UAV Keys

Public Key:

Private Key:
 [Download Private Key](#)

Figure 4.23: Register page

Figure 4.23 streamlines UAV registration. It collects key details like UAV ID, model, and owner’s email. Plus, it auto-generates a public-private key pair for secure communication. After submission, users receive their private key, ready for download.

Register Your UAV

[Home](#) [Login](#) [Register](#) [Devices](#) [Logs](#)

UAV Login Form

UAV ID:

Private Key:

Login

© 2024 SDN System. All rights reserved.

Figure 4.24: Login page

Figure 4.24 shows the login page which provides secure access to the SDN system. Users enter their credentials to authenticate and gain access to system features and functionalities.

Table	Action	Rows	Type	Collation	Size	Overhead
☐ keypair	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐ records	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐ uavregistration	★ Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐ uav_registration	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
4 tables	Sum	25	InnoDB	utf8mb4_general_ci	64.0 KiB	0 B

Figure 4.25: Database

Figure 4.25 shows the web-based database connection using MySQL. It stores additional information about the UAV and forwarding device.

4.5 Proposed System's strengths and limitations

The proposed system, by integrating blockchain technology into the SDN architecture, introduces a robust and resilient recovery mechanism that effectively addresses the SPOF issue of SDN controllers, ensuring continuous network availability and fault tolerance. Additionally, the system presents a flow migration mechanism for SDN forwarding devices, enhancing the reliability and updateability of network configurations. The use of blockchain technology guarantees secure and tamper-proof updates, which are crucial for maintaining the integrity and trustworthiness of the communication network.

Deployment of smart contracts on platforms like Hyperledger Besu and Ganache ensures that all transactions and rule updates are immutably recorded on the blockchain. This transparency facilitates auditability and accountability in network management. The decentralized nature of blockchain mitigates the risk of central point failures, bolstering overall system security. Furthermore, Hyperledger Besu's private configuration ensures that data access is restricted to authenticated participants, preserving data integrity and confidentiality. The deterministic properties of smart contracts help mitigate risks, ensuring consistent adherence to predefined terms and reducing the likelihood of network anomalies or disruptions.

However, the system also has limitations. The reliance on a permissioned network, as seen with Hyperledger Besu, may hinder scalability, limiting widespread adoption across various UAV communication networks, especially those requiring extensive scalability. The use of consensus protocols like QBFT, while effective at ensuring security, may introduce latency in transaction validations, particularly

during high transaction volumes. This latency can affect the real-time performance required for UAV communications. Additionally, the immutable nature of blockchain poses challenges in error rectification. Any defect or vulnerability in the smart contract logic can lead to significant issues, and correcting such errors within a blockchain environment can be complex and challenging.

In summary, while the proposed system significantly enhances fault tolerance, security, and transparency in UAV communication networks using SDN and blockchain technologies, it faces challenges related to scalability, transaction latency, and error rectification. Addressing these limitations will be crucial for the system's broader adoption and effective implementation. The system's success will depend on overcoming these hurdles to fully realize the benefits of integrating blockchain with SDN for UAV communication.

Chapter 5

Business Model

An in-depth business model presents essential elements such as the project's creation, delivery, and value capture. The business model of this project is given below:

5.1 Value Proposition

A project's value proposition shows the unique value that it offers to its users or solves a problem. The proposed system provides a secure and efficient controller recovery and forwarding device migration in Software-Defined Networking for UAV network. Blockchain technology is used to enable decentralized, tamper-proof controller recovery and safe flow rule migration to guarantee that UAV networks continue to function even in the case of controller failure. This is how the proposed solution, enhanced reliability, security of the system and seamless operation of the UAV network.

5.2 Customer Segments

Customer segments define the diverse group of people or institutions that a company or business seeks to reach to and engage with by their business or system. There are different customer segments for this project, which cover a variety of industries and uses. They are described below:

5.2.1 Commercial UAV Operators

In agriculture, UAVs are utilized for crop monitoring, pesticide spraying, and precision farming. It is also used for site inspections, safety checks and surveillance in construction and mining. In fact, the media used UAVs for aerial photography and filming. Moreover, UAVs are used for package delivery by businesses like UPS and Amazon. Thus, industries like logistics, agriculture, construction, mining, and media will be the customer or users of the system.

5.2.2 Government and Defense

Agencies such as the military, homeland security, and law enforcement will be the other type of customers of this solution. Because UAVs are used for surveillance, reconnaissance and battle missions in the military. Even in homeland security, UAVs monitor critical infrastructure and borders. Unmanned Aerial Vehicles are assisting in search and rescue, traffic monitoring as well as investigations into crime scenes in law enforcement.

5.2.3 Telecommunication Companies

UAVs are used for the inspection and maintenance of telecommunications infrastructure. In fact, it is extending coverage of the network to distant and unreachable areas. And the consistent delivery of services is ensured by reliable network management.

5.2.4 Emergency Services

The system will provide its services to emergency services such as disaster response teams, search and rescue operations and fire departments. In challenging terrains, UAVs can be useful in the search for missing persons. In addition, UAVs provide real time data and situational awareness in case of natural disasters. Thus, the reliability of the network is crucial for timely decisions and coordination.

5.2.5 Environmental Monitoring

Environmental Agencies, Research Institutions and NGOs will be the customers of this proposed system. Weather patterns and environmental changes can be collected by UAVs. Moreover, UAVs can monitor wildlife and their lifestyle. Continuous collection and analysis of data is supported by reliable networks.

5.3 Channels

Channels are the ways by which a business supplies its products or services to users. The project can maximize its market reach, drive customer acquisition and retention by effectively using these channels in order to establish a strong presence in the UAV, blockchains and SDN markets. A detailed overview of channels for this project is given below:

5.3.1 Direct Sales

Potential customers, in particular large businesses and government agencies, will be directly contacted by a dedicated team of sales professionals. This personalized approach will help us to build strong relationships, understand customers' requirements in detail and provide tailored solutions. To form strategic partnerships with world leaders in UAV manufacturing and SDN solution providers to ensure that this technology is readily available for their products.

5.3.2 Online Platform

By website which will provide a central point of information on the product, including features, benefits, prices and case studies. Demonstration videos, whitepapers and testimonials from users will also be hosted. An online store where small businesses and early adopters can buy the software or sign up for it directly.

5.3.3 Digital Marketing

Optimizing the content on the internet, driving organic traffic to the website, in order to rank highly in search engine results. Online advertising campaigns using LinkedIn, Google Ads, and other platforms to target particular client categories. To interact with the community, share updates and promote content using platforms such as LinkedIn, Twitter or YouTube.

5.3.4 Reseller and Distributor Networks

Collaborating with technology resellers that can reach a wider audience by showcasing the solution as part of their portfolio, especially in areas where direct presence is scarce. expanding the solution's reach by working with distributors who can manage sales, distribution, and support in different geographic regions.

5.3.5 Customer Support and Success

Establishing a systematic onboarding procedure for new clients that includes setup support, instruction, and preliminary configuration to guarantee adoption. Providing 24/7 hours technical support, advice, and assistance via a variety of channels (phone, email, live chat).

5.4 Customer Relationships

It means the type of the relationship a business or company has built with its clients or users. And it is necessary for the maintenance of customer satisfaction, loyalty and a lasting success.

5.4.1 Online Support and Self-Service

To help with technical problems, troubleshooting and other questions, via different channels such as telephone, e.g. phone, email or online chat. Build and manage online discussion boards where people may share queries, experience, and offer assistance to one another. Users are able to share knowledge and develop a sense of community as a result.

5.4.2 Customer Success Programs

To help new users get up and running quickly and efficiently, providing structured onboarding programs like training sessions. Appointing specialized customer success managers (CSMs) to make sure clients are using the product to achieve their intended goals. CSMs will keep an eye on usage, offer advice and insights, and work with customers to get the most out of their products.

5.4.3 Loyalty and Rewards Programs

Establishing loyalty programmes that provide discounts on renewals, access to unique features, or invitations to events as a way of appreciating clients for their on-going patronage and involvement. And giving discounts, credits, or extra services to clients who recommend new customers to the solution.

5.5 Revenue Streams

It is a way by which a business or company generates income from users. The revenue models of the proposed system are given:

5.5.1 Licensing Fees

Licensing the blockchain-based recovery and SDN forwarding device migration system to consumers may incur fees. This can be set up as a yearly licensing charge or as a one-time purchase, with varying levels according to the number of UAVs or size of the network. And providing a subscription-based business strategy in which users pay a regular charge (monthly, quarterly, or yearly) to access the system.

5.5.2 Partner and Reseller Programs

Generating revenue by collaborating with distributors and resellers to market the solution on your behalf and giving them a commission on the sales that they bring in. And collaborate with SDN providers and UAV manufacturers to integrate the solution into their devices directly. Licensing fee charging for every device sold that has the integrated solution.

5.5.3 Consulting Services

Charging for consulting services to assist clients in integrating and modifying the solution to suit their particular requirements. This covers the basic setting up, adjusting, and personalizing. Providing customer teams, workshops and training programmes on advanced features, best practices, and efficient blockchain-based system management.

5.5.4 Service Contracts

Delivering continuing support and maintenance services in accordance with a service agreement. This guarantees that users will always have access to technical assistance, bug patches, and updates. Service agreements will be offered either alone or in conjunction with the software license. In addition, providing enhanced support packages with faster response times, specialized staff, and extras like performance enhancements

5.5.5 Performance-Based Pricing

Adopting a price structure that takes into account the system's real usage, such as the quantity of data processed, the number of UAVs under management, or the number of blockchain transactions completed. Furthermore, providing performance-based pricing, in which users only pay when certain goals are accomplished, such as lower downtime, higher network reliability, or better security measures.

5.6 Key Resources

The resources required to provide users with the value proposition. The key resources required this proposed system are given below:

5.6.1 Technology Infrastructure

A tamper-proof and transparent log of all network activity is ensured in a blockchain-based system by using the blockchain architecture to validate and record the flow rules. By dynamically modifying the flow rules in response to conditions and data in real time, SDN controllers oversee the network. To effectively route data across the network, forwarding devices like switches and routers implement these flow rules. The combination of SDN's flexibility and control and blockchain's immutable record improves security and dependability.

5.6.2 Human Resources

Blockchain developers to develop and maintain the blockchain technology. Experts in the development, management, and deployment of SDN solutions. The platform will offer customers several ways to contact technical assistance, such as live chat, phone, and email. Any platform-specific technical problems will be handled by the technical support team.

5.6.3 Intellectual Property

Consensus algorithms that guarantee consensus on the blockchain state amongst all nodes, such proof-of-stack (PoS) systems. Blockchain and SDN technology integration techniques for UAV systems, including fault-tolerance, security, and forwarding device migration techniques. Innovative methods for blockchain-based controller recovery that guarantee system resilience and ongoing operation.

5.7 Key Activities

The most crucial steps that an organization or a company needs to do in order to function well. In order to guarantee that the system is creative, safe, dependable, and fulfills the needs of its users and stakeholders, key activities are necessary for the system's effective development, deployment, and operation. The activities for the proposed system is given below:

5.7.1 Research and Development

Continuously improving the system functionality to enhance performance, security, and reliability of the system. Developing and testing hardware and software components and conducting testing to ensure reliability and efficiency. Developing and testing procedures that allow smoothly integrated SDN with blockchain systems.

5.7.2 System Design and Architecture

Designing the network's entire architecture, including the positioning of forwarding devices and SDN controllers. Building robust secure methods for data transmission, authentication and storage in the network.

5.7.3 Implementation and Deployment

Writing and updating the controller, SDN, and blockchain platform code. SDN controller and forwarding device installation and configuration. Integrating every element to ensure smooth operation and communication between SDN devices and blockchain nodes.

5.7.4 Operations and Maintenance

Maintaining constant network monitoring to guarantee peak performance and promptly resolving any problems that may occur. Utilizing blockchain technology to implement recovery procedures for SDN controllers to guarantee continued operation. Managing the migration of forwarding devices to adapt to network changes.

5.7.5 Customer Support and Service

Provide assistance to users in troubleshooting and issue resolution. Sustaining close bonds with users by continuing to interact and offer support. Providing training sessions so that users may take use of the system's advantages and use it efficiently.

5.7.6 Marketing and Sales

Determining the technology's possible markets and clientele. Building marketing plans to increase brand recognition and system promotion. To efficiently reach and serve customers, manage the sales process and distribution networks.

5.8 Key Partnership

The network of vendors, associates, and supporters who support the company's or business operations. Key partnerships are essential to this project's business model since they increase capabilities, broaden reach, and guarantee the system's overall success. They are given:

5.8.1 Technology Providers

Alliances with businesses that specialize in blockchain technology to supply resources and assistance for putting strong blockchain solutions into practice. Partnerships with companies that build SDN controllers and network related hardwares to guarantee easy integration and improved network performance.

5.8.2 Investors and Venture Capitalists

Reaching out to venture capitalists and investors to obtain money for growing businesses, and reaching new markets.

5.8.3 Software and Platform Developers

Collaborating with businesses that specialize in creating software solutions for SDN operations and blockchain network management.

5.8.4 Funding and Grants

Obtaining funds and grants from government initiatives that promote technological innovation and the use of UAVs.

5.8.5 Industry Associations and Logistics

Associating with trade associations in order to promote laws and regulations that will benefit the industry and to remain up to date on standards, best practices, and trends. Working together with logistics companies to put SDN and blockchain into practice for safe and effective management.

5.9 Cost Structure

The numerous expenses incurred in running the business plan. The cost structure includes all of the costs related to creating, implementing, and managing the system. For this project the cost structure is given below:

5.9.1 Research and Development Costs

Payment for the project's engineers, developers, researchers, and other technical personnel. Costs associated with developing prototypes, running experiments, and improving technologies. Expenditure on the specialized tools and infrastructure required for the creation and testing of innovative technologies.

5.9.2 Infrastructure Costs

Expenses related to establishing and managing blockchain nodes, encompassing cloud services, software, and hardware. Costs associated with buying and setting up switches, routers, SDN controllers, and other networking hardwares.

5.9.3 Operational Costs

Charges for network operations centers and software tools, as well as for monitoring and maintaining the network. Costs associated with routine blockchain infrastructure and networking equipment maintenance and repairs.

5.9.4 Marketing and Sales Costs

Expenses associated with advertising, public relations, and digital marketing as well as other means of system promotion. Commissions and salaries for sales teams, together with costs for travel and client meetings all fall under the category of

sales-related expenses. Charges for bringing in new clients, such as those related to promotions, discounts, and onboarding assistance.

5.9.5 Customer Support and Service Costs

Payment and costs associated with technical support teams that help users and customers. Expenses associated with creating and providing training programs to users so they may utilize the system efficiently.

5.9.6 Partnership and Collaboration Costs

Expenses related to establishing and keeping alliances with suppliers of technology, academic institutions, government agencies, and other interested parties. Investment in licensing contracts, joint ventures, and other cooperative projects.

5.9.7 Financial and Other Costs

Costs incurred when a business is financed by loans or other debt. Rent, utilities, and other costs associated with keeping up an office for operations and administration. Refunds, such as dividends, given to venture capitalists and investors.

Chapter 6

Result Analysis

6.1 Performance Evaluation

The performance evaluation is set to benchmark the efficiency and responsiveness of the deployed contract on both Hyperledger Besu and Ganache blockchain. To test the performance of the smart contract on Hyperledger Besu, a machine with Core i5-3470 processor and 8GB DDR4 Ram was used. And for Ganache, another machine with a Core i5-8300H machine and 16GB DDR4 RAM was used. Test scripts were written using JavaScript and executed from the truffle develop console. Each containing batches of transactions and block information fetch functions to measure block creation time and retrieval time by altering send rate amount. The goal is to investigate the smart contract's performance on both platform and reliability under diverse transaction loads.

6.1.1 Configuration

The benchmarking was done by measuring the performance of smart contract functions: 'addFlowRule' (Block Creation Function), 'UpdateDeviceIdForPrevious-Device' (Block Creation Function), and 'getFlowRulesByDeviceId' (Retrieve Block Data Function). Each function underwent a four-cycle tiered transaction test, with each tier ranging from 25 to 150 transactions. The tests evaluated the efficiency and responsiveness of these functions on both Hyperledger Besu and Ganache under varying loads. Metrics such as block creation time, information retrieval time were meticulously recorded and analyzed. This comprehensive approach provided

insights into the performance and reliability of the smart contract across the two blockchain platforms.

6.1.2 Metric

The following metrics were observed during the performance evaluation:

- **Block Creation Time (s):** Block creation time refers to the average duration needed for the blockchain network to gather, validate, and include a new block of transactions into the ledger. This metric is crucial for understanding the platform's ability to efficiently process and integrate incoming transactions, ensuring that the blockchain remains up-to-date and transactions are confirmed within a reasonable timeframe. Factors influencing block creation time include network congestion, consensus algorithm efficiency, and the computational power of the nodes.
- **Information Retrieval Time (s):** Information retrieval time measures the latency involved in accessing specific data from the blockchain. This includes the time taken to fetch transaction details, read smart contract states, or retrieve block information. This metric is vital for assessing the responsiveness of the blockchain, particularly for applications that require real-time or near-real-time data access. High retrieval times can indicate bottlenecks in the data querying processes or limitations in the underlying architecture of the blockchain platform.
- **Send Rate (tx/s):** The send rate, expressed in transactions per second (tx/s), indicates the frequency at which transactions are submitted to the blockchain network. This metric is used to evaluate the scalability and throughput capacity of the blockchain under different transaction loads. A higher send rate suggests the platform can handle a larger volume of transactions within a given period, which is essential for applications requiring high transaction throughput. Monitoring the send rate helps in understanding how the blockchain performs under stress and whether it can maintain performance levels as the transaction load increases.

6.2 Proposed System's Performance Results

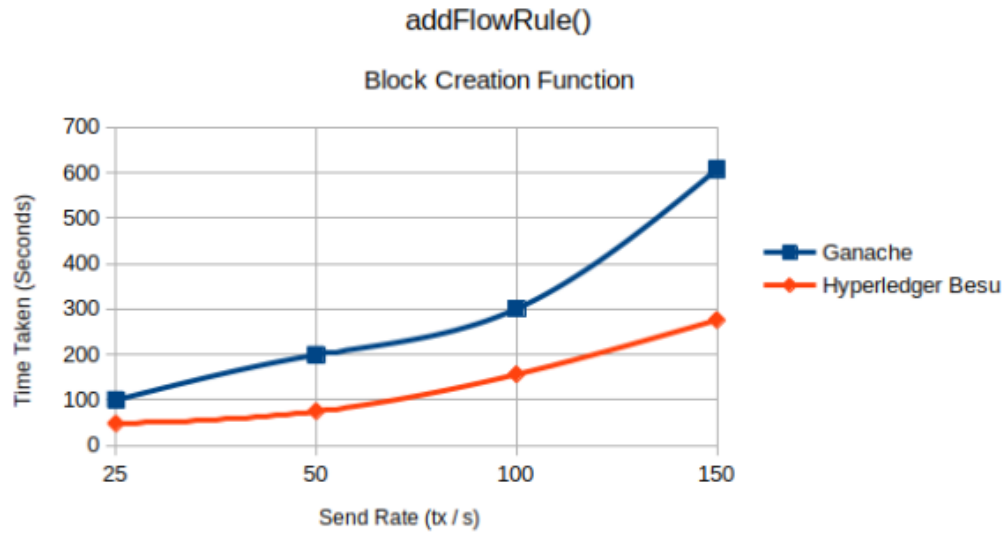


Figure 6.1: Flow Creation Function

The graph in **Figure 6.1** shows the comparison of block creation time taken between Ganache and Hyperledger Besu. Both lines show an upward trend, which is to be expected, as the transaction send rate is increasing gradually. It is evident from the graph that Hyperledger Besu takes almost half the transaction time of Ganache while creating blocks from flow rules.

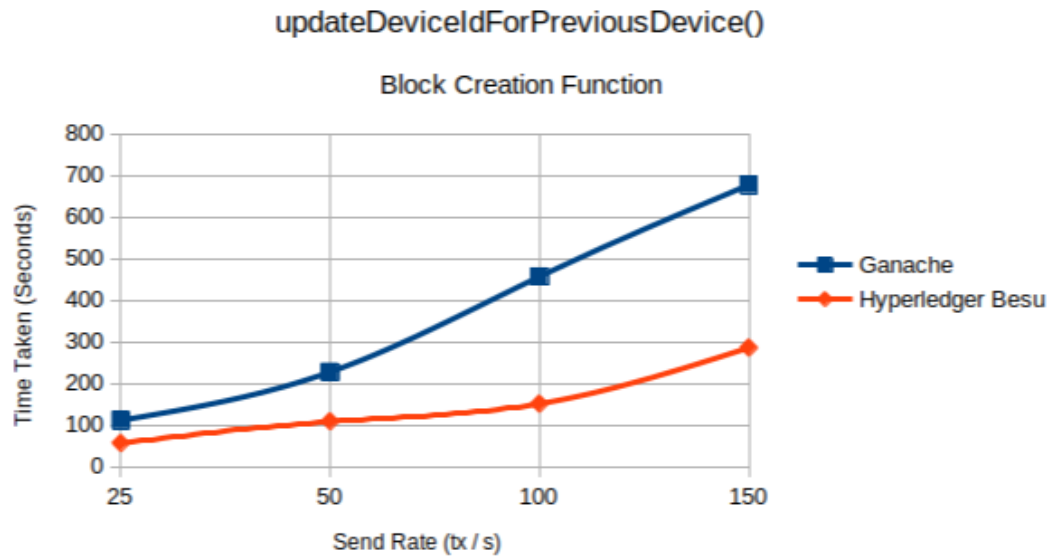


Figure 6.2: Flow Migration Function

The graph in **Figure 6.2** shows the time required to migrate flow rules for forwarding devices. This function takes much longer than the previous function to create blocks, as each forwarding device may contain many flow rules with its id. At first it seems both of them are performing consistently, but after some time, Ganache block creation time keeps increasing almost linearly compared to Besu's slight upward rise.

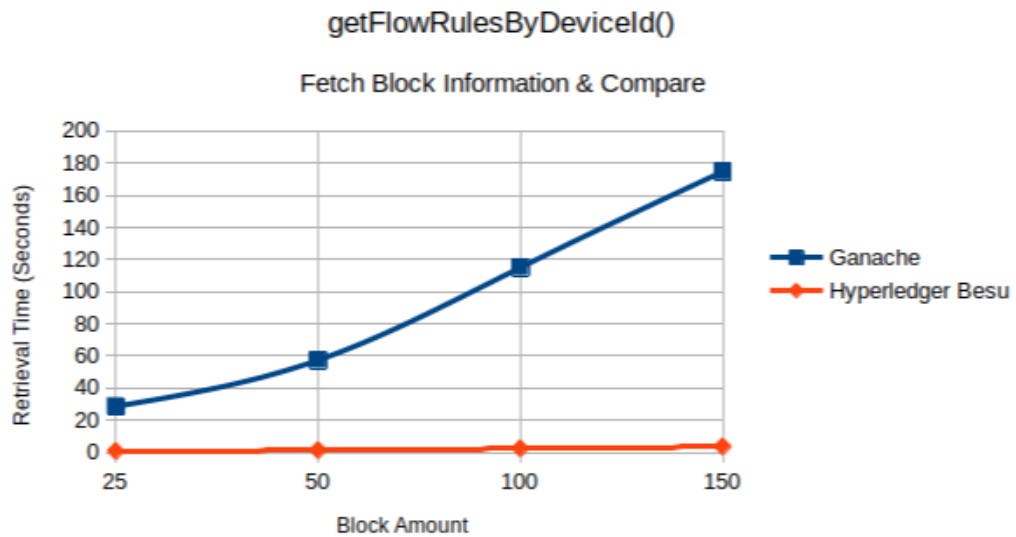


Figure 6.3: Flow Fetch Compare Function

Figure 6.3 shows a drastic performance difference between both platforms while fetching flow rule block data information. Besu takes almost no time while fetching a large amount of data, but Ganache retrieval time keeps increasing linearly as the block amount goes up.

6.2.1 Performance Comparison

In evaluating the performance of Hyperledger Besu and Ganache, particularly in terms of block creation time and block data retrieval, notable differences have emerged. Hyperledger Besu exhibits a significant advantage, demonstrating twice the speed in block creation compared to Ganache. Moreover, the retrieval of block data from Hyperledger Besu is nearly instantaneous, contrasting with Ganache's performance. This is crucial while building the proposed system for SDN controllers, as the retrieval time can greatly affect the necessary recovery time. This performance comparison underscores the importance of understanding the efficiency and effectiveness of different blockchain platforms, particularly in real-world applications where speed and responsiveness are critical factors.

Hyperledger Besu emerges as an optimal choice for the SDN recovery system for several compelling reasons. Firstly, its superior performance in block creation time, as evidenced by the comparison, ensures swift execution of recovery mechanisms, crucial for maintaining network availability and resilience in SDN environments. Additionally, Besu's efficient block data retrieval further enhances the system's responsiveness, enabling quick access to critical network information for recovery processes.

Moreover, one of the standout advantages of utilizing Hyperledger Besu lies in its permissioned network architecture, which eliminates the need for miners and transaction fees typically associated with public blockchain networks. In the context of the SDN recovery system, this translates to cost savings and predictable operating expenses, as there is no expenditure on incentivizing miners for transaction validation. By leveraging a permissioned network like Hyperledger Besu, the SDN recovery system can operate efficiently without the overhead costs commonly associated with public blockchain networks, thereby optimizing resource allocation and budget management and can give a better UAV communication system.

Chapter 7

Conclusion

In this chapter, we will discuss the impact of our capstone project on society and the environment. We will also discuss the ethical aspect and develop a Sustainability while concluding the report and discussing the plan.

7.1 Social, Ethical, and Environmental aspects

This proposed system, blockchain-based controller recovery and SDN forwarding device rule migration for UAVs has the potential to revolutionize network management. However, the implementation of this system will have an impact on society, cultural contexts, environment and ethics that need to be resolved. They are described below.

7.1.1 Social

Blockchain-based controller recovery and SDN forwarding device rule migration for UAVs can have various social issues. They are given below

- **Privacy:** UAVs that possess sophisticated data collecting capabilities are capable of obtaining photos, videos, and other private information about people and places. People could feel that they are being watched or surveilled without their knowledge, which presents serious privacy issues. The extensive usage of UAVs in both public and private areas may cause a sense of intrusion and general unease, which may spark community resistance.

- **Trust:** The trust and acceptance of new technologies such as this proposed system may be hindered by the general skepticism about them. In order to build trust, it is essential that benefits and guarantees be clearly communicated. Trust and acceptance can be fostered by engaging communities and addressing their concerns through a participatory approach.
- **Unequal access:** Access to such advanced technologies may be unequal, favoring the richer regions and leading to a technological divide. This can enhance existing social unevenness. In order to achieve broader social benefits, it is necessary to democratize access to UAVs and Blockchain technologies.
- **Public Safety:** There are concerns to public safety while using UAVs, particularly in crowded regions. Accidents, breakdowns, or a loss of control may result in property damage or bodily injury to persons. A layer of complexity is added when blockchain is integrated for control and recovery, and if this complexity is not adequately handled, it may result in more frequent and severe problems. It is essential to have strong safety protocols and fail-safes in place to reduce these hazards.

Blockchain-based controller recovery and SDN forwarding device rule migration for UAVs require addressing social issues in order to be implemented successfully. Establishing safety, building trust, and gaining broad acceptability all depend on a thorough, inclusive strategy that takes into account the varied effects on numerous communities and stakeholders.

7.1.2 Ethical

The application of blockchain-based controller recovery and SDN forwarding device rule migration for unmanned aerial vehicles (UAVs) raises a number of ethical issues that need to be taken into account.. They are described below

- **Inequity:** Social inequalities can be exacerbated by the use of modern technologies such as unmanned aerial vehicles and blockchain. In order to ensure that the technology serves broader societal interests, it is necessary to involve diverse communities in the development and implementation of the technology.

- **Privacy:** Without the explicit consent of the individual, UAVs may collect data without the consent of the individual, leading to potential privacy infringements. Ensuring that the data collected are anonymised and cannot be traced back to individuals without their consent.
- **Accessibility:** As it's not possible for everyone to use this technology, there may be a moral issue of injustice and prejudice. Thus the system might create new biases or promote prejudice based on gender, race, religion, or other disparities.
- **Responsibility:** The technology might be misused for monitoring or censorship purposes, and it's unclear who should be responsible for dealing with these issues.

Developers and operators can promote the appropriate and ethical use of blockchain-based controller recovery and SDN forwarding device rule migration for UAVs by taking these ethical aspects into account.

7.1.3 Environmental

A number of environmental issues need to be carefully managed in order to minimize the environmental impact of the implementation of the blockchain based SDN controller recovery and forwarding of UAV device rule migration. The main environmental issues related to this technology are listed here.

- **Energy Consumption:** An enormous amount of energy can be used by blockchain. If non-renewable energy is used in blockchain network mining, there will be a major environmental impact.
- **Electronic Waste(E-Waste):** Due to the high processing power requirements of blockchain activities, mining hardware must eventually be disposed of and upgraded on a regular basis. On the other hand , uavs have a finite lifespan and eventually turn into e-waste. Environment pollution may result from improper disposal of these components.
- **Carbon Footprint:** The energy consumption of the blockchain network and UAVs may have a significant carbon impact which contributes to global warming.

- **Resource Consumption:** Utilizing physical infrastructure and other limited resources can have negative environmental effects, as both UAVs and blockchain hardware depend on them.

A comprehensive strategy is needed to address these environmental problems, one that involves enforcing regulations to reduce disturbance of ecosystems, encouraging the use of sustainable methods in the manufacture and disposal of blockchain hardware and UAVs, and embracing energy-efficient technology. The implementation of blockchain-based controller recovery and SDN forwarding device rule migration for UAVs can be made more environmentally friendly and sustainable by taking these effects into account and reducing them.

7.2 Brief Summary

This article proposed a robust and reliable SDN framework for UAVs communication network, focusing on security and fault tolerance. Two critical challenges are addressed in the proposal: maintaining a continuous network service when it is interrupted, and verifying that flow rules are valid. Moreover, the proposed system is significantly enhanced by the inherent characteristics of the blockchain, such as immutability, transparency and traceability.

The framework includes a mechanism for SDN controllers to recover their functions quickly after network interruptions, thus enabling them to restore function in the shortest possible time. This will ensure that even in the event of a failure, the UAV Communication Network is capable of maintaining continuous service. For applications requiring real time data transmission and control, rapid recovery of controllers is crucial.

The system has a flow rule transfer mechanism in place to allow for the adaptation of UAVs operating in volatile and often unpredictable environments. This forwarding mechanism enables a seamless replacement of SDN forward devices, ensuring that the network will continue to function smoothly even when devices are added, removed or shifted. In order to maintain the effectiveness and reliability of UAV operations in various scenarios, this adaptability is essential.

In conclusion, the inclusion of blockchain technology to enhance security is a unique feature of the proposed scheme. The system ensures that the records of flow rules are unaltered and resistant to tampering as they are stored on a blockchain.

Before implementing flow rules, the SDN controller will be able to check their authenticity and substantially reduce the risk of false flow rule injections and illegal activities. This verification procedure ensures that only validated flow rules are inserted to enhance the security of the UAV network.

7.3 Future works

The development of this Blockchain-based SDN system for UAVs opens numerous avenues for future research and improvement for the system. Moreover, the popularity and necessity of blockchain technology enhances more path to future progress. These future works aim at further enhancing the system's capabilities, addressing potential limitations and expanding its applicability.

Interoperability in the context of Blockchain-based SDN systems for UAV communication refers to the ability of different blockchain networks, SDN controllers, and UAV platforms to seamlessly communicate, exchange data, and interoperate with each other. Interoperability problems arise as the softwarized UAV system is integrated with non-vendor-specific network hardware components like switches and routers. This required careful handling with the appropriate remedies. To enable communication and data exchange between heterogeneous blockchain networks, research is required to create interoperable blockchain protocols. This involves designing standardized protocols, consensus mechanisms, and interoperability layers to enable seamless interaction between different blockchain platforms used in SDN systems. Researching ways to manage massive amounts of flow rule records without sacrificing performance, including sharding, sidechains, and off-chain transactions. To increase the sustainability of the blockchain component, alternative consensus methods beyond proof-of-stake (PoW) such as delegated proof-of-stake (DPoS) and other energy-efficient algorithms can be researched. To ensure faster and more efficient recovery, it is necessary to develop adaptive recovery protocols that can dynamically adjust to different types and sizes of network failures.

To enhance the security of data and transactions on the blockchain, more advanced encryption methods can be implemented. Moreover, to identify and respond to threats in real time, the development of advanced intrusion detection and prevention systems that leverage the transparency of the blockchain. Doing projects in various operational environments to test the system's performance, scalability,

and security in real-world conditions. This proposed work will focus in the future on optimizing performance, enhancing security, ensuring regulatory compliance and expanding the applicability of the system. The aim of addressing these areas is knocking on the researchers and developers so that they can further refine the system, making it more efficient, reliable, and adaptable to a wide range of operational scenarios.

References

- Abou El Houda, Z., Hafid, A. S., & Khoukhi, L. (2019). Cochain-sc: An intra-and inter-domain ddos mitigation scheme based on blockchain using sdn and smart contract. *IEEE Access*, 7, 98893–98907.
- Aldaej, A., Atiquzzaman, M., Ahanger, T. A., & Shukla, P. K. (2023). Multidomain blockchain-based intelligent routing in uav-iot networks. *Computer Communications*, 205, 158–169.
- Alladi, T., Chamola, V., Sahu, N., & Guizani, M. (2020). Applications of blockchain in unmanned aerial vehicles: A review. *Vehicular Communications*, 23, 100249.
- Amin, R., Reisslein, M., & Shah, N. (2018). Hybrid sdn networks: A survey of existing approaches. *IEEE Communications Surveys & Tutorials*, 20(4), 3259–3306.
- Angelis, J., & Da Silva, E. R. (2019). Blockchain adoption: A value driver perspective. *Business Horizons*, 62(3), 307–314.
- Aoki, Y., Otsuki, K., Kaneko, T., Banno, R., & Shudo, K. (2019). Simblock: A blockchain network simulator. In *Ieee infocom 2019-ieee conference on computer communications workshops (infocom wkshps)* (pp. 325–329).
- Asaithambi, S., Ravi, L., Kotb, H., Milyani, A. H., Azhari, A. A., Nallusamy, S., ... Vairavasundaram, S. (2022). An energy-efficient and blockchain-integrated software defined network for the industrial internet of things. *Sensors*, 22(20), 7917.
- Belchior, R., Vasconcelos, A., Guerreiro, S., & Correia, M. (2021). A survey on blockchain interoperability: Past, present, and future trends. *ACM Computing Surveys (CSUR)*, 54(8), 1–41.
- Bharany, S., Badotra, S., Sharma, S., Rani, S., Alazab, M., Jhaveri, R. H., & Gadekallu, T. R. (2022). Energy efficient fault tolerance techniques in green cloud computing: A systematic survey and taxonomy. *Sustainable Energy Technologies and Assessments*, 53, 102613.

- Bholebawa, I. Z., Jha, R. K., & Dalal, U. D. (2016). Performance analysis of proposed openflow-based network architecture using mininet. *Wireless Personal Communications*, 86, 943–958.
- Boukria, S., Guerroumi, M., & Romdhani, I. (2019). Bcfr: Blockchain-based controller against false flow rule injection in sdn. In *2019 IEEE Symposium on Computers and Communications (ISCC)* (pp. 1034–1039).
- Bwalya, B., & Zimba, A. (2021). An sdn approach to mitigating network management challenges in traditional networks. *International Journal on Information Technologies and Security*, 13(4), 3–14.
- Cabaj, K., Wytrebowicz, J., Kuklinski, S., Radziszewski, P., & Dinh, K. T. (2014). Sdn architecture impact on network security. In *Fedcsis (position papers)* (pp. 143–148).
- Cachin, C., et al. (2016). Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers* (Vol. 310, pp. 1–4).
- Coit, D. W., & Zio, E. (2019). The evolution of system reliability optimization. *Reliability Engineering & System Safety*, 192, 106259.
- Derhab, A., Guerroumi, M., Belaoued, M., & Cheikhrouhou, O. (2021). Bmc-sdn: Blockchain-based multicontroller architecture for secure software-defined networks. *Wireless Communications and Mobile Computing*, 2021, 1–12.
- Dholakiya, D., Kshirsagar, T., & Nayak, A. (2021). Survey of mininet challenges, opportunities, and application in software-defined network (sdn). *Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2020, Volume 2*, 213–221.
- Dongre, J. G., Tikam, S. M., & Gharat, V. B. (2020). Education degree fraud detection and student certificate verification using blockchain. *Int. J. Eng. Res. Technol*, 9, 300–303.
- Faizullah, S., Khan, M. A., Alzahrani, A., & Khan, I. (2020). Permissioned blockchain-based security for sdn in iot cloud networks. In *2019 international conference on advances in the emerging computing technologies (aect)* (pp. 1–6).
- Foytik, P., Shetty, S., Gochhayat, S. P., Herath, E., Tosh, D., & Njilla, L. (2020). A blockchain simulator for evaluating consensus algorithms in diverse networking environments. In *2020 spring simulation conference (springsim)* (pp. 1–12).
- Garcia, D. F., Rodrigo, G., Entrialgo, J., Garcia, J., & Garcia, M. (2008). Experimental evaluation of horizontal and vertical scalability of cluster-based application servers for transactional workloads. In *8th international conference on applied informatics and communications (aic'08)* (pp. 29–34).

- Girela-López, F., López-Jiménez, J., Jiménez-López, M., Rodríguez, R., Ros, E., & Díaz, J. (2020). Ieee 1588 high accuracy default profile: Applications and challenges. *IEEE Access*, 8, 45211–45220.
- Gourisetti, S. N. G., Mylrea, M., & Patangia, H. (2019). Evaluation and demonstration of blockchain applicability framework. *IEEE Transactions on Engineering Management*, 67(4), 1142–1156.
- Guegan, D. (2017). Public blockchain versus private blockchain.
- Gupta, L., Jain, R., & Vaszkun, G. (2015). Survey of important issues in uav communication networks. *IEEE communications surveys & tutorials*, 18(2), 1123–1152.
- Gupta, R., Patel, M. M., Tanwar, S., Kumar, N., & Zeadally, S. (2021). Blockchain-based data dissemination scheme for 5g-enabled softwarized uav networks. *IEEE Transactions on Green Communications and Networking*, 5(4), 1712–1721.
- Haji, S. H., Zeebaree, S., Saeed, R. H., Ameen, S. Y., Shukur, H. M., Omar, N., ... others (2021). Comparison of software defined networking with traditional networking. *Asian Journal of Research in Computer Science*, 9(2), 1–18.
- Hasan, M., Rahman, A., & Islam, M. J. (2020). Distb-cvs: A distributed secure blockchain based online certificate verification system from bangladesh perspective. In *2020 2nd international conference on advanced information and communication technology (icaict)* (pp. 460–465).
- Hellman, M. E. (2002). An overview of public key cryptography. *IEEE Communications Magazine*, 40(5), 42–49.
- Hepp, T., Sharinghousen, M., Ehret, P., Schoenhals, A., & Gipp, B. (n.d.). *On-chain vs. off-chain storage for supply-and blockchain integration. it-inf. technol.* 60 (5–6), 283–291 (2018).
- Hosen, A. S., Singh, S., Sharma, P. K., Ghosh, U., Wang, J., Ra, I.-H., & Cho, G. H. (2020). Blockchain-based transaction validation protocol for a secure distributed iot network. *IEEE Access*, 8, 117266–117277.
- Hu, J., Reed, M., Thomos, N., Al-Naday, M. F., & Yang, K. (2020). Securing sdn-controlled iot networks through edge blockchain. *IEEE Internet of Things Journal*, 8(4), 2102–2115.
- Hu, N., Tian, Z., Sun, Y., Yin, L., Zhao, B., Du, X., & Guizani, N. (2021). Building agile and resilient uav networks based on sdn and blockchain. *IEEE Network*, 35(1), 57–63.

- Khalid, M., Hameed, S., Qadir, A., Shah, S. A., & Draheim, D. (2023). Towards sdn-based smart contract solution for iot access control. *Computer Communications*, 198, 1–31.
- Khan, M. A., Kumar, N., Mohsan, S. A. H., Khan, W. U., Nasralla, M. M., Alsharif, M. H., ... Ullah, I. (2022). Swarm of uavs for network management in 6g: A technical review. *IEEE Transactions on Network and Service Management*, 20(1), 741–761.
- Khan, S. N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E., & Bani-Hani, A. (2021). Blockchain smart contracts: Applications, challenges, and future trends. *Peer-to-peer Networking and Applications*, 14, 2901–2925.
- Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 ieee symposium on security and privacy (sp)* (pp. 839–858).
- Kumar, P., Kumar, R., Kumar, A., Franklin, A. A., & Jolfaei, A. (2022). Blockchain and deep learning empowered secure data sharing framework for softwarized uavs. In *2022 ieee international conference on communications workshops (icc workshops)* (pp. 770–775).
- Kumari, A., Gupta, R., Tanwar, S., & Kumar, N. (2020). A taxonomy of blockchain-enabled softwarization for secure uav network. *Computer Communications*, 161, 304–323.
- Kumutha, K., & Jayalakshmi, S. (2022). Blockchain technology and academic certificate authenticity—a review. *Expert Clouds and Applications: Proceedings of ICOECA 2021*, 321–334.
- Liao, S., Wu, J., Li, J., Bashir, A. K., & Yang, W. (2021). Securing collaborative environment monitoring in smart cities using blockchain enabled software-defined internet of drones. *IEEE Internet of Things Magazine*, 4(1), 12–18.
- Liu, B., Yu, X. L., Chen, S., Xu, X., & Zhu, L. (2017). Blockchain based data integrity service framework for iot data. In *2017 ieee international conference on web services (icws)* (pp. 468–475).
- Liu, H., Yoo, S.-J., & Kwak, K. S. (2018). Opportunistic relaying for low-altitude uav swarm secure communications with multiple eavesdroppers. *Journal of Communications and Networks*, 20(5), 496–508.
- Liu, L., Feng, W., Chen, C., Zhang, Y., Lan, D., Yuan, X., & Vashisht, S. (2020). Bs-iot: blockchain based software defined network framework for internet of things. In *Ieee infocom 2020-ieee conference on computer communications workshops (infocom wkshps)* (pp. 496–501).

- Lokesh, B., & Rajagopalan, N. (2020). A blockchain-based security model for sdn. In *2020 ieee international conference on electronics, computing and communication technologies (conecct)* (pp. 1–6).
- Lotlikar, T., & Shah, D. (2019). A defense mechanism for dos attacks in sdn (software defined network). In *2019 international conference on nascent technologies in engineering (icnte)* (pp. 1–7).
- Mann, S., Jain, T., & Vyas, A. (2020). The blockchain revolution: paradigm shifts in traditional voting practices. *International Journal of Computer Applications*, 176(37), 36–42.
- Mansoor, N., Hossain, M. I., Rozario, A., Zareei, M., & Arreola, A. R. (2023). A fresh look at routing protocols in unmanned aerial vehicular networks: a survey. *IEEE Access*.
- Manuskin, A., Mirkin, M., & Eyal, I. (2020). Ostraka: Secure blockchain scaling by node sharding. In *2020 ieee european symposium on security and privacy workshops (euros&pw)* (pp. 397–406).
- Mendiboure, L., Chalouf, M. A., & Krief, F. (2020). A scalable blockchain-based approach for authentication and access control in software defined vehicular networks. In *2020 29th international conference on computer communications and networks (icccn)* (pp. 1–11).
- Misra, S., Sarkar, K., & Ahmed, N. (2020). Blockchain-based controller recovery in sdn. In *Ieee infocom 2020-ieee conference on computer communications workshops (infocom wkshps)* (pp. 1063–1068).
- Moldamurat, K., Seitkulov, Y., Atanov, S., Bakyt, M., & Yergaliyeva, B. (2024). Enhancing cryptographic protection, authentication, and authorization in cellular networks: a comprehensive research study. *International Journal of Electrical & Computer Engineering* (2088-8708), 14(1).
- Narváez, R.-C., Carrión, J., Maldonado, L. B., & del Cisne Ruilova, M. (2022). Smart contracts for user registration on ethereum technology: Systematic literature review. *Communication, Smart Technologies and Innovation for Society: Proceedings of CITIS 2021*, 699–707.
- Natarajan, H., Krause, S., & Gradstein, H. (2017). Distributed ledger technology and blockchain.
- O'Neill, O. (2020). Consistency in action. In *Immanuel kant* (pp. 154–181). Routledge.
- PrasadRao, K. V., RadhaKrishna, P. K. S., Teja, G. M. C., & Panda, S. K. (n.d.). Blockchain based smart contract deployment on ethereum platform using web3. js and solidity.

- Preeth, E., Mulerickal, F. J. P., Paul, B., & Sastri, Y. (2015). Evaluation of docker containers based on hardware utilization. In *2015 international conference on control communication & computing india (iccc)* (pp. 697–700).
- Qiao, L., Dang, S., Shihada, B., Alouini, M.-S., Nowak, R., & Lv, Z. (2022). Can blockchain link the future? *Digital Communications and Networks*, 8(5), 687–694.
- Rahardja, U., Hidayanto, A. N., Lutfiani, N., Febiani, D. A., & Aini, Q. (2021). Immutability of distributed hash model on blockchain node storage. *Sci. J. Informatics*, 8(1), 137–143.
- Rahman, A., Islam, M. J., Khan, M. S. I., Kabir, S., Pritom, A. I., & Karim, M. R. (2020). Block-sdotcloud: Enhancing security of cloud storage through blockchain-based sdn in iot network. In *2020 2nd international conference on sustainable technologies for industry 4.0 (sti)* (pp. 1–6).
- Rahman, A., Islam, M. J., Montieri, A., Nasir, M. K., Reza, M. M., Band, S. S., ... Mosavi, A. (2021). Smartblock-sdn: An optimized blockchain-sdn framework for resource management in iot. *IEEE Access*, 9, 28361–28376.
- Rahman, T., Mouno, S. I., Raatul, A. M., Al Azad, A. K., & Mansoor, N. (2023). Verifi-chain: a credentials verifier using blockchain and ipfs. In *International conference on information, communication and computing technology* (pp. 361–371).
- Randles, M., Lamb, D., Odat, E., & Taleb-Bendiab, A. (2011). Distributed redundancy and robustness in complex systems. *Journal of Computer and System Sciences*, 77(2), 293–304.
- Saleh, O. S., Ghazali, O., & Rana, M. E. (2020). Blockchain based framework for educational certificates verification. *Journal of critical reviews*.
- Sarmah, S. S. (2018). Understanding blockchain technology. *Computer Science and Engineering*, 8(2), 23–29.
- Schehlmann, L., Abt, S., & Baier, H. (2014). Blessing or curse? revisiting security aspects of software-defined networking. In *10th international conference on network and service management (cnsm) and workshop* (pp. 382–387).
- Shi, S., He, D., Li, L., Kumar, N., Khan, M. K., & Choo, K.-K. R. (2020). Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey. *Computers & security*, 97, 101966.
- Shukla, A., Katt, B., Nweke, L. O., Yeng, P. K., & Weldehawaryat, G. K. (2022). System security assurance: A systematic literature review. *Computer Science Review*, 45, 100496.

- Sultana, N., Chilamkurti, N., Peng, W., & Alhadad, R. (2019). Survey on sdn based network intrusion detection system using machine learning approaches. *Peer-to-Peer Networking and Applications*, 12(2), 493–501.
- Turner, S. W., Karakus, M., Guler, E., & Uludag, S. (2023). A promising integration of sdn and blockchain for iot networks: A survey. *IEEE Access*.
- Uddin, M., Salah, K., Jayaraman, R., Pesic, S., & Ellahham, S. (2021). Blockchain for drug traceability: Architectures and open challenges. *Health informatics journal*, 27(2), 14604582211011228.
- Wan, L., Eysers, D., & Zhang, H. (2019). Evaluating the impact of network latency on the safety of blockchain transactions. In *2019 ieee international conference on blockchain (blockchain)* (pp. 194–201).
- Wang, H., Wang, J., Ding, G., Wang, L., Tsiftsis, T. A., & Sharma, P. K. (2017). Resource allocation for energy harvesting-powered d2d communication underlying uav-assisted networks. *IEEE Transactions on Green Communications and Networking*, 2(1), 14–24.
- Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X., & Wang, F.-Y. (2019). Blockchain-enabled smart contracts: architecture, applications, and future trends. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(11), 2266–2277.
- Wohrer, M., & Zdun, U. (2018). Smart contracts: security patterns in the ethereum ecosystem and solidity. In *2018 international workshop on blockchain oriented software engineering (iwbose)* (pp. 2–8).
- Zarrin, J., Wen Phang, H., Babu Saheer, L., & Zarrin, B. (2021). Blockchain for decentralization of internet: prospects, trends, and challenges. *Cluster Computing*, 24(4), 2841–2866.
- Zeng, Z., Zhang, X., & Xia, Z. (2022). Intelligent blockchain-based secure routing for multidomain sdn-enabled iot networks. *Wireless Communications and Mobile Computing*, 2022, 1–10.