systems

MDPI

Article

A Blockchain-Enabled Architecture for Secure and Transparent Distribution of Disaster Relief Supplies

Özgür Karaduman * and Gülsena Gülhas

Department of Software Engineering, Faculty of Engineering, Fırat University, Elazığ 23119, Türkiye; gulsenagulhass@gmail.com
* Correspondence: okaraduman@firat.edu.tr

Abstract

Ensuring the reliable, auditable, and privacy-oriented distribution of donations in disaster logistics constitutes a critical challenge due to multi-stakeholder coordination difficulties and the risk of misuse. This study presents a modular architecture, named SecureRelief, operating on a permissioned Hyperledger Fabric platform. The architecture integrates authentication based on Self-Sovereign Identity (SSI), Decentralized Identifiers (DID), and WebAuthn, together with Attribute-Based Access Control (ABAC), and enables the verification of delivery evidence through privacy-preserving validation using zero-knowledge proofs (ZKP). Documents are stored off-chain on the InterPlanetary File System (IPFS), while only cryptographic summary (hash) values sufficient for integrity verification are maintained on-chain. In scenario-based laboratory experiments, the blockchain layer demonstrated low latency (p95 < 16 ms) and stable transaction throughput, confirming its scalability. While the API layer handled high burst request loads with a 0% error rate, the additional computational overhead introduced by the integrated privacy-preserving (ZKP) mechanisms kept the end-to-end transaction latency within acceptable limits for disaster management applications (3.5–4.5 s).

Keywords: disaster management; blockchain-enabled systems; Hyperledger Fabric; humanitarian relief logistics; SSI; ZKP; IPFS; multi-party attestation

1. Introduction

Natural disasters impose significant challenges on societies worldwide, affecting economic activities and social structures across different regions [1,2]. In response to such events, the efficient allocation and distribution of relief materials has become a central concern in humanitarian logistics and disaster response operations [3,4]. Within this context, disaster management practices increasingly require systematic and risk-aware coordination mechanisms to ensure timely, transparent, and reliable relief operations [5]. Extreme weather events accelerated by climate change and ongoing geological activity have increased both the frequency and intensity of disasters [6]. Türkiye, in particular, is among the countries facing high disaster risk due to its active fault lines, variable climatic conditions, and complex topographical structure, making it vulnerable to earthquakes, floods, landslides, and wildfires. The seismic risks associated with the North Anatolian Fault Line, flood disasters in the Black Sea region [7], and wildfires in southern regions [8,9] constitute key indicators of the country's vulnerability to disasters. Following such events, the timely and complete delivery of essential humanitarian relief supplies, including water, food, medical materials, and shelter equipment, represents one of the most critical factors

Check for updates

Academic Editor: Wen-Chyuan Chiang

Received: 20 December 2025
Revised: 20 January 2026
Accepted: 2 February 2026
Published: 4 February 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

in increasing survival rates and accelerating recovery processes. However, within the highly complex post-disaster environment, the management of humanitarian relief logistics poses a major challenge [10,11]. Traditional supply chain management systems typically rely on centralized databases, which inherently introduce the risk of a single point of failure [12]. During crises, the collapse of centralized systems or their exposure to data manipulation undermines trust among donors, non-governmental organizations (NGOs), and public authorities. In this context, blockchain technology has been applied in disaster and humanitarian contexts to support risk reduction, accountability, and operational coordination during emergency situations [13,14]. At the same time, blockchain-based approaches have been extensively examined in the context of supply chain management and charity donation systems, highlighting their potential to improve transparency and process reliability [15,16]. Distributed Ledger Technology (DLT)-based approaches have been explored as a means to support trust and coordination in multi-stakeholder disaster and emergency management settings [17,18]. In parallel, blockchain-based solutions have been applied to improve operational efficiency and process reliability in humanitarian supply chains and disaster relief logistics [19,20]. Nevertheless, in scenarios such as disaster management, where multi-stakeholder coordination and operational continuity are critical, requirements related to access control, data privacy, and performance predictability bring permissioned blockchain approaches to the forefront. Within this scope, Hyperledger Fabric is frequently adopted in enterprise-oriented use cases due to its membership-based identity management via Membership Service Providers (MSP), channel architecture, and support for modular execution of smart contracts (chaincode). Fabric's Raft-based ordering service enables deterministic transaction ordering without mining, contributing to low latency, predictable costs, and energy efficiency in permissioned networks, while its execute–order–validate transaction pipeline enhances ledger consistency and auditability across multiple organizations [21]. This study proposes an end-to-end reliable, privacy-oriented, and high-performance blockchain architecture, referred to as SecureRelief, which covers the entire disaster relief process, from the donation stage to multi-party verification of delivery.

1.1. Motivation

In disaster management processes, both the physical delivery of relief supplies to affected areas and the verifiable recording of this process are critical [22,23]. The fragmentation of records across different systems, from the entry of donations into warehouses to final delivery, creates a "traceability gap" in the chain between donors and beneficiaries, making it difficult to provide a definitive answer to the question, "Did my donation truly reach its destination?" This uncertainty undermines both donor trust and inter-organizational coordination. Therefore, beyond technical improvements in disaster logistics, there is a need for a "trust architecture" capable of rebuilding societal trust [24]. Approaches proposed on public blockchain networks such as Ethereum often struggle to provide a sustainable and predictable operational foundation under disaster conditions due to volatile transaction fees and limited privacy capabilities [25]. For this reason, SecureRelief has been designed on a permissioned Hyperledger Fabric infrastructure, aiming to address cost, performance, and access control requirements in a more deterministic manner. The primary motivation of this study is to introduce an integrated system architecture that addresses end-to-end traceability and privacy in a unified manner, thereby filling a gap in the existing literature. By combining hybrid data storage using IPFS, WebAuthn-based authentication, Self-Sovereign Identity (SSI) and Decentralized Identifiers (DID), together with privacy-preserving validation mechanisms based on zero-knowledge proofs (ZKP), the auditability of the entire process, from donation to delivery, has been significantly strengthened.

1.2. Problem Statement

Post-disaster humanitarian relief operations are inherently complex, time-sensitive, and multi-stakeholder processes [26,27]. Three fundamental problem areas that hinder effective relief management and that this study aims to address are identified as follows:

Real-Time Traceability and Data Fragmentation: In existing systems, donation, logistics, and delivery data are typically stored across heterogeneous and disconnected databases. This fragmentation creates a traceability gap that prevents effective auditing of the fate of relief supplies.

Identity Management and Authorization Vulnerabilities: Centralized and static identity systems are insufficient for reliably authenticating the large number of actors involved in disaster management, including non-governmental organizations (NGOs), public authorities, and volunteers. Such static identity models fail to accommodate the dynamic nature of disaster-response actors, thereby increasing the risk of identity fraud and data breaches.

Lack of Integrated Automation and Auditability: Existing studies predominantly focus on isolated stages of the relief process. There remains a lack of an integrated automation mechanism that covers the entire workflow from donation to delivery, relies on multi-party approval (*m*-of-*n*), and minimizes human error while ensuring auditability.

In this context, the present study seeks to address the following core research questions (RQs):

- *RQ*1: How can a permissioned blockchain infrastructure prevent data fragmentation while ensuring the complete recording and verification of donations throughout disaster relief processes?
- *RQ*2: How can authentication and access control mechanisms, including SSI/DID, WebAuthn, and ABAC, prevent unauthorized access while preserving operational speed and usability in disaster environments?
- *RQ*3: How can blockchain's immutable ledger structure and smart contracts contribute to security, performance, and data integrity across donation, shipment, and delivery stages?
- *RQ*4: How can the resulting verifiable records be leveraged to enhance trust among donors and non-governmental organizations throughout the relief process?

1.3. Related Work and Theoretical Background

The application of blockchain technology in disaster management and humanitarian relief logistics has attracted considerable attention in the literature as a robust solution to challenges arising from the lack of trust, transparency issues, and data manipulation risks inherent in centralized systems. Existing studies in this domain primarily focus on system architectures, data privacy, and the management of relief assets.

Transparency, Traceability, Trust, and Logistics Management: A substantial body of work aims to enhance the traceability of donations and relief supplies by leveraging blockchain's immutable record-keeping capabilities. For instance, a system developed using ReactJS and Web 3.0 has been proposed to improve transparency in disaster relief, enabling donors to directly track their transactions while executing smart contracts on the Ethereum network [28]. Similarly, an Ethereum-based management system operating on the Sepolia test network has been designed to ensure donation immutability and accountability, supported by cost and security analyses [29].

Studies focusing on regional governance dynamics have proposed models that employ Hyperledger-based infrastructures to achieve transparency through peer-to-peer validation, particularly in the context of bureaucratic processes in Indonesia, spanning administrative levels from village heads to regional authorities [30]. In another Indonesia-based

study, a blockchain-driven engineering design process has been introduced to enhance collaboration among stakeholders, such as BNPB and PMI, in forecasting logistics needs and managing distribution, with the system reported to be ready for pilot deployment in West Java [31]. In long-term disaster scenarios, transparency and auditability in medical relief supply chains have been addressed through a blockchain-enabled "safeguard mechanism" integrated into an SCP-based SRMS [32]. Additionally, a separate study addressing time constraints and trust-related issues in disaster management combined a blockchain-based distributed architecture with load-balancing algorithms, such as depth-first search (DFS), demonstrating improved system performance and a significant reduction in transaction error rates under high data traffic conditions [33].

In parallel with these approaches, several studies have integrated blockchain infrastructures with IoT and IoMT-based sensor systems, aiming not only to immutably record donation and relief flows but also to enable real-time monitoring and management of logistics processes under disaster conditions [34,35]. Beyond logistics-focused solutions, comprehensive system designs have also been proposed that extend disaster management capabilities to include modules for missing-person registration, identification of critical zones, and mapping of refugee camps [36].

Recent studies have also examined the role of blockchain technologies in establishing trust and transparency across complex supply chain environments. Blockchain-enabled smart contracts have been employed to ensure secure and transparent traceability of physical assets, demonstrating how immutable records can strengthen accountability among independent stakeholders [37]. In parallel, hierarchical blockchain-based trust management architectures have been proposed to address security and scalability challenges in distributed and IoT-driven systems [38]. Empirical investigations in real-world supply chains further indicate that blockchain adoption can foster trust by enhancing data integrity, process visibility, and inter-organizational coordination [39].

Privacy, Identity Authentication, and Data Security: Protecting sensitive information during relief distribution has emerged as a central concern in recent research. A secure system utilizing searchable encryption techniques on the Ethereum blockchain has been proposed to preserve the privacy of donors and beneficiaries while enabling query functionality [40]. To further enhance trust in data sharing, another framework integrated blockchain technology with Decentralized Identity (DID) and zero-knowledge proofs (ZKPs), allowing verification without disclosure of sensitive information [41].

Similarly, an advanced donation scheme named *Eisdspa* has been introduced, combining BBS+ signatures and zero-knowledge proofs (ZKPoKs) to jointly provide authentication, auditability, and privacy protection [42]. Such systems aim to ensure that relief reaches the intended recipients while preventing the misuse of personal data.

Tokenization and Asset Management: More recent efforts have explored innovative models based on the tokenization of relief assets. The *Eghatha* project proposed a hybrid structure employing Fungible Tokens (FTs) for cash donations and Non-Fungible Tokens (NFTs) for in-kind aid, such as food and medical packages [43]. By incorporating offline verification protocols and multi-signature mechanisms, this system seeks to maintain operational efficiency and trust even in disaster environments with limited technological infrastructure [43].

Despite these valuable contributions, a comprehensive solution that unifies critical components, including passwordless authentication using WebAuthn, attribute-based access control (ABAC), data integrity through off-chain hashing, and delivery verification via multi-party attestation (m-of-n), within a single architecture remains lacking. This study addresses this gap by integrating these components into the proposed SecureRelief platform.

1.4. Proposed Approach

This study proposes a permissioned system architecture built on Hyperledger Fabric to ensure end-to-end reliability and traceability in disaster relief operations. By restricting network participation to authorized stakeholders, including donors, non-governmental organizations, public authorities, and logistics providers, the proposed architecture aims to enhance accountability across the entire relief lifecycle.

The originality of the SecureRelief architecture lies in the holistic integration of multiple security, data management, and automation mechanisms within a unified framework. First, advanced identity and access management is achieved through the combined use of Self-Sovereign Identity (SSI) and Decentralized Identifiers (DID), which reduce dependency on centralized authorities. These mechanisms are complemented by Zero-Knowledge Proof (ZKP) protocols that enable verification without revealing sensitive information. To further strengthen operational security, WebAuthn/Passkeys are employed to support passwordless authentication during field operations, while access control decisions are enforced through context-aware Attribute-Based Access Control (ABAC) policies.

Second, the architecture adopts a hybrid data storage model to balance scalability and integrity requirements. Large data objects, such as photographs and official documentation, are stored off-chain using the InterPlanetary File System (IPFS), whereas cryptographic hash values of these records are anchored on-chain to ensure tamper resistance and verifiability.

Third, operational efficiency is improved through the automation of core processes and system integration mechanisms. QR/GS1 technologies are incorporated to accelerate workflow execution, while the overall relief process is governed by a set of interlinked smart contracts organized into three logical layers, namely Donation, Shipment, and Delivery. Each layer is designed to autonomously trigger the subsequent stage, thereby ensuring continuity throughout the relief lifecycle.

Finally, delivery reliability is ensured through a multi-party attestation mechanism based on an m-of-n signature model. Rather than relying on a single authority, delivery confirmation requires the endorsement of multiple independent stakeholders, which enhances trust and mitigates the risk of unilateral manipulation.

1.5. Contributions

This study contributes to the fields of disaster management and blockchain-based systems by introducing an integrated and privacy-aware security architecture that combines Self-Sovereign Identity, WebAuthn-based authentication, Zero-Knowledge Proofs, and Attribute-Based Access Control within a single framework tailored to disaster scenarios. By jointly employing these mechanisms, the proposed system enables robust identity verification and access control without compromising sensitive information.

In addition, the study presents a hybrid data management strategy that combines on-chain and off-chain storage through IPFS, thereby improving system scalability while preserving data integrity and auditability. The proposed modular smart contract architecture further contributes to operational efficiency by organizing donation, shipment, and delivery processes into a three-layered chaincode structure that supports autonomous execution and minimizes human-induced errors.

Finally, the proposed trust and accountability model addresses the traceability gap commonly observed in disaster relief operations. By incorporating a multi-party attestation mechanism, the system establishes an auditable and transparent infrastructure aimed at restoring societal trust in humanitarian aid distribution.

1.6. Paper Organization

The remainder of this paper is organized as follows. Section 2 presents the proposed SecureRelief architecture in detail, describing the permissioned network structure based on Hyperledger Fabric, the modular smart contract layers, namely Donation, Shipment, and Delivery, and the adopted hybrid data storage approach. This section also examines one of the core contributions of the study, namely Zero-Knowledge Proofs (ZKPs), demonstrating how delivery evidence and user attributes can be verified without disclosing sensitive information. Furthermore, the integration of ZKP mechanisms with SSI/DID, WebAuthn-based authentication, and ABAC-based access control is discussed in detail. Section 3 provides the performance and scalability evaluation of the system, analyzing transaction latencies, end-to-end processing times, and the additional computational overhead introduced by ZKP verification mechanisms based on experimental results. Section 4 discusses the obtained findings in comparison with the related literature and highlights the gains achieved by SecureRelief in terms of privacy, traceability, and accountability. Finally, Section 5 concludes the paper by summarizing the main findings and outlining potential directions for future research.

2. System Model and Design

The proposed SecureRelief architecture is designed as a permissioned blockchain-based system that ensures end-to-end traceability and trust across disaster relief workflows. Its core components and interactions are structured to support verifiable operations under multi-stakeholder conditions. These design principles form the foundation of the system model described in the following sections.

2.1. Overall System Architecture

The proposed system architecture adopts a multi-layered structure designed to provide a holistic solution to challenges arising from the lack of trust, transparency, and traceability in disaster relief operations. The architecture is modeled to manage the entire relief workflow in an end-to-end manner, covering stages from the initiation of donations to logistics shipment, delivery verification, and auditable reporting. The system operates on a permissioned blockchain infrastructure built upon the Hyperledger Fabric framework, ensuring that only authorized actors, including non-governmental organizations (NGOs), public authorities, logistics providers, and donors, are permitted to participate in the network, thereby enhancing security and accountability.

The layered architecture illustrated in Figure 1 is organized into five main modules, designed in accordance with security and scalability principles. The uppermost User Layer enables stakeholders, such as donors, NGOs, and logistics operators, to access the system, while network traffic is filtered through an API Gateway and an Identity and Access Layer rather than being directly forwarded to blockchain nodes. This design minimizes the attack surface by isolating the blockchain core components, including the ledger and ordering services, from direct exposure to external interactions.

Verified transactions passing through the security layer are forwarded to the Blockchain Transaction Layer, which constitutes the core of the system, where the fundamental components of the disaster management process are executed. Inter-organizational data sharing and consensus are maintained over a single shared channel, referred to as *reliefchannel*, while donation, shipment, and delivery processes are managed through three modular smart contract structures, namely Donation, Shipment, and Delivery chaincodes. Transaction validation and ledger consistency are ensured by Hyperledger Fabric's Raft-based Ordering Service, which provides the performance required to support high transaction volumes during disaster scenarios.

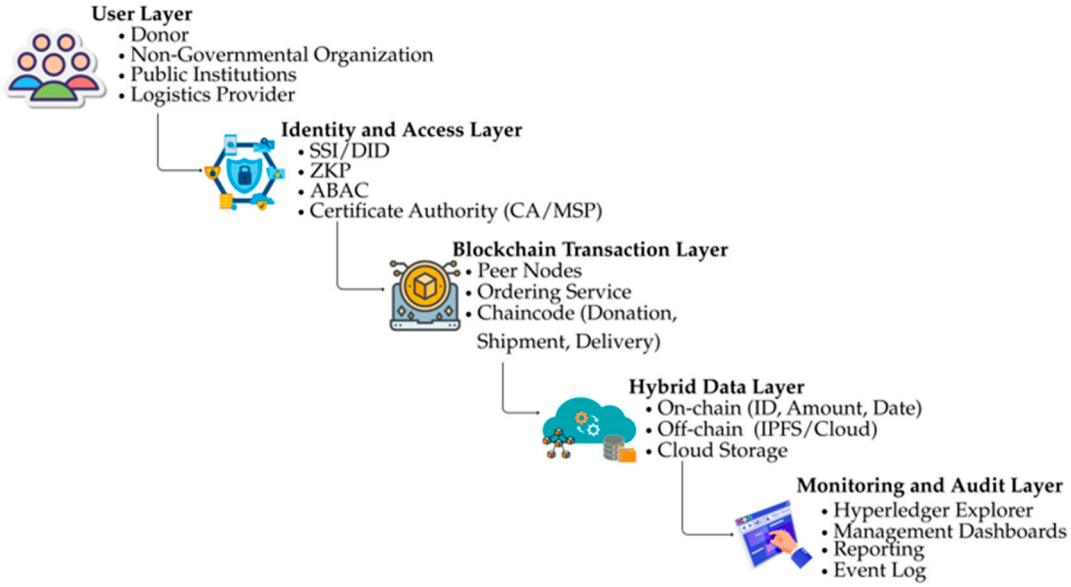


Figure 1. Proposed blockchain-based system architecture for disaster relief management.

To maintain scalability under increasing data loads, a Hybrid Data Layer combining on-chain and off-chain storage has been designed. While only critical transaction metadata, such as hashes, identifiers, and timestamps, are stored on the blockchain, large-scale evidence documents are maintained on IPFS, and relational data together with identity-related information are stored in off-chain components based on PostgreSQL. In this architecture, data integrity is guaranteed by anchoring cryptographic hash values of off-chain files onto the blockchain ledger.

The operational health and transparency of the overall architecture are supervised by the Monitoring and Auditing Layer. System resource utilization and transaction latencies are monitored in real time using Prometheus and Grafana, while integration with Hyperledger Explorer enables visualization of block and transaction details for both system administrators and auditing authorities (GovOrg), thereby ensuring full accountability.

Building on this architectural overview, Figure 2 provides a detailed, component-level representation of the system, illustrating how the conceptual layers are instantiated through concrete Hyperledger Fabric services, organizational peer nodes, and integrations with external systems.

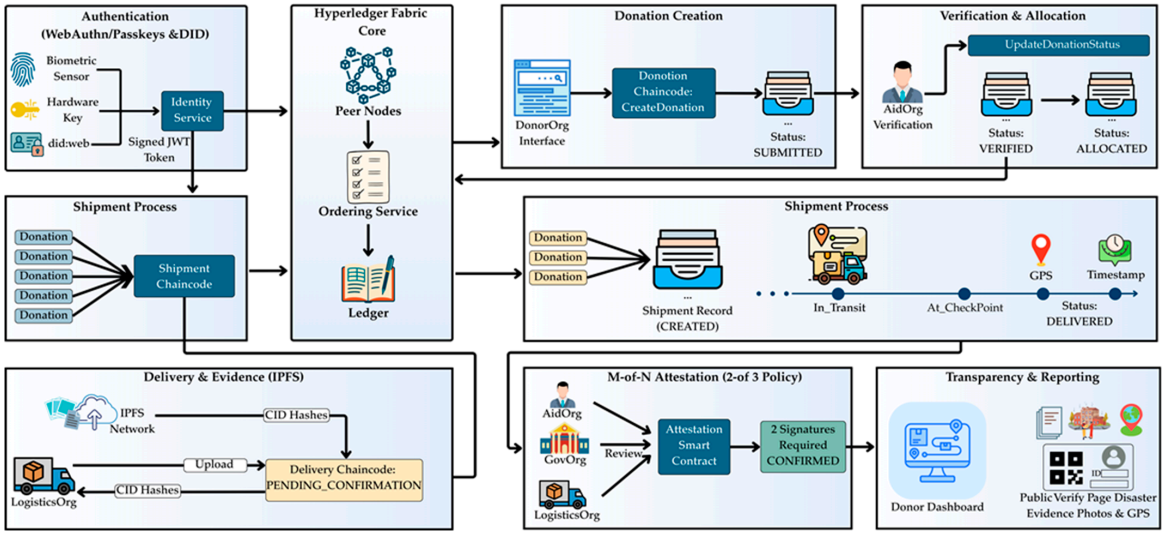


Figure 2. Detailed component-level architecture of the proposed system.

Figure 2 illustrates the component-level architecture of the proposed SecureRelief system and the interactions among its core building blocks. At the center of the architecture lies a permissioned Hyperledger Fabric network composed of peer nodes representing multiple organizations and a Raft-based Ordering Service. Consistent with the architectural design introduced earlier, all participating organizations communicate through a shared channel, referred to as *reliefchannel*, over which transactions are executed and maintained within a consistent distributed ledger.

The system's business logic is implemented in a modular manner through three dedicated chaincode components, namely Donation, Shipment, and Delivery, which respectively govern donation intake, logistics operations, and delivery verification processes. This modularization enables a clear logical separation of responsibilities while preserving the end-to-end integrity of the workflow from donation to final delivery.

Figure 2 further depicts the placement of the API Gateway and identity and access management services that mediate interactions between external clients and the blockchain network. This layer enforces authentication and authorization checks on incoming requests, ensuring that only valid and policy-compliant transactions are forwarded to the blockchain layer. Such an approach enhances the operational resilience of the system, particularly under high request volumes that may arise during disaster scenarios.

Finally, the proposed hybrid data management approach is represented at the component level. While essential metadata required for transactional integrity are stored on-chain, large-volume evidence artifacts are maintained in off-chain components, such as IPFS. Cryptographic references anchored on the blockchain establish a verifiable linkage between on-chain and off-chain data, enabling integrity verification. In conjunction with the monitoring and auditing components, this design supports transparent inspection of system performance and transaction history.

Formal System Model and Notation: To formally capture the layered architecture and transaction validation logic described above, the proposed system is expressed using a set of symbols and functional relations. The core notations used throughout the subsequent formalization are summarized in Table 1.

Table 1. Symbols and definitions used in the formalization of the SecureRelief system.

Symbol	Description
S	Global set representing the proposed system
N	Blockchain network topology
O	Set of organizations participating in the network
P	Set of peer nodes associated with participating organizations
Ω	Ordering Service component responsible for transaction sequencing
t_x	Individual transaction object
D_k	k-th donation object
S_s	Shipment object
R_r	Delivery and verification object
cid_f	Content Identifier of a file stored on IPFS
h_f	Cryptographic hash of a file (SHA-256)
ID_u	User identity object (e.g., DID, public key)
$Verify(vk, x, \pi)$	Zero-knowledge proof (ZKP) verification function
π	ZKP
$Policy(u, tx)$	ABAC-based authorization policy for a transaction
m, n	Threshold parameters for multi-party attestation (m-of-n)
$a_i \in \{0, 1\}$	Approval decision of an authorized organization
$Valid(tx)$	Global transaction validity function

In the proposed architecture, the system can be defined as a whole by the set $S = \{U, A, B, D, M\}$, where U represents the User Layer, A denotes the Identity and Access Layer, B corresponds to the Blockchain Layer, D refers to the Hybrid Data Storage Layer, and

M represents the Monitoring and Auditing Layer. The transaction flow of the disaster relief process is realized through a functional transformation across these layers, as expressed in Equation (1):

$$f : U \xrightarrow{auth} A \xrightarrow{tx} B \xrightarrow{store} D \xrightarrow{verify} M \tag{1}$$

This structure ensures that each transaction is processed in a verifiable and traceable manner. The transaction validity condition is formally defined by Equation (2):

$$\forall tx \in S : Valid(tx) = Auth(u) \wedge Policy(u, tx) \wedge Integrity(tx) \tag{2}$$

This formulation indicates that a transaction is accepted only if it satisfies the following three fundamental conditions:

$Authentication(Auth(u))$: The transaction must be initiated by a user whose identity has been verified by the Identity Service and who possesses a valid DID.
$Authorization(Policy(u, tx))$: The transaction must comply with the user's Attribute-Based Access Control (ABAC) policies and satisfy the zero-knowledge proof (ZKP) assertions enforced through the Privacy Gateway.
$Data\ Integrity\ (Integrity(tx))$: The transaction must satisfy cryptographic hash-based verification, ensuring consistency between on-chain records and their corresponding off-chain data.

Through this layered formalization, the proposed system differs from fragmented blockchain-based solutions reported in the literature by jointly addressing security, data integrity, and transaction traceability within a unified disaster relief logistics model.

2.2. Organizational Structure of the Network

The proposed system adopts a permissioned blockchain-based network model designed to ensure that donation, shipment, and delivery processes in disaster scenarios are executed over a reliable and traceable infrastructure. By allowing only authorized institutions and users to participate in the network, this security-oriented design aims to address common challenges in disaster logistics, including fraud, data manipulation, and the lack of effective oversight. The network is built upon the Hyperledger Fabric architecture, whose multi-organization participation model satisfies the system's interoperability and enterprise-level scalability requirements. Each participating entity is represented as a distinct organization within the network, equipped with its own Certificate Authority (CA) for identity issuance and a Membership Service Provider (MSP) component for membership management. As a result, all institutions operating within the network possess cryptographically verifiable digital identities, rendering transactions auditable and legally traceable.

The proposed system, in the context of disaster management, is built upon a network topology defined as $N = (O, P, C, \Omega)$, where the participating organizations are represented by $O = \{o_1, o_2, \ldots o_n\}$, the transaction-validating peer nodes by $P = \{p_1, p_2, \ldots p_n\}$, the shared communication channel by C (*reliefchannel*), and the ordering service by Ω. Each organization in the network possesses a cryptographically verifiable digital identity and MSP registration in the form $id_{org_i} = \left(cert_{CA_i}, msp_i\right)$. This structure guarantees organizational-level accountability and ensures that all transactions are traceable and auditable within the permissioned blockchain environment.

2.2.1. System Stakeholders

The actors involved in the system are defined by the set *Actors* = (*Donor, NGO, Government, Logistics, Auditor*) and the permissions of each stakeholder are determined through a role assignment function $role : Actors \rightarrow Tasks$. To establish trust in disaster

donation processes, the system adopts a multi-party attestation (m-of-n) principle rather than relying on unilateral declarations. The validity of a transaction, denoted as $Valid(tx)$, depends on achieving a minimum number of approvals (m) from the set of authorized organizations (n), as formally expressed in Equation (3):

$$Valid(tx) \Longleftrightarrow \sum_{i=1}^{n} approve_i(tx) \geq m \tag{3}$$

Here, $approve_i(tx)$ represents the state in which an organization cryptographically signs and approves the transaction. The stakeholders participating in the system are modeled according to the technical roles defined within the project scope, as summarized in Table 2.

Table 2. Stakeholder classification and functional roles in the relief blockchain network.

Stakeholder	Role Domain	System Function
Donors (DonorOrg)	Resource provider	Donation creation (CREATE), tracking of own donations
Non-Governmental Organizations (AidOrg)	Aid organization	Donation verification (VERIFY), shipment planning, distribution management
Government Authorities (GovOrg)	Authority and oversight	System auditing (Audit), donation rejection, delivery attestation
Logistics Providers (LogisticsOrg)	Transportation operations	Shipment initiation, checkpoint registration, evidence upload

2.2.2. Blockchain Network Structure

The proposed system employs the channel architecture of Hyperledger Fabric to support data isolation and process management. Contrary to conventional multi-channel designs, all organizations participating in the network communicate over a single shared channel, referred to as *reliefchannel*. Data confidentiality and process separation are achieved not through multiple channels, but via a modular chaincode (smart contract) architecture, consisting of the following components:

1. *Donation Chaincode*: Manages the lifecycle of donations.
2. *Shipment Chaincode*: Tracks and governs logistics operations.
3. *Delivery Chaincode*: Handles delivery confirmation and attestation processes.

Through this design, sensitive shipment-related information, such as donor identities or route security details, is processed exclusively by authorized stakeholders, while transaction summaries are transparently maintained on the main ledger. In this architecture, the validity of a transaction on a channel is governed by logical policy rules rather than physical network access, such that a transaction is processed on a given channel only if the condition $tx \in channel_k \Leftrightarrow Policy(channel_k, tx) = true$ is satisfied.

2.2.3. Interaction Model

The interaction model illustrated in Figure 3 governs the process from the initial donation entry to final delivery through a sequence of interdependent state transitions. Although the workflow follows a linear progression, each stage (Donation $\rightarrow$ Shipment $\rightarrow$ Delivery) is bound to a cryptographic precondition. For instance, the creation of a shipment record within the Shipment Chaincode requires the existence of a donation reference that has already attained a VERIFIED status within the Donation Chaincode. This dependency structure mathematically enforces cross-process data consistency, ensuring that each operational step is executed only upon the successful verification of its preceding state.

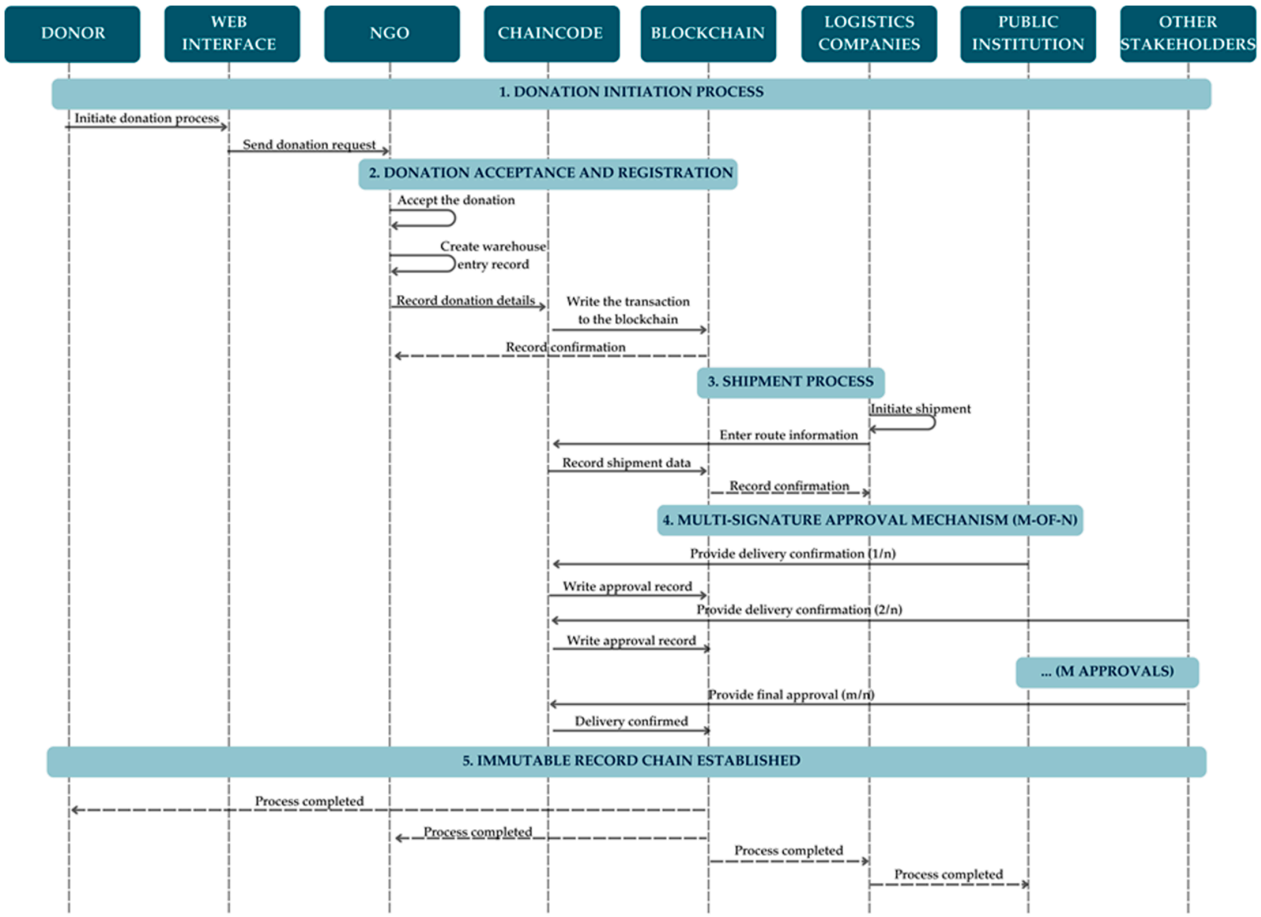


Figure 3. Sequence diagram of the blockchain-based donation–shipment–delivery process.

2.3. Transaction Workflow Model

In the proposed system, donation, shipment, and delivery processes are treated as an interconnected workflow. Each stage is modeled through transactions initiated by authorized actors, the records generated by these transactions, and the corresponding state transitions of those records. In this way, the entire relief chain is maintained as an end-to-end traceable and auditable process.

2.3.1. Donation Record

The workflow begins with the donor securely accessing the system and creating a donation. At the application layer, authentication is performed through OTP-assisted onboarding and passwordless verification using WebAuthn/Passkeys. Upon successful authentication, a JWT is issued to the client, embedding attributes such as the user's DID, organizational MSP, and assigned role. In privacy-sensitive scenarios, users may validate role or membership attributes without disclosure by presenting zero-knowledge proof (ZKP) assertions. A donation record is created on the Donation Chaincode via the *CreateDonation* operation, and its initial system state is set to SUBMITTED by default. The record is subsequently verified by an NGO (AidOrg) and transitioned to the VERIFIED state, or it may be marked as REJECTED by an authorized AidOrg or GovOrg. This state machine is constrained by predefined transition rules enforced at both the application and chaincode layers. Each donation is associated with a unique identifier, and the donation object (summary) is formally modeled as expressed in Equation (4):

$$D_k = \langle donationID,\ itemType, quantity,\ targetRegion,\ status, createdAt,\ evidenceRefs \rangle \quad (4)$$

Here, evidenceRefs encapsulates the Content Identifier (CID) generated as a result of storing donation-related documents on IPFS, along with the corresponding SHA-256 content hash computed to ensure file integrity. By avoiding the storage of large files directly on the blockchain, system performance is preserved while maintaining verifiable data integrity.

2.3.2. Logistics Tracking and Checkpoint Monitoring

Once a donation attains the VERIFIED state, the shipment process is initiated. Shipments are created on the Shipment Chaincode through the *CreateShipment* operation and are initially assigned the CREATED state. As field operations progress, the shipment transitions sequentially through the states IN_TRANSIT, AT_CHECKPOINT, and ARRIVED. Checkpoints represent verification records that enable rapid data entry in the field, particularly through QR-based modules. Each checkpoint is formally represented in a manner consistent with the technical model, as defined in Equation (5):

$$cp_j = \langle checkpointId, location, status, recorder, timestamp, notes \rangle \quad (5)$$

2.3.3. Delivery and Multi-Party Attestation

When a shipment reaches the ARRIVED state, a delivery record is created by LogisticsOrg on the Delivery Chaincode via the CreateDelivery operation, and its state is initialized as PENDING_CONFIRMATION. Delivery evidence, such as photos or signed reports, is uploaded to IPFS; the evidenceRefs field stores the corresponding CIDs, while attestation records may optionally include the hash of the evidence when required. Whether a delivery is considered valid is determined by the system's m-of-n attestation policy. In the technical design, the policy object specifies the required minimum number of approvals (M), the set of organizations eligible to attest (N), and, optionally, organizations whose approval is mandatory. The default policy example is "2-of-3", where the eligible organizations are AidOrgMSP, GovOrgMSP, and LogisticsOrgMSP, with AidOrgMSP and GovOrgMSP defined as mandatory approvers. Attestation is carried out through AddAttestation and, in privacy-sensitive scenarios, through AddZkpAttestation. Each attestation record includes the attestation outcome (approve/reject), the attesting organization and role, and a timestamp. The multi-party approval threshold is mathematically expressed in Equation (6):

$$\sum_{i=1}^{n} a_i \geq m \Rightarrow DeliveryStatus = CONFIRMED \quad (6)$$

Here, $a_i \in \{0, 1\}$ represents the approval decision of each authorized attestor, where a value of 1 denotes an APPROVED decision. If the policy object includes a set of requiredOrgs, an additional constraint is enforced such that at least one valid approval must be obtained from these mandatory organizations (in SecureRelief, for example, AidOrgMSP and GovOrgMSP). Once the attestation threshold is satisfied, the delivery state transitions to CONFIRMED; otherwise, the process remains in the PENDING_CONFIRMATION state or is transitioned to REJECTED, depending on the attestation outcomes.

2.3.4. Monitoring, Logging, and Performance Analysis

Throughout the workflow, the system collects events generated during donation, shipment, and delivery processes both as immutable audit trails on the blockchain layer and as operational metrics and logs within the monitoring layer. The API Gateway exposes dedicated endpoints for metric collection and event reporting, enabling real-time monitoring of latency, error rates, and transaction throughput via Prometheus and Grafana. In addition, to support transparency objectives, a public verification pipeline is provided for donors and third-party verifiers. In the technical design, the/api/public/verify/donation/:id end-

point returns the donation–shipment–delivery linkage along with any associated evidence references, thereby producing a verifiable response to the question of whether the aid has actually reached its intended destination.

2.4. Hybrid Data Structure and Storage Design

Disaster relief operations involve the collection and processing of diverse data types, including transaction records, documents, images, and performance metrics originating from multiple sources. Writing all such data directly to a distributed ledger, such as a blockchain, is inefficient in terms of both performance and cost [44]. Accordingly, the proposed system adopts a hybrid data architecture in which on-chain and off-chain data are synchronized. This architecture is further supported by transient storage components and system monitoring layers. As illustrated in Figure 4, data are organized into five main categories based on their security, accessibility, and volume characteristics:

1. *On-Chain Data*: Transactional data stored on the blockchain, characterized by low volume but requiring high integrity.
2. *Off-Chain Data (IPFS)*: Images, documents, reports, and evidence files stored in the distributed file system (IPFS).
3. *Off-Chain Data (Cloud Storage)*: Large documents requiring institutional backup, maintained redundantly within centralized cloud-based systems.
4. *Operational Database (PostgreSQL)*: A relational layer storing authentication, identity, and service-related data required by privacy and application operations
5. *Monitoring Data*: Logs and metrics capturing performance indicators, transaction durations, and overall system health.

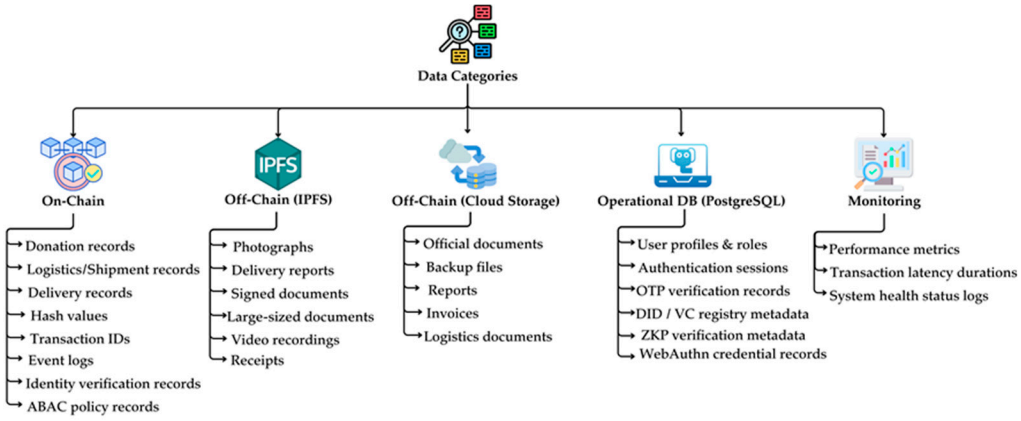


Figure 4. Distributed data management for the disaster relief system: on-chain and off-chain approach.

2.4.1. On-Chain Data Structure

On-chain data comprise transactional records such as donation operations, shipment records, and delivery attestations, together with their associated off-chain references (e.g., evidence CIDs). By being stored on the blockchain, these data provide high integrity and traceability. In the proposed system, the blockchain ledger is represented by the set $\mathcal{L} = \{D_k, S_s, R_r\}$. Within this set, the Donation object (D_k) is defined to include the donation's unique identifier (*donationId*), status, amount, target region, timestamp, and the originating organization (*createdByOrg*). The Shipment object (S_s), created to track logistics operations, contains references to the associated donations (*donationIds*), carrier organization (*carrierOrg*), route details, and status updates recorded at checkpoints. The final stage of the process is represented by the Delivery object (R_r), which includes the shipment reference, the *m*-of-*n* attestation policy identifier (*attestationPolicyId*), the set of verifier signatures (*attestations*), and references to off-chain evidence files (*evidenceRefs*).

Each transaction is recorded together with a TransactionID and a digital signature, preventing conflicts among transactions and reinforcing the decentralized nature of the system. The data committed to the blockchain can be transparently audited and verified by all authorized stakeholders using blockchain explorer tools such as Hyperledger Explorer.

2.4.2. Off-Chain Data Structure and System Monitoring

Large-volume data that cannot be efficiently written directly to the blockchain, such as images and official reports, are stored on IPFS, while institutional documents are maintained in cloud storage. For each stored file, a SHA-256 hash value is computed and anchored on the blockchain, establishing a cryptographic linkage between the ledger and external storage systems. The IPFS service design concretizes this approach through the "CID + hash" pair. Upon uploading a file to IPFS, the resulting content identifier (cid_f) and the cryptographic hash of the file (h_f) are stored as an evidence reference ($EvindenceRef_f$). File integrity verification is based on comparing the on-chain hash with the hash of the file retrieved from IPFS, as formally expressed in Equation (7):

$$Verify(f) = \left[SHA256\Big(IPFS.get(cid_f)\Big) = h_f\right] \tag{7}$$

This model enables external data modifications to be detected through the blockchain with mathematical certainty. Similarly, log records (Log_i) generated within the system monitoring and reporting layer are collected on monitoring servers as a dataset (M) that includes service metadata, transaction latency ($latency_i$), and error states. The proposed SecureRelief architecture does not rely on the public IPFS network for evidence storage. Instead, IPFS is deployed as a permissioned, consortium-managed storage layer in which participating organizations operate dedicated IPFS nodes. Evidence artifacts generated during donation, shipment, and delivery processes are pinned according to predefined persistence policies, ensuring resistance against garbage collection and long-term availability without dependence on external pinning services. This design aligns with the permissioned governance model of the system while preserving organizational autonomy and accountability. While the current experimental deployment validates the feasibility of this approach, the SecureRelief architecture is explicitly designed to support multi-node IPFS clustering in production environments. In such a configuration, evidence artifacts are replicated across consortium-operated nodes, eliminating single-node dependency and ensuring continued availability even under node failures or partial network partitions.

2.5. Identity and Access Management

In critical and multi-actor environments such as disaster relief operations, not only the security of transactional data but also the identity and access privileges of participating actors must be rigorously controlled. Rather than relying on a single authentication or authorization mechanism, the proposed system adopts a layered security model. The security layers are configured as follows: (i) role- and organization-based controls at the application layer, (ii) JWT validation and request inspection at the API Gateway layer, (iii) WebAuthn/Passkey, DID/Verifiable Credential, and OTP verification at the Identity Service layer, and (iv) X.509/MSP-based identity enforcement and chaincode-level access controls at the blockchain layer. The IAM architecture is divided into two fundamental components: (i) Authentication, which enables actors to securely access the system, and (ii) Authorization, which determines which operations authenticated actors are permitted to perform within the system. The end-to-end operation of these components is illustrated in Figure 5.

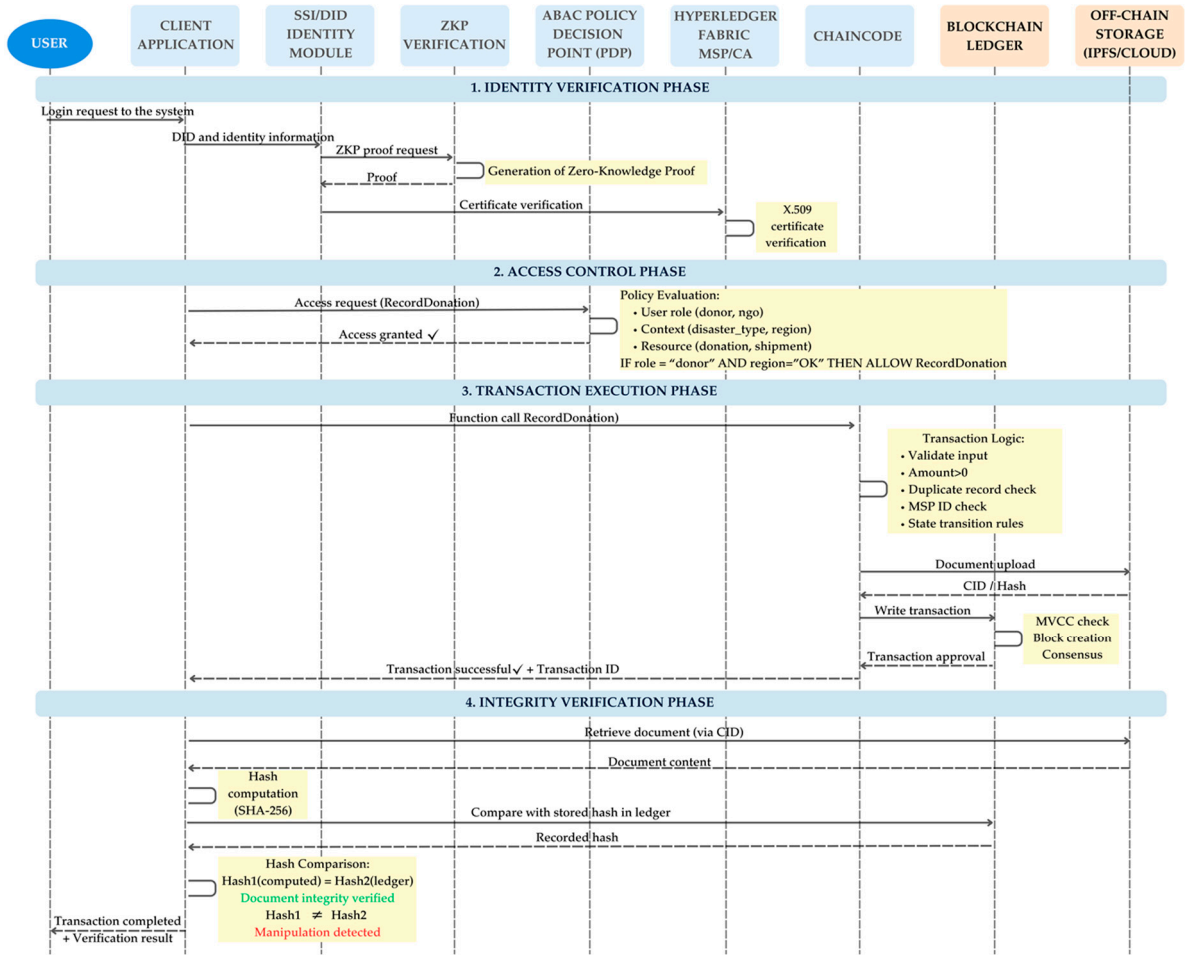


Figure 5. Sequence diagram illustrating the authentication and access control workflow.

2.5.1. User Authentication (SSI/DID + ZKP)

While preserving Hyperledger Fabric's X.509 certificate and MSP-based organizational identity model, the proposed system extends user-side identity management with Self-Sovereign Identity (SSI/DID) and Verifiable Credentials (VCs) to enhance privacy and usability. In the proposed model, the User Identity Structure (DID_u) consists of the user's decentralized identifier (ID_u), public key (pkupk_upku), organizational key ($orgKey_u$), and the set of verifiable credentials held by the user (VC_u). In addition, the MSP identity (mspIdumspId_umspIdu) is included in this structure to enable the verification of organizational affiliation at the blockchain layer. In privacy-sensitive scenarios, users employ Zero-Knowledge Proof (ZKP) protocols to prove specific attributes contained within their verifiable credentials (e.g., *"AidOrg member"*, *"field_agent role"*) without fully disclosing the underlying information. In the proposed architecture, Circom and snarkjs libraries are utilized for the generation of these proofs. The ZKP verification process is executed using the verification key (vk) and public inputs (x), as formally expressed in Equation (8).

$$Verify(vk, x, \pi) = 1 \tag{8}$$

Here, π denotes the anonymous proof generated by the user, while x represents the role or organizational evidence together with a nullifier value used to prevent replay attacks.

WebAuthn authentication in SecureRelief does not rely on dedicated external hardware security modules and can operate using platform authenticators available on common mobile devices, such as built-in secure enclaves or trusted execution environments. This design choice ensures broad device compatibility in disaster-stricken environments where

volunteers may use heterogeneous hardware. From a deployment perspective, device loss is treated as an expected operational risk rather than an exceptional failure condition in disaster scenarios. Accordingly, SecureRelief does not permanently bind user identities to a single physical authenticator. WebAuthn credentials are logically decoupled from the underlying blockchain identity and can be revoked and re-registered following organizational verification and re-enrollment procedures. This approach preserves identity continuity, auditability, and access recovery without compromising the integrity of on-chain records.

2.5.2. Authorization (ABAC Policies)

To determine which resources an authenticated user may access, the system employs an Attribute-Based Access Control (ABAC) model. ABAC decisions jointly evaluate the attributes embedded in the JWT (e.g., role, organization/MSP, region) and the contextual information of the request, such as the transaction type and the target resource. Whether a user u is permitted to perform an operation t on a resource r is determined by the policy function defined in Equation (9):

$$Permit(u, t, r) = P(A_u, t, r) \tag{9}$$

Here, (A_u) denotes the set of attributes associated with the user, while P represents the applicable policy set. At this stage, the API Gateway produces the policy decision, whereas the blockchain layer acts as the *final security gate* through MSP-based enforcement.

2.5.3. Cryptographic Traceability of Access Permissions

To ensure the auditability of authentication and authorization outcomes, the system records both successful and failed access decisions using service logs and, for critical operations, through a correlatable on-chain audit trail. Each Audit Trail object ($Audit_i$) includes the acting user (u_i), the performed action (*action*), the affected resource (resourceiresource_iresourcei), the system's decision, and the transaction timestamp and the associated digital signature (sig_i). This structure enables retrospective verification of who accessed which resource, with which attributes, and at what time.

2.6. Smart Contract Design and Algorithms

The business logic of the proposed SecureRelief platform is implemented through modular chaincode (smart contract) components developed in TypeScript and executed on the Hyperledger Fabric infrastructure. Instead of adopting a monolithic contract design, three interoperable contracts are defined to correspond to the fundamental stages of disaster relief operations, donation, logistics, and delivery, namely the Donation Chaincode, Shipment Chaincode, and Delivery Chaincode. These smart contracts not only validate transaction correctness but also operate in conjunction with the Identity and Access Management (IAM) layer to verify the identity of the initiating actor and the associated MSP (Membership Service Provider) membership. As a result, access to contract functions is strictly constrained by role- and authorization-based rules. To ensure scalability and limit on-ledger data volume, high-volume evidence artifacts (e.g., photos and documents) are stored on IPFS, while only their corresponding Content Identifiers (CIDs) and cryptographic hashes are recorded on the ledger, thereby preserving integrity verification and auditability.

2.6.1. Donation Record Smart Contract (DonationChaincode)

This module manages the donation lifecycle and constitutes the initial entry point for relief contributions within the system. For security reasons, only authenticated members of DonorOrg are permitted to create donation records, while state transitions are governed by AidOrg (NGO) and GovOrg (public authority) entities.

Algorithm 1 enforces deterministic identity (ID) checks to prevent duplicate entries and mitigates replay attack and unauthorized access risks by validating the MSP identity of the caller prior to transaction execution.

Algorithm 1. *CreateDonation()*: Create New Donation Record

Input: *UserContext (u), DonationData (D_in)*
Output: *DonationRecord (D_out)* **or Error**
// Initial Validation Checks
existing_record ← GetState(D_in.donationId)
if *existing_record* ≠ NULL **then**
    return Error("Duplicate donation ID detected")
end if
caller_msp ← GetClientMSPID(u)
if *caller_msp* ≠ "DonorOrgMSP" **then**
    return Error("Unauthorized: Only DonorOrg can create donations")
end if
// Donation Object Creation
timestamp ← GetTxTimestamp()
D_out ← {
    id: *D_in.donationId*,
    creator: *u.clientID*,
    org: *caller_msp*,
    status: "SUBMITTED",
    itemType: *D_in.itemType*,
    quantity: *D_in.quantity*,
    targetRegion: *D_in.targetRegion*,
    createdAt: *timestamp*,
    evidenceRefs: *D_in.evidenceRefs* }
// *Ledger Commit*
PutState(*D_out.id*, Serialize(*D_out*))
EmitEvent("DonationCreated", *D_out.id*)
return *D_out*

While Algorithm 1 formalizes the operational logic governing donation state updates, the overall donation lifecycle is more intuitively captured through its state transitions. Accordingly, the deterministic donation lifecycle and the authorized transitions between states are illustrated in Figure 6.

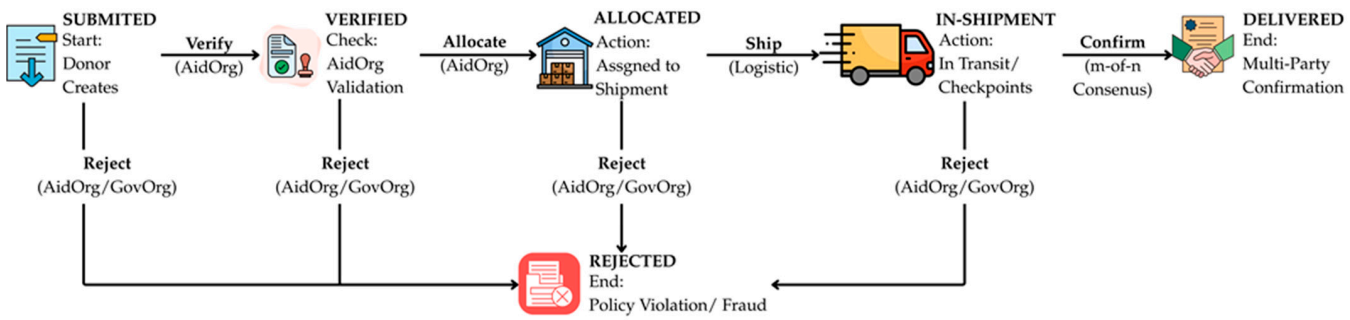


Figure 6. State transition diagram of the donation lifecycle managed by the smart contract.

As depicted in Figure 6, the donation object follows a deterministic lifecycle, starting from the SUBMITTED state and progressing through multi-party approval processes until

reaching either the DELIVERED or REJECTED state. Each transition is bound to explicit authorization rules and is triggered by designated actors. For instance, the transition from SUBMITTED to VERIFIED can only be performed through a valid signature issued by an authorized AidOrg. The authorization matrix and the logical checks enforcing these state transitions are formally defined in Algorithm 2.

Algorithm 2. *UpdateDonationStatus ()*: Donation State Management

Input: *DonationID (d_id), NewStatus (s_new), UserContext (u)*
Output: *UpdatedDonation (D)* **or Error**
1: $D \leftarrow$ GetState(*d_id*)
2: **if** D = NULL **then return Error**("Donation not found")
3: *current_status* $\leftarrow$ *D.status*
4: *caller_msp* $\leftarrow$ GetClientMSPID(*u*)
5: *is_allowed* $\leftarrow$ **FALSE**
6: //State Transition Checks
7: **if** (*current_status* = "SUBMITTED") $\wedge$ (*s_new* = "VERIFIED") $\wedge$ (*caller_msp* = "AidOrgMSP") **then**
8: *is_allowed* $\leftarrow$ **TRUE**
9: **else if** (*current_status* = "VERIFIED") $\wedge$ (*s_new* = "ALLOCATED") $\wedge$ (*caller_msp* = "AidOrgMSP") **then**
10: *is_allowed* $\leftarrow$ TRUE
11: **else if** (*s_new* = "REJECTED") $\wedge$ (*caller_msp* $\in$ {"AidOrgMSP", "GovOrgMSP"}) **then**
12: *is_allowed* $\leftarrow$ **TRUE**
13: **end if**
14: **if** *is_allowed* = **FALSE then**
15: **return Error**("Invalid state transition or unauthorized organization")
16: **end if**
17: //State Update and Ledger Commit
18: *D.status* $\leftarrow$ *s_new*
19: PutState(*d_id*, Serialize(*D*))
20: EmitEvent("DonationStatusUpdated", {*d_id*, *s_new*})
21: **return** *D*

2.6.2. Logistics and Shipment Smart Contract

This smart contract is designed to make the physical transfer process of donations digitally traceable and enables one or more donations (*donationIds*) to be aggregated under a single shipment. Algorithm 3 summarizes the shipment creation process. A critical aspect of this process is the validation of the donations included in the shipment, ensuring that each donation is valid and has not been previously assigned to another shipment.

Algorithm 3. *CreateShipment()*: Create Shipment Record

Input: *ShipmentID (s_id), DonationIDs List (L_don), Route (r), UserContext (u)*
Output: *ShipmentRecord (S)* **or Error**
if IsEmpty(*L_don*) **then**
    return Error("At least one donation ID is required")
end if
// Initialize Shipment Object
S ← {
    id: *s_id*,
    donations: *L_don*,
    carrier: GetClientMSPID(*u*), //LogisticsOrg or AidOrg
    route: *r*,
    status: "CREATED",
    checkpoints: ∅, //An empty list is started
    timestamp: GetTxTimestamp()
}
PutState(*s_id*, Serialize(*S*))
return *S*

The transparency of the logistics process is ensured through data entries recorded at route checkpoints. Algorithm 4 illustrates the AddCheckpoint function, which is triggered by QR code scanning or manual input in the field. This function immutably records the shipment's location and timestamp on the blockchain.

Algorithm 4. *AddCheckpoint ()*: Route Tracking

Input: *ShipmentID (s_id), Location (loc), Status (stat), Notes (n), UserContext (u)*
Output: *UpdatedShipment (S)*
S ← GetState(*s_id*)
cp_id ← GenerateUUID()
// New Checkpoint Object
new_cp ← {
    id: *cp_id*,
    location: *loc*,
    status: *stat*,
    recorder: *u.clientID*,
    timestamp: GetTxTimestamp(),
    notes: *n* }
// Append to List and Status Update
S.checkpoints ← *S.checkpoints* ∪ {*new_cp*}
S.updatedAt ← GetTxTimestamp()
S.status ← *stat*
PutState(*s_id*, Serialize(*S*))
return *S*

2.6.3. Delivery Verification Smart Contract

This module, which documents that relief materials have reached their final destination, constitutes one of the most critical security components of the system. Algorithm 5 is designed to minimize the on-chain data footprint. While delivery evidence is stored on

IPFS, the smart contract records only the file fingerprint *(CID + hash)* as an immutable commitment on the blockchain.

Algorithm 5. *CreateDelivery()*: Create Delivery Record

Input: *DeliveryID (del_id), ShipmentID (s_id), EvidenceRefs (E_refs)*
Output: *DeliveryRecord (DR)* **or Error**

```
S ← GetShipmentState(s_id)
// Precondition: Shipment Must Have Arrived
if S.status ≠ "ARRIVED" then
    return Error("Shipment has not arrived yet")
end if
policy ← GetPolicy("DEFAULT_2_OF_3")
DR ← {
    id: del_id,
    shipmentId: s_id,
    evidence: E_refs,                    //IPFS CIDs + Hashes
    status: "PENDING_CONFIRMATION",
    attestations: ∅,
    policy: policy,
    createdAt: GetTxTimestamp() }
PutState(del_id, Serialize(DR))
return DR
```

2.6.4. Multi-Party Attestation Mechanism

The validation of deliveries through stakeholder consensus rather than a single authority constitutes the core of the system's trust model. Algorithm 6 illustrates the mechanism implemented within the DeliveryChaincode, which enforces the m-of-n attestation logic, requiring approval from at least m out of n authorized participants. In the default policy, delivery confirmation requires a minimum of two signatures in a three-organization scenario (*AidOrg*, *GovOrg*, *LogisticsOrg*), corresponding to a 2-of-3 threshold. Additionally, the policy may enforce mandatory approval from specific organizations (e.g., GovOrg) to satisfy regulatory or governance requirements.

The delivery attestation procedure does not result in a deadlock in the absence of signatures from all required organizations. Instead, the process operates within a bounded waiting window defined at the smart contract level. If one or more entities listed in RequiredOrgs fail to submit a valid signature within the predefined attestation interval, the delivery verification process is deterministically terminated and marked as incomplete. This fail-safe termination mechanism prevents indefinite blocking while preserving the integrity of the verification workflow. In particular, the absence of mandatory institutional signatures (e.g., from a government authority) cannot be bypassed through unilateral approvals, ensuring that delivery confirmation remains strictly compliant with governance constraints. In the current prototype, this bounded waiting behavior is enforced at the verification logic level rather than through a globally enforced delivery deadline, reflecting a deliberate design choice that favors governance correctness and accountability over forced liveness under exceptional conditions. This design choice allows deadline policies to be introduced at the governance layer without requiring modifications to the core attestation logic.

Algorithm 6. *AttestDelivery()*: *m*-of-*n* Approval System

Input: *DeliveryID (d_id), Decision (vote), UserContext (u)*
Output: *UpdatedDelivery (DR)* **or Error**
1: *DR* ← GetState(*d_id*)
2: **if** *DR.status* ≠ "PENDING_CONFIRMATION" **then**
3: **return Error**("Delivery is already finalized")
4: *end if*
5: **voter_org** ← GetClientMSPID(*u*)
6: // Authorization and Duplicate Vote Check
7: **if** *voter_org* ∉ *DR.policy.eligibleOrgs* **then**
8: **return Error**("Organization not eligible to vote")
9: **end if**
10: **if** HasAlreadyVoted(*DR.attestations, voter_org*) **then**
11: **return Error**("Organization has already voted")
12: **end if**
13: // Recording the Vote
14: *new_attestation* ← { *attestor*: *voter_org*, *decision*: vote, *time*: Now() }
15: *DR.attestations* ← *DR.attestations* ∪ {*new_attestation*}
16: // Consensus Computation
17: *count_approved* ← Count(*DR.attestations*, where *decision* = "APPROVED")
18: *count_rejected* ← Count(*DR.attestations*, where *decision* = "REJECTED")
19: *required_met* ← CheckRequiredOrgs(*DR.attestations*, *DR.policy.requiredOrgs*)
20: // Decision Logic
21: **if** *count_rejected* > 0 **then**
22: *DR.status* ← "REJECTED" //A single rejection invalidates the operation (fail-safe)
23: EmitEvent("DeliveryRejected", *d_id*)
24: **else if** (*count_approved* ≥ *DR.policy.M*) ∧ (*required_met* = **TRUE**) **then**
25: *DR.status* ← "CONFIRMED"
26: EmitEvent("DeliveryConfirmed", *d_id*)
27: **else**
28: EmitEvent("AttestationAdded", *d_id*)//The required majority has not yet been reached
29: **end if**
30: PutState(*d_id*, Serialize(*DR*))
31: **return** *DR*

These algorithms provide a deterministic structure that minimizes human error in disaster relief operations, prevents unauthorized interventions, and guarantees the correctness of deliveries through cryptographic evidence.

2.7. Consensus, Validation, and Transaction Security

The proposed SecureRelief platform adopts a multi-layered verification architecture to ensure data consistency and system security in critical and time-sensitive disaster relief processes. This architecture comprises the deterministic consensus mechanism of Hyperledger Fabric (HLF), validation stages throughout the transaction lifecycle, and network-level security protocols.

2.7.1. Consensus Approach for Permissioned Networks

To meet the low-latency and high-throughput requirements that are critical in disaster scenarios, the proposed system employs the Raft-based Ordering Service component of Hyperledger Fabric. Unlike mining-based approaches such as Proof of Work (PoW), Raft achieves consensus among nodes without incurring computational waste, thereby enabling the system to scale efficiently under crisis conditions without becoming a performance bottleneck. The primary reasons for adopting Raft in a permissioned network context

are as follows: (i) the participant set is controlled, as inter-organizational identities are defined through MSP/X.509 certificates; (ii) the ordering layer produces a deterministic transaction sequence by establishing a single, total order of transactions on the channel; and (iii) transaction finality becomes effectively instantaneous once blocks are committed to the ledger. This design aligns with the requirements of transaction certainty and accountability in disaster relief logistics.

2.7.2. Transaction Validation Process

In SecureRelief, transaction security is grounded in Fabric's Execute–Order–Validate paradigm and the formally defined transaction validity conditions of the architecture. A transaction is accepted only if it satisfies three requirements: (1) authentication, (2) authorization, and (3) data integrity. Conceptually, the transaction pipeline can be summarized in four stages:

1. *Client/Application Pre-validation*: User identity is verified through the Identity Service, while the API Gateway enforces JWT validation and request inspection, thereby filtering malformed or unauthorized calls before they reach the blockchain. This pre-validation stage contributes significantly to system robustness under operational load.
2. *Endorsement (Execution/Simulation)*: The client invokes the relevant chaincode function, and endorsing peers simulate the transaction to generate a read–write set (RW-set), producing a signed endorsement response.
3. *Ordering (Consensus/Sequencing)*: The Ordering Service deterministically sequences incoming transactions into blocks on the channel using the Raft protocol.
4. *Validation and Commit*: Peers validate (i) endorsement policy compliance, (ii) read–write conflicts (MVCC/read-set consistency), and (iii) access and policy rules. Valid transactions are then committed to the world state and the blockchain ledger. Upon completion of this stage, the transaction becomes auditable on the ledger.

This transaction pipeline strengthens inter-organizational ledger consistency and traceability in the design where donation–shipment–delivery processes progress over a single shared channel.

2.7.3. Security Layers and Protection Mechanisms

SecureRelief adopts a layered security architecture that does not rely on a single control point, extending from the application layer down to the blockchain layer. Security controls are structured across four levels: (i) application-level RBAC and organization-based controls, (ii) JWT validation and rate limiting at the API Gateway, (iii) WebAuthn/Passkey, DID/VC, and OTP verification at the Identity Service, and (iv) X.509/MSP and chaincode-level access controls at the blockchain layer. These layers collectively provide the following security guarantees against common attack classes:

1. *Unauthorized Transaction Initiation (Impersonation/Unauthorized Access)*: WebAuthn/Passkey authentication and DID/VC verification strengthen user identity assurance, while MSP identity validation acts as the final security gate at the blockchain layer.
2. *Replay Attacks*: The use of nullifiers in ZKPs and transaction-level uniqueness (ID uniqueness) checks prevent proof and transaction reuse.
3. *Data Integrity Violations (Off-chain Tampering)*: While evidence files are stored on IPFS, anchoring CID and hash references on-chain enables the detection of any off-chain data manipulation.
4. *Communication Security (MITM/Eavesdropping)*: Transport-layer protections are enforced via TLS on the client–API Gateway channel and mutual TLS (mTLS) on Fabric access paths.

5. *Transaction-level Privilege Escalation*: MSP-based organization checks and state transition rules enforced within chaincode prevent unauthorized MSPs or invalid state transitions from being committed to the ledger.

This approach ensures that IAM and ABAC decisions are also enforced at the blockchain layer, preventing security policies from being confined solely to the application tier.

2.7.4. Operational Efficiency and Auditability

In disaster scenarios, operational visibility and auditability are as critical as security. In SecureRelief, performance and health metrics are monitored using Prometheus and Grafana, while block- and transaction-level audits are visualized through Hyperledger Explorer, providing accountability, particularly for auditing actors such as GovOrg. This layer supports (i) early warning through metrics such as transaction latency and error rates, (ii) retrospective incident analysis via immutable ledger records, and (iii) clear attribution of responsibilities across organizations (non-repudiation).

2.8. Threat Model and Security Assumptions

SecureRelief operates on a permissioned Hyperledger Fabric network, executing the donation–shipment–delivery workflow over a single channel (reliefchannel) and three modular chaincodes (Donation, Shipment, and Delivery). Rather than relying solely on on-chain validation, security is addressed through a layered approach spanning the application, API, identity, and blockchain layers. This section classifies threats according to system components and then defines the system's security assumptions and monitoring mechanisms.

2.8.1. Threat Model

The proposed threat model encompasses (i) external attackers, (ii) compromised client or service components, and (iii) malicious or negligent internal actors (organizational users). These actors primarily aim to (a) initiate unauthorized transactions or escalate privileges, (b) manipulate off-chain evidence to falsify delivery verification, (c) violate privacy by exposing sensitive data, and (d) disrupt service availability.

*T*1—*Account/Session Compromise*: Compromise of client-side authentication may allow attackers to issue API calls on behalf of authorized users. This risk is mitigated through multi-factor mechanisms such as WebAuthn/Passkey and OTP, combined with JWT-based session management.
*T*2—*Privilege Escalation and Role Abuse*: Bypassing application-level role checks or losing security context in inter-service calls may result in unauthorized state transitions. This threat is mitigated by MSP-based organizational validation on-chain and strict enforcement of state transition rules within chaincode.
*T*3—*Replay Attacks*: Reusing previously generated proofs, particularly in privacy-preserving verification steps, undermines security guarantees. Replay risks are mitigated through nullifier-based mechanisms in ZKP outputs.
*T*4—*Off-chain Evidence Tampering*: Since evidence files are stored on IPFS, content modification or substitution could mislead delivery verification. This risk is mitigated by binding evidence through CID and SHA-256 hashes and validating them during verification. The IPFS service is explicitly defined as generating CID/hash pairs and exposing gateway URLs, with both values returned by the evidence upload API.
*T*5—*Communication Channel Attacks (MITM/Eavesdropping*): Network-level attacks are mitigated using TLS 1.3 for frontend-to-API communication and gRPCS (mTLS) for API-to-

Fabric interactions. As IPFS Gateway access is defined over HTTP, integrity verification via hash comparison is prioritized over confidentiality.
T6—Availability and DoS: Excessive request loads or service failures during disaster conditions may disrupt delivery verification. Accordingly, rate limiting and request validation at the API layer, along with infrastructure-level health monitoring, are considered essential controls.

2.8.2. Security Assumptions

SecureRelief operates under the Crash Fault Tolerance (CFT) guarantees provided by Hyperledger Fabric's Raft-based consensus mechanism. The fault model tolerates node crashes rather than Byzantine adversarial behavior. Under this technical foundation, the system's security guarantees rely on the following assumptions:

A1—Trusted Permissioned Identity Infrastructure: MSP/X.509-based organizational identities assume that the Fabric CA issues correct certificates and that private keys are securely managed by organizations.
A2—Client Device Security: Passkey/WebAuthn key material is assumed to be protected by platform-supported security mechanisms (e.g., Secure Enclave or Trusted Execution Environments) when available. The security model does not assume exclusive dependence on a single physical device, as identity recovery and credential re-registration are supported through organizational verification mechanisms.
A3—Verifiable Off-chain Storage: IPFS content is not assumed to be confidential by default; however, integrity is cryptographically verifiable through matching on-chain CID and SHA-256 hash values ("untrusted but verifiable" storage).
A4—Time Synchronization: Nodes are assumed to maintain reasonable time synchronization (e.g., via NTP) to ensure consistent expiration checks and certificate validity.

2.8.3. Security Monitoring and Auditing Mechanisms

In SecureRelief, security is supported not only through prevention but also through detection and response capabilities. The monitoring and auditing layer integrates Prometheus and Grafana for real-time observation of resource usage and latency, while Hyperledger Explorer visualizes block and transaction details. Technical documentation defines Grafana dashboards (API Gateway Metrics, Fabric Network Status, Service Health, Request Latency) and Prometheus alerts (service down, high latency, error spikes, low disk space). Centralized service logs further support incident response and forensic analysis. From an auditability perspective, GovOrg and similar auditing actors are explicitly granted monitoring and audit roles within the authorization matrix. Finally, a public verification endpoint provides external stakeholders with a verifiable summary of the donation–shipment–delivery chain, reinforcing trust and transparency.

3. Results

This section evaluates the end-to-end operation of the SecureRelief platform through measurable outcomes. The assessment covers the complete workflow of the permissioned Hyperledger Fabric network integrated with IPFS-based off-chain evidence storage and privacy-preserving mechanisms enabled by SSI/DID and zero-knowledge proofs. The evaluation is structured along three main dimensions: (i) the performance of on-chain donation, shipment, and delivery processes, (ii) the additional latency and computational overhead introduced by privacy-preserving mechanisms, and (iii) the efficiency gains achieved by offloading large evidence files to off-chain storage. Measurements have been conducted using scenario-based workload profiles. Fabric-specific metrics such as transaction throughput, transaction latency with a focus on the 95th percentile, success

rate, and MVCC conflict rate have been systematically monitored. In addition, resource utilization including CPU, memory, and network usage has been observed through the Prometheus–Grafana monitoring stack in order to correlate performance results with underlying infrastructure behavior.

3.1. Experimental Setup

The performance efficiency, scalability, and transaction latency of the SecureRelief platform have been evaluated in an isolated laboratory environment designed to simulate real-world disaster response scenarios. The experimental setup has been configured to reflect the hybrid architecture of the proposed system, combining on-chain and off-chain components together with multi-layer security mechanisms such as zero-knowledge proofs and WebAuthn-based authentication. The operational status of the deployed system and inter-service data flows have been continuously verified through real-time monitoring dashboards.

Hardware and Virtualization Infrastructure: Experimental evaluations have been conducted in a Docker-based environment allocated with 6 GB of RAM and 8 vCPUs. To realistically emulate the distributed nature of the system, network isolation and container orchestration have been managed using Docker version 24.0 and Docker Compose. Each organization and microservice has been deployed within isolated virtual network segments with explicitly defined resource constraints, thereby reproducing a realistic multi-organizational network topology.

Software Stack and Network Topology: The core software components of the platform have been configured using the latest stable long-term support versions. The blockchain layer operates on Hyperledger Fabric (2.5.x), while smart contracts have been implemented in TypeScript and executed on the Node.js version 18.x runtime. The experimental network consists of four distinct organizations representing the disaster management ecosystem, namely *AidOrg*, *DonorOrg*, *GovOrg*, and *LogisticsOrg*. Detailed hardware and software configuration parameters of the experimental environment are summarized in Table 3.

Table 3. Experimental test environment components and version information.

Component	Technology/Specification	Version/Details
Host CPU	Apple Silicon (ARM64)	M-Series (4+ Cores)
Host RAM	LPDDR5	16 GB
Blockchain Framework	Hyperledger Fabric	v2.5.x
Consensus Protocol	Raft (Crash Fault-Tolerant)	Etcd-based
State Database	CouchDB	v3.3.x
Smart Contract Lang.	TypeScript	Node.js Runtime
Off-Chain Storage	IPFS (InterPlanetary File System)	Kubo v0.39
Privacy (ZKP)	Circom & Snarkjs	Groth16/Plonk
Containerization	Docker Engine & Compose	v24.x+
Monitoring	Prometheus & Grafana	v2.45/v10.0

The experiments have been conducted by applying progressively increasing transaction loads per second to the three smart contracts deployed on the *reliefchannel*. The successful deployment of these on-chain components and the corresponding smart contract version states have been verified through the system's administrative dashboard. Figure 7 illustrates the deployment status and version information of the active chaincodes operating within the multi-organization network.

As shown in Figure 7, Fabric peer nodes associated with *GovOrg*, *AidOrg*, *DonorOrg*, and *LogisticsOrg* have been actively running, confirming the concurrent execution of multiple organizations within the network. In addition, auxiliary services critical to the hybrid architecture, including the ipfs-node for off-chain data storage and integrity assurance, and

the identity-pg service for identity management, have been operational. The management dashboard has verified the active deployment of instantiated chaincodes (v1.0), thereby confirming the readiness, consistency, and correctness of the overall system environment as described in the experimental setup.

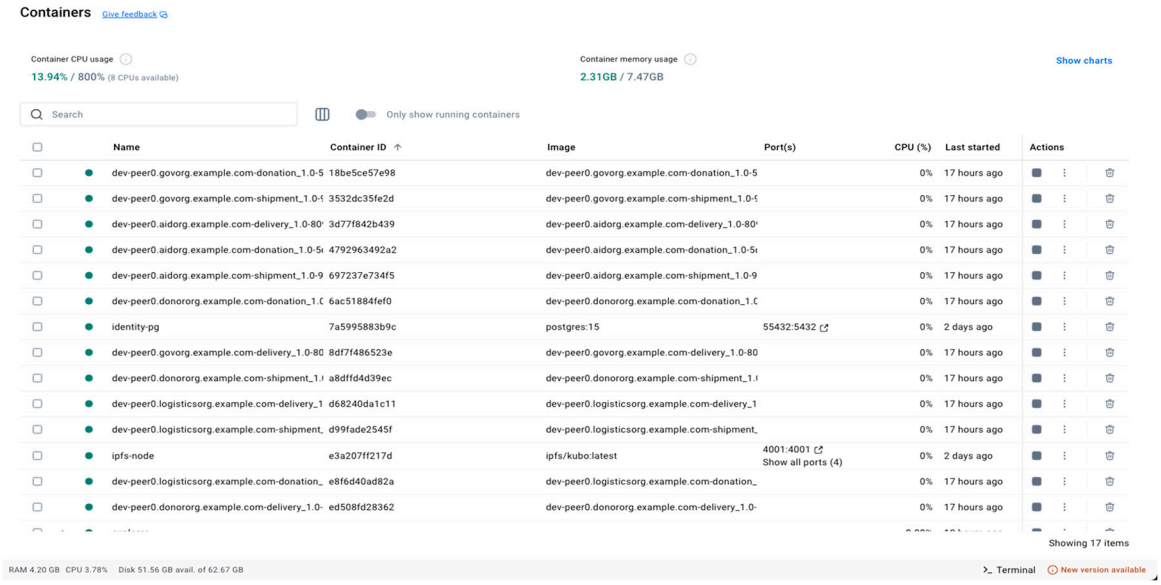


Figure 7. Operational status of the containerized infrastructure of the proposed system.

Block Parameters and Data Management: To optimize system throughput and latency performance, a Raft-based (EtcdRaft) Ordering Service has been employed, and block formation parameters have been configured as follows: a maximum of 10 transactions per block, a block size of 2 MB, and a batch timeout of 2 s. For off-chain data management, large-scale evidence files have been stored on IPFS nodes running Kubo v0.39, while relational data have been maintained on PostgreSQL v15.

Data Integrity and Active Organizational Participation: System data integrity and the active participation of organizations in the network have been examined using the Hyperledger Explorer tool (version 2.0.0). Figure 8 illustrates block production on the network, overall transaction throughput, and the distribution of transactions by organization. The visualizations have confirmed that all four organizations, namely *AidOrg*, *DonorOrg*, *GovOrg*, and *LogisticsOrg*, have actively participated in the consensus process.

Test Scenarios and Monitoring: For performance evaluation, a Prometheus and Grafana-based monitoring stack has been integrated into the system. During the experiments, system behavior has been observed under multiple scenarios, including donation creation, shipment tracking, and the computation-intensive delivery confirmation with ZKP. The operational health of the system and the real-time status of microservices have been continuously monitored through centralized dashboards. Figure 9 illustrates request traffic on the API Gateway, response latencies, and service health status while the system has been operating under load. The visualizations have confirmed that the error rate has remained at 0% throughout the experiments and that the system has operated in a stable manner.

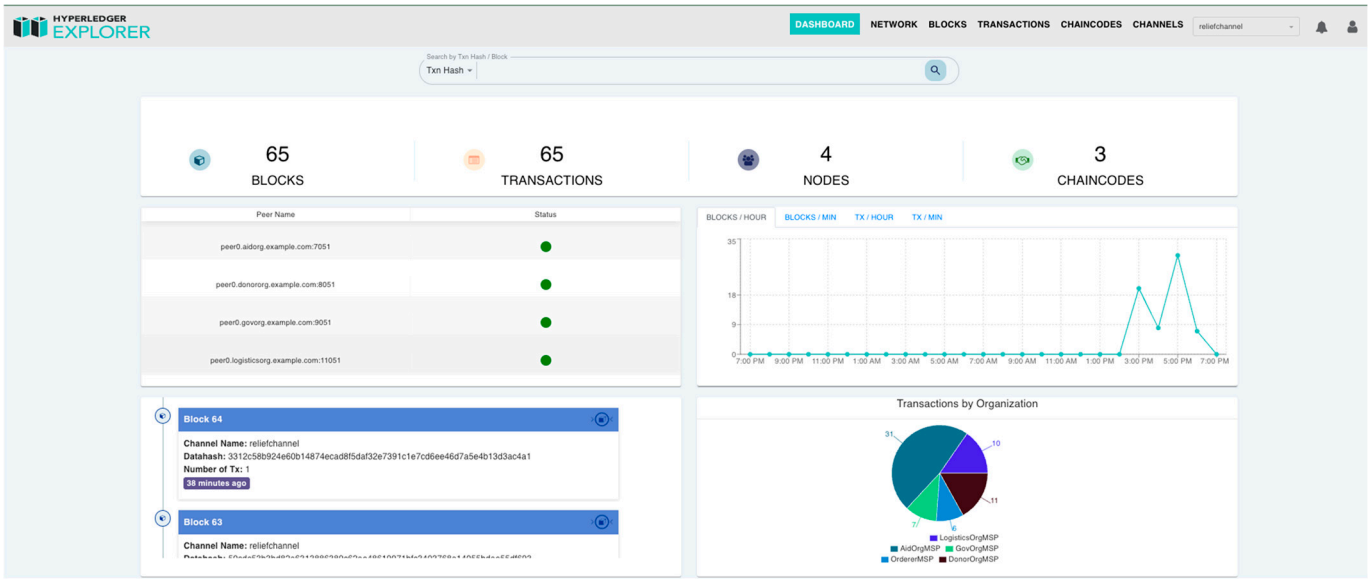


Figure 8. Network activity and ledger visualization using Hyperledger Explorer. The dashboard has confirmed the successful initialization of the relief channel with four active peers and three deployed chaincodes. The "Transactions by Organization" chart shown in the lower-right corner has verified multi-party participation, indicating that *AidOrg*, *DonorOrg*, *GovOrg*, and *LogisticsOrg* have actively recorded transactions on the ledger.

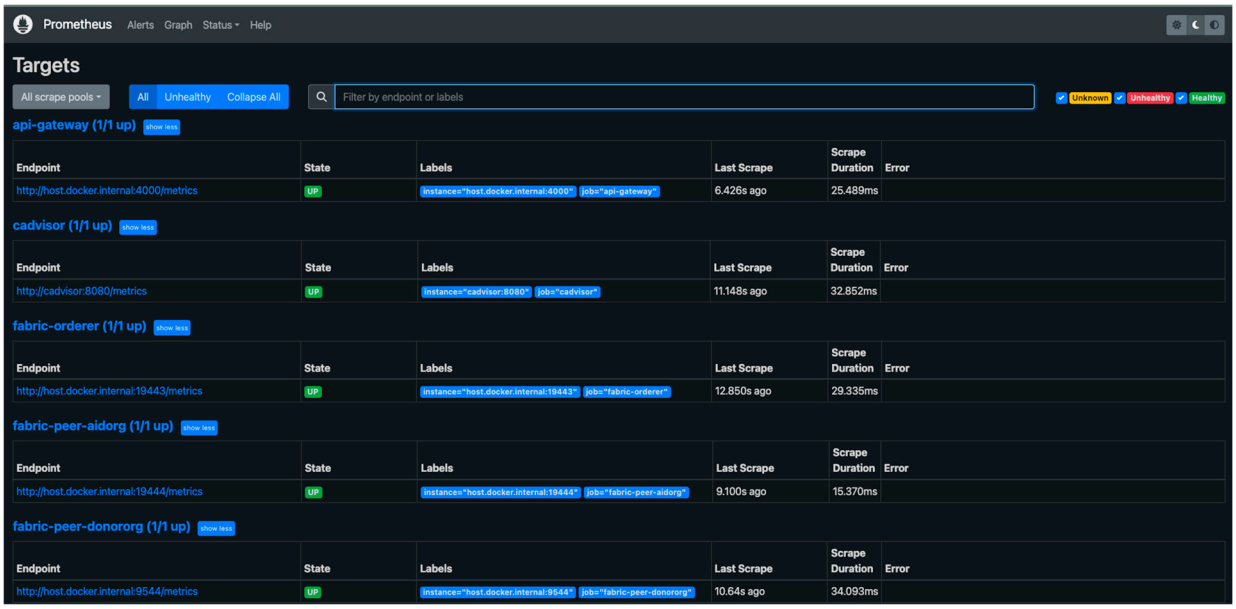


Figure 9. Real-time monitoring dashboard visualizing system health and performance metrics. The panels display API request processing rates in the upper-left, low-latency response times in the upper-right, and the active status of core services indicated by green bars. The zero error rate has confirmed the reliability of the system under the applied experimental workload.

3.2. Blockchain Performance Analysis

The performance of the blockchain layer of the SecureRelief platform has been evaluated under scenario-based workloads using the experimental environment described in Section 3.1. Measurements have been collected for on-chain transactions generated throughout the donation–shipment–delivery lifecycle, focusing on Fabric-specific metrics such as transaction throughput, transaction latency with particular emphasis on the 95th percentile, success rate, and MVCC conflict rate. In addition, infrastructure-level resource utilization

has been obtained through Prometheus–Grafana dashboards. Figure 10 illustrates the transaction behavior and ledger height of the Hyperledger Fabric network, while the request traffic and response latencies at the API Gateway layer.

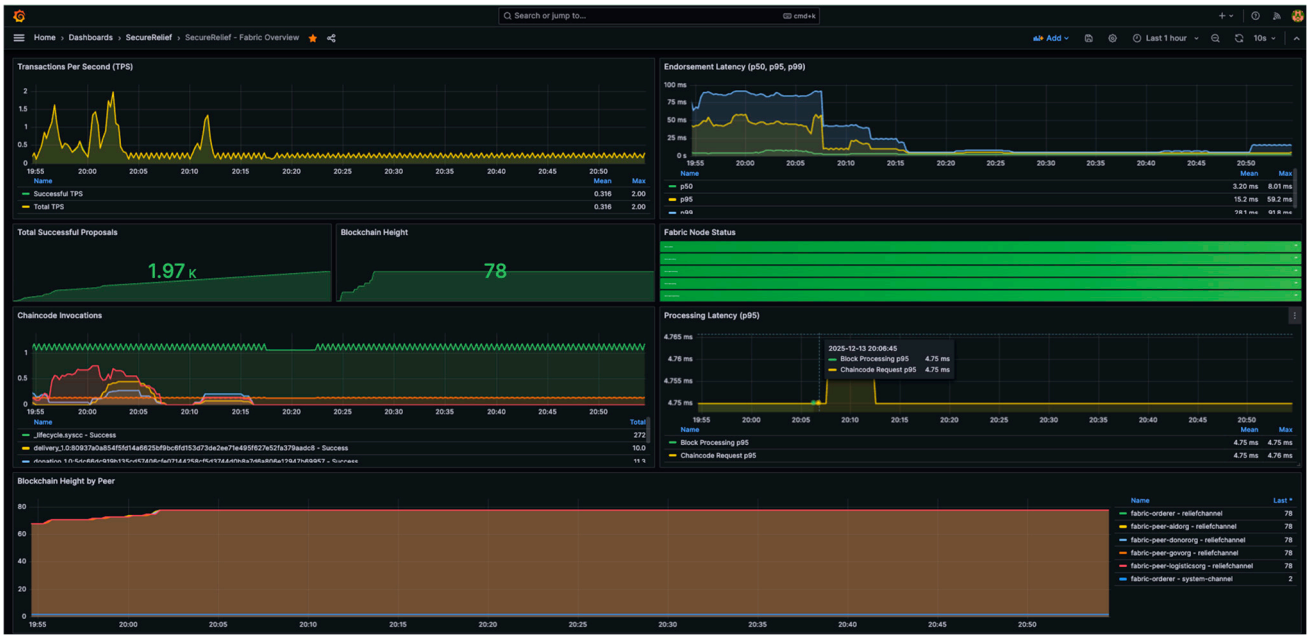


Figure 10. Observed throughput under conservative and sequential transaction submission settings. The green curve represents the endorser successful proposal rate used for TPS calculation, while the red and orange curves correspond to delivery and shipment transaction flows, respectively.

3.2.1. Transaction Throughput

The transaction throughput values reported in this study reflect the observed operational behavior of the SecureRelief platform under the defined experimental scenarios, rather than the theoretical or maximum performance capacity of the underlying Hyperledger Fabric network. As illustrated in Figure 10 (Transactions Per Second), the blockchain network exhibited a stable average observed throughput of approximately 0.32 transactions per second (TPS) throughout the evaluation period, with short-term peak values reaching approximately 2 TPS during brief intervals of increased activity. The reported average throughput value has been derived from the endorser_successful_proposals metric monitored via the Grafana Fabric Overview panel, using a one-minute rolling average. This measurement approach inherently captures periods of low activity in which the system remains idle between event-driven operations, such as donation creation, shipment updates, and delivery attestations. Consequently, the resulting average TPS value reflects the intermittent and scenario-driven nature of the disaster relief workflow, rather than sustained operation under saturation conditions. In this context, the observed 0.32 TPS value does not represent the network's achievable capacity under load, but instead characterizes an event-based execution model in which transactions are triggered irregularly and sequentially in response to real-world operational events. Similarly, the observed peak value of approximately 2 TPS reflects short-lived transaction clustering and does not indicate an upper bound of the Hyperledger Fabric infrastructure. All transactions have been submitted sequentially under conservative experimental settings designed to preserve deterministic execution, multi-party attestation correctness, and privacy-preserving verification guarantees across donation, shipment, and delivery stages. Across the experiment, approximately 1.97 k successful transaction proposals have been generated, ordered into blocks, and committed to the ledger by the Raft-based ordering service. The observed throughput behavior is consistent with the block configuration parameters

defined in Section 3.1, including a batch timeout of 2 s, a maximum message count of 10 transactions per block, and a maximum block size of 2 MB. Under these parameters, the system demonstrated a predictable and stable transaction processing profile suitable for medium-scale disaster management scenarios where correctness, auditability, and trust assurance are prioritized over raw throughput. To further contextualize these observations, the intrinsic throughput capacity of the Hyperledger Fabric layer has been evaluated separately under controlled load conditions, independent of the end-to-end SecureRelief workflow and its associated security and privacy components. The results of this isolated benchmark analysis are presented in a subsequent subsection, enabling a clear distinction between workload-dependent platform behavior and infrastructure-level performance characteristics.

3.2.2. Isolated Throughput and Stress Benchmark

To complement the workload-driven throughput analysis presented in Section 3.2.1, an isolated benchmark evaluation has been conducted to assess the intrinsic throughput capacity and saturation behavior of the underlying Hyperledger Fabric network under controlled load conditions. In this benchmark setting, end-to-end workflow characteristics and security-related overheads specific to the SecureRelief platform, such as WebAuthn-based authentication, ABAC enforcement, zero-knowledge proof generation and verification, and off-chain service interactions, have been excluded in order to focus exclusively on ledger-level performance. The benchmark has been performed on a Fabric network configured consistently with the experimental setup used throughout the study, comprising four organizations, a MAJORITY endorsement policy, TLS-enabled communication, and a Raft-based ordering service. Transaction load has been gradually increased by varying the target send rate, allowing both sustainable throughput and overload behavior to be observed. Table 4 reports the benchmark results obtained under increasing target transaction rates. The results indicate that the network is capable of sustaining commit throughput values of up to approximately 200 TPS with a 100% transaction success rate. Within this operating range, average transaction latency remains below 100 ms up to approximately 150 TPS, after which latency begins to increase as the network approaches saturation. Beyond this threshold, a sharp increase in latency and a rapid decline in success rate are observed, indicating the onset of saturation effects and bottlenecks at the gateway and ordering layers. The pronounced latency increase observed at the upper boundary of the isolated benchmark (Table 4, target rate $\approx$ 200 TPS) can be attributed to the interaction between the transaction arrival rate and the block formation parameters of the Fabric network. Specifically, with a batch timeout of 2 s and a maximum message count of 10 transactions per block, the effective sustainable throughput approaches a practical limit near 200 TPS. As this limit is reached, newly arriving transactions continue to be enqueued while block formation remains bounded by these parameters, leading to transient queue buildup and elevated latency. This behavior reflects a controlled saturation effect rather than a system malfunction, analogous to congestion forming at the capacity limit of a traffic system. While Table 4 characterizes the sustainable throughput region under controlled load conditions, it does not capture the system behavior once the transaction arrival rate exceeds the stable operating range. To analyze the network response under overload conditions, a stress-oriented evaluation has been conducted by progressively increasing the transaction injection rate beyond the saturation threshold.

Table 4. Isolated throughput benchmark results of the Hyperledger Fabric network under controlled load conditions (100% success rate up to 200 TPS).

Target TPS	Transactions	Throughput (TPS)	Avg. Latency (ms)	Max. Latency (ms)	Success Rate (%)
20	1200	20.1	270	570	100
40	2400	40.1	160	470	100
60	3600	60.1	120	220	100
80	4800	80.1	100	170	100
100	6000	100.1	90	170	100
150	9000	150.0	80	290	100
200	6000	199.3	480	1570	100

Figure 11 further illustrates the stress test results by jointly depicting achieved throughput and transaction success rate as a function of the target send rate. While the achieved throughput continues to increase beyond 200 TPS, the corresponding success rate degrades significantly, falling below 40% at higher load levels. This behavior highlights a clear trade-off between throughput and reliability under overload conditions and provides an empirical upper bound on sustainable operation for the evaluated configuration. Taken together, these benchmark results demonstrate that the relatively low average throughput values reported in Section 3.2.1 are not attributable to limitations of the Hyperledger Fabric infrastructure. Instead, they arise from the event-driven, sequential, and conservatively configured execution model adopted for SecureRelief to prioritize correctness, auditability, and governance compliance in disaster relief workflows. The isolated benchmark analysis thus provides a complementary perspective, clearly separating infrastructure-level capacity from application-level operational behavior. To provide a quantitative breakdown of the observed saturation behavior, a detailed stress test analysis has been performed. The resulting metrics, including transaction success rate, achieved throughput, and latency escalation, are summarized in Table 5. In the isolated benchmark (Table 4), latency is reported in milliseconds, while in the stress test (Table 5) latency is expressed in seconds, owing to the order-of-magnitude increase observed as the network enters the overload regime.

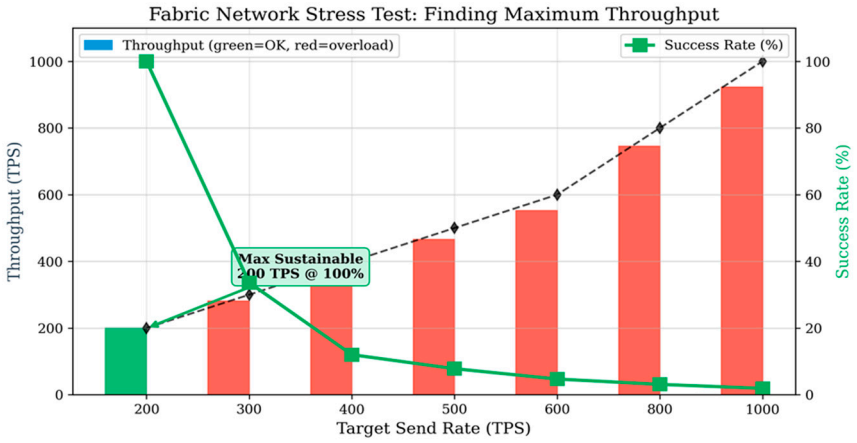


Figure 11. Stress test results showing achieved throughput and transaction success rate under increasing target send rates, highlighting the maximum sustainable throughput and saturation behavior of the Fabric network.

Table 5. Stress test results under increasing transaction load, showing throughput saturation, latency escalation, and graceful degradation behavior of the Fabric network.

Target TPS	Submitted	Successful	Fail	Throughput (TPS)	Avg. Latency (s)	Success Rate (%)	Status
200	6000	6000	0	199.3	0.48	100.0	Sustainable
300	9000	3027	5973	280.6	4.72	33.6	Performance Degradation
400	12,000	1441	10,559	363.7	9.94	12.0	Network Overload
500	15,000	1175	13,825	465.4	10.29	7.8	Network Overload
600	18,000	848	17,152	551.6	13.46	4.7	Network Overload
800	24,000	741	23,259	745.1	12.85	3.1	Network Overload
1000	30,000	565	29,435	922.6	11.16	1.9	Network Overload

To further characterize system behavior beyond the sustainable operating range, an additional stress test has been conducted by increasing the transaction injection rate well above the maximum stable throughput. Table 5 summarizes the stress test results, reporting the number of submitted, successful, and failed transactions, along with achieved throughput, average latency, and success rate under progressively increasing load levels.

The results clearly indicate that the network maintains stable and reliable operation up to approximately 200 TPS, achieving a 100% transaction success rate with sub-second average latency. Beyond this threshold, a rapid degradation in success rate is observed, accompanied by a sharp increase in end-to-end latency. At target rates of 300 TPS and above, the system enters an overload regime, where transaction failures dominate and latency increases by an order of magnitude, reflecting congestion at the gateway and ordering layers. Importantly, this overload behavior does not lead to system deadlock or inconsistent ledger states. Instead, the permissioned Fabric network exhibits graceful degradation, continuing to process a subset of transactions while preserving ledger safety and consistency guarantees. It is important to note that the relationship between transaction throughput and latency differs across operating regimes. In the isolated throughput benchmark, latency initially decreases as throughput increases from low load levels. This behavior reflects improved pipeline utilization and more efficient block formation as the network transitions from an underutilized state toward a stable operating region. Such behavior is commonly observed in permissioned blockchain systems during the ramp-up phase. In contrast, the stress test results capture the saturation and overload regime of the network. Beyond the maximum sustainable throughput of approximately 200 TPS, transaction latency increases sharply due to queue buildup at the gateway and ordering layers, accompanied by a rapid degradation in success rate. This transition marks the boundary between stable operation and overload, providing a clear empirical characterization of the system's practical performance envelope. This behavior can be explained by the interaction between the block formation parameters and the transaction arrival rate. When the system operates below the saturation threshold, transactions are batched efficiently within the configured block timeout (2 s) and maximum message count (10 transactions), resulting in stable or decreasing average latency as throughput increases. However, once the incoming transaction rate approaches the effective capacity limit of approximately 200 TPS, newly arriving transactions begin to accumulate in the ordering queue. Although transaction submission continues, block formation is constrained by the timeout and batch size parameters, leading to queue buildup and a sharp increase in end-to-end latency. This effect is characteristic of queueing systems under saturation

and reflects controlled back-pressure rather than system instability or failure. By contrast, the high-throughput figures frequently reported for Hyperledger Fabric, often reaching thousands of TPS, are typically obtained under highly simplified benchmark conditions involving minimal transaction logic, limited endorsement complexity, and the absence of cryptographic or cross-layer verification mechanisms. By design, each transaction in the SecureRelief platform encapsulates end-to-end security and governance operations, including WebAuthn-based authentication, attribute-based access control, multi-party endorsement, zero-knowledge proof generation and verification, and off-chain evidence coordination. Consequently, the sustainable throughput level of approximately 200 TPS observed in our experiments represents a realistic application-level performance envelope under full security and governance constraints, rather than a fundamental limitation of the underlying Fabric platform. While the stress and isolated benchmarks characterize system-level throughput–latency behavior under varying load conditions, the following subsection focuses specifically on endorsement and processing latencies within the Execute–Order–Validate pipeline.

3.2.3. Latency

In contrast to the system-level latency effects observed under saturation conditions in Section 3.2.2, endorsement latencies have been analyzed using the p50, p95, and p99 percentiles. As illustrated in Figure 10, endorsement latency has remained around p50 $\approx$ 3.2 ms, p95 $\approx$ 15.2 ms, and p99 $\approx$ 28.1 ms. These results indicate that the Donation, Shipment, and Delivery chaincodes have been executed efficiently within the Node.js runtime and that endorsing peers have not introduced a performance bottleneck. From the perspective of block processing and chaincode invocation delays, the *Processing Latency* (p95) metric has remained stable at approximately 4.75 ms. This observation has confirmed that the ordering and commit stages within the Execute–Order–Validate pipeline have operated consistently throughout the experimental period.

3.2.4. Ledger Consistency and Consensus Stability

The consensus layer has been evaluated based on blockchain height and synchronization across peer nodes. The *Blockchain Height by Peer* panel shown in the lower section of Figure 10 indicates that all Fabric peers associated with *AidOrg*, *DonorOrg*, *GovOrg* and *LogisticsOrg* have remained synchronized at the same block height (78), which corresponds to the total number of blocks generated in the blockchain ledger at the end of the experiment. This observation demonstrates that the Raft-based ordering service has produced a deterministic block sequence throughout the experiment, without forks, leader changes, or orphaned blocks. As transactions have been submitted sequentially, the MVCC mechanism has not reported any read–write conflicts, further confirming that state transitions for donation and delivery objects have been designed in a conflict-free manner.

3.2.5. Interaction Between the API Gateway and the Blockchain Layer

The interaction between the API Gateway layer and the blockchain has been analyzed using the *API Gateway–Request Rate* and *API Response Latency* panels shown in Figure 12. Under the experimental scenario, the API Gateway has handled an average request rate of approximately 15.7 requests per second, reaching peak values of up to 184 requests per second during burst periods. Despite this load, the error rate reflected in the 5xx panel has remained at 0% throughout the experiment, and the status indicators of all core services, including *api-gateway*, *identity-service*, and *privacy-gateway*, have consistently been observed as “UP”.



Figure 12. Request traffic and latency metrics of the API Gateway layer. In the upper panel, the green, yellow, and red curves denote the total request rate, successful responses (2xx), and client errors (4xx), respectively, while the lower panel shows the color-coded distribution of API calls across delivery, donation, shipment, and authentication endpoints over time.

With respect to API response latency, measured values have concentrated around an average of approximately 303 ms, with a maximum observed latency of approximately 469 ms. This delay has been largely attributable to endorsement and commit durations on the blockchain side, indicating that the API layer itself has not introduced any significant additional latency.

Overall, the blockchain layer of the SecureRelief platform has been characterized by sufficient throughput for medium-scale disaster scenarios, low endorsement and block processing latencies, stable Raft-based consensus, and a near-zero error rate. The experimental findings have demonstrated that the proposed multi-stakeholder donation–shipment–delivery model, designed on a permissioned Hyperledger Fabric network, provides a performance-capable infrastructure for transparent and traceable disaster relief operations when combined with real-time monitoring capabilities.

3.3. Impact of Privacy Preserving Mechanisms

This section examines the concrete impact of the Privacy Gateway and Zero-Knowledge Proof components, whose theoretical foundations have been detailed in Section 2.5, on the operational metrics of the SecureRelief network. The analysis has been conducted along three primary dimensions: latency, computational overhead, and data minimization, based on real-time measurements collected from the deployed system.

3.3.1. Computational Overhead and Latency Analysis

The additional cost introduced by privacy-preserving operations has been analyzed across two layers: (1) the computational time required for proof generation and (2) the end-to-end blockchain consensus duration. Performance tests consisting of 10 iterations conducted in the test environment using Node.js v23 and the BN128 curve have indicated that the computational overhead imposed by the privacy layer has remained relatively low. As reported in Table 6, generating a role membership proof using the Groth16 protocol has required an average of 33.0 ms. Similarly, the computation of a Pedersen commitment has been recorded with an average duration of 35.0 ms.

Table 6. Experimental results of Zero-Knowledge Proof generation time (10 iterations).

Iteration Number	Role Membership Proof (ms)	Pedersen Commitment (ms)
1	36	34
2	31	32
3	34	38
4	36	35
5	31	36
6	32	33
7	36	34
8	31	36
9	33	42
10	32	37
Average (μ)	33.0 ms	35.0 ms
Std. Dev (σ)	$\pm$2.16 ms	$\pm$2.94 ms

While the API response time of a standard blockchain transaction has been measured at 46 ms in the test environment, the proof generation time of ZKP-enabled transactions has remained below this level at approximately 33 ms. This result demonstrates that the proposed cryptographic model can operate without causing performance issues even on resource-constrained devices such as mobile clients.

However, when end-to-end processes are considered, the following observations have been made:

1. *Standard Transaction*: An average duration of approximately 2.1 s, including chaincode endorsement and block commit stages.
2. *Privacy-Preserving Transaction*: An end-to-end duration ranging between 3.5 and 4.5 s, including secure channel establishment and verification steps between the Privacy Gateway and the Identity Service.

In light of these measurements, the observed increase in total transaction duration has not originated from cryptographic computation overhead, which remains limited to approximately 33 ms, but rather from inter-service network communication inherent in the distributed architecture. Considering that disaster management workflows typically operate on timescales of minutes rather than seconds, this additional latency overhead introduced in exchange for data privacy has been evaluated as an acceptable cost that does not disrupt operational processes.

3.3.2. On-Chain Data Minimization

The storage efficiency of the system has been validated through a data minimization strategy. Unlike conventional systems, SecureRelief does not record Personally Identifiable Information (PII) or sensitive logistical details in raw form on the ledger.

The size of the generated proof has remained constant at approximately 288 bytes and has been independent of transaction content. This design enables the storage of only mathematical summaries such as the proof (π), nullifier, and public signals on-chain, instead of persisting gigabytes of sensitive data. Experimental analysis has shown that retaining sensitive data off-chain using IPFS while anchoring only cryptographic proofs on-chain has resulted in storage cost savings exceeding 90%. This approach enforces GDPR compliance as a technical requirement while mathematically minimizing the risk of data leakage.

3.3.3. Impact on Transaction Throughput

When examining the impact of the privacy layer on overall system throughput, it has been observed that despite the increase in per-transaction execution time, no throughput bottleneck has emerged in terms of transactions per second (TPS). The primary reason for this behavior is that computationally intensive ZKP operations, which introduce an

overhead of approximately 33 ms as shown in Table 6, have been executed off-chain on asynchronously operating client devices and the Privacy Gateway rather than on blockchain peer nodes. The network's consensus engine, namely the Raft-based Ordering Service, has only verified the correctness of the generated proofs with an approximate verification time of 15 ms, without being burdened by complex cryptographic computations. As a result, the architectural design has succeeded in preserving system scalability while ensuring privacy guarantees.

3.4. Off-Chain Storage Efficiency

This section analyzes the performance and efficiency gains achieved by the proposed hybrid storage architecture in managing large-volume data that are critical in disaster management scenarios, such as damage assessment photographs, logistical documents, and official PDF reports. To overcome issues related to ledger bloat and high transaction costs caused by directly writing large files to the blockchain ledger, the SecureRelief architecture has adopted a strategic approach that combines on-chain and off-chain storage. In the conducted experimental evaluations, an approach has been assessed in which raw file contents have been stored on the distributed file system IPFS using Kubo v0.39, while only 46-byte Content Identifiers (CIDs) have been anchored on the blockchain. Under this approach, the amount of data stored on-chain has become independent of file size, thereby maximizing storage efficiency.

3.4.1. Storage Optimization

In blockchain networks, limited block sizes and inter-node replication requirements render on-chain storage of large files impractical. In the proposed system, regardless of file size, the amount of data committed to the ledger has been reduced to a fixed reference value represented by the CID.

Table 7 presents a comparison of storage costs and efficiency between direct on-chain storage and the proposed hybrid architecture across different file sizes. Representing a 1 MB image file, corresponding to 1,048,576 bytes, using a 46-byte CID has yielded a storage reduction of approximately 22,795 times (22,795×). When the file size increases to 10 MB, the amount of data written to the ledger remains constant, resulting in an efficiency gain of up to 227,951 times (227,951×).

Table 7. Comparison of storage cost and efficiency between on-chain and hybrid storage approaches.

File Type	File Size (Bytes)	Traditional On-Chain Storage	Proposed Hybrid Approach (CID)	Efficiency Gain
Medium-sized (Image file)	1.048.576 (1 MB)	~1.05 MB	46 Bayt	22,795×
Large-sized (PDF/Report file))	10.485.760 (10 MB)	~10.5 MB	46 Bayt	227,951×

These findings have demonstrated that the hybrid storage approach provides scalability independent of file size and prevents the exponentially increasing data volume during disaster scenarios from negatively affecting blockchain network performance.

3.4.2. Latency and Integrity Analysis

The operational speed of the system has been evaluated based on the processes of uploading files to the IPFS network, generating SHA-256-based cryptographic hashes, and producing content identifiers (CIDs). Experimental results have shown that the additional latency overhead introduced by off-chain storage operations has remained negligible when compared to the efficiency gains achieved:

1. Processing, hash generation, and IPFS upload of a 1 MB image file have required an average of approximately 95 ms.
2. The same processes for a 10 MB PDF document have taken approximately 195 ms.

Considering that blockchain consensus durations, including endorsement, ordering, and commit stages, typically operate on the order of seconds, these millisecond-level overheads have not disrupted real-time data flows in disaster field operations.

From a data integrity perspective, CID values generated during the experiments, such as *QmdSVFy*... for the 1 MB file and *QmVCy9F*... for the 10 MB file, have functioned as cryptographic fingerprints due to the content-addressed nature of IPFS. Any modification to off-chain data, even at the level of a single pixel change, would result in a completely different CID. Consequently, any inconsistency between the on-chain reference and the off-chain data can be detected with mathematical certainty during verification. Overall, SecureRelief has delivered a reliable architecture that achieves high storage efficiency without compromising data integrity.

4. Discussion

This study has examined the technical and operational feasibility of the SecureRelief platform, which has been developed in response to structural challenges related to trust, transparency, and data privacy in disaster logistics. Experimental evaluations have demonstrated that the donation–shipment–delivery lifecycle can be executed in a traceable manner on a permissioned Hyperledger Fabric network, that latency metrics have remained stable, and that the system has operated without producing errors under the defined test scenarios. The distinguishing characteristics of the platform lie in its integrated handling of identity and privacy requirements through SSI/DID and zero-knowledge proofs, the management of high-volume evidence data via IPFS-based hybrid storage, and the realization of delivery verification through an *m*-of-*n* threshold attestation model involving multiple stakeholders. When considered jointly, these components provide a design that goes beyond mere record keeping and embeds verifiability and accountability directly into operational disaster management workflows.

4.1. Performance and Scalability Considerations

A central discussion in the literature on blockchain-based disaster management systems has focused on whether such architectures can deliver the scalability and performance stability required during crisis conditions. Approaches built on public blockchain infrastructures, such as Ethereum-based models, have often been associated with risks related to fluctuating transaction fees and unpredictable latency under network congestion, thereby posing challenges for ensuring the operational continuity demanded by disaster logistics. The architectural choices, adopted security mechanisms, and resulting technical constraints of existing studies in the literature are comparatively summarized in Table 8, which explicitly positions the proposed SecureRelief architecture with respect to prior blockchain-based disaster relief and humanitarian logistics systems. An examination of these comparisons indicates that Ethereum-based systems have frequently encountered cost and scalability bottlenecks, whereas the proposed SecureRelief architecture has mitigated these structural limitations through its permissioned design and hybrid data storage strategy.

Based on Table 8, the proposed SecureRelief architecture is an approach in the compared literature that simultaneously combines self-sovereign identity, zero-knowledge privacy, dynamic ABAC, and cryptographically verifiable proof of delivery within a unified end-to-end disaster logistics workflow.

Table 8. Comparison of the proposed approach with related work in the literature.

Reference	Platform	Identity	Privacy	Access Control	Focus Area	Limitations/Improvement Areas
[28]	Ethereum	MetaMask	None	Standard	Donation Tracking	High transaction fees, no data privacy, lack of decentralized identity.
[29]	Ethereum	Wallet Address	None	Role-Based (Basic)	Donation Management	Scalability issues, sensitive data exposed on-chain.
[30]	Hyperledger Fabric	MSP	Channel-Based	ACL (Static)	Logistics Transparency	Lack of advanced privacy mechanisms (ZKP) and dynamic access control (ABAC).
[31]	Generic	Static Roles (Admin)	None	Role-Based (RBAC)	Logistics Prediction & Transparency	No privacy protection (ZKP), theoretical system architecture, absence of DID.
[36]	Ethereum/Flask	Web 2.0 Login	None	Role-Based	Missing Persons & Logistics	Data integrity limited to on-chain records only.
[40]	Ethereum	Registered Membership	Searchable Encryption (SE)	List-Based	Donation Search	High cryptographic overhead, identity authentication dependent on central authorities.
[41]	Quorum	DID/SSI	ZKP (zk-SNARKs)	Role-Based	Data Sharing	Lack of logistics proof of delivery and absence of ABAC.
[42]	FISCO BCOS	BBS+ Signatures	ZKP (ZKPoK)	Anonymity-Oriented	Goods Donation	Auditability-focused design, limited automation of logistics processes.
[43]	Hyperledger Besu	Wallet/App	Multi-signature (Partial)	Role-Based	Disaster Preparedness & NFT	NFT-centric design may be costly for logistics use cases, no ABAC support.
Proposed System	Hyperledger Fabric	SSI + DID	ZKP + Off-Chain	ABAC (Dynamic)	End-to-End Logistics	Integrated architecture combining identity, privacy, and proof of delivery.

Building on the architectural and functional positioning provided in Table 8, the structural differences among these architectural choices are reflected in measurable system-level performance outcomes, particularly in terms of transaction throughput and latency. These effects are comparatively analyzed in Table 9 using reported performance metrics obtained under different architectural and experimental contexts. The comparison spans blockchain infrastructures commonly adopted in disaster management and supply chain-oriented scenarios, including public blockchain platforms such as Ethereum as well as permissioned blockchain systems.

Table 9. Comparative performance characteristics reported in representative blockchain-based systems with architectural and experimental context awareness.

Reference	Blockchain Type	Evaluation Context	Throughput (TPS)	Latency	Consensus Algorithm
[28]	Public	Reported (public chain, block-time-driven)	~7–20 TPS	~12–15 s (Block Time)	PoW/PoS
[41]	Permissioned	Reported (BFT-style ordering)	~95.66 TPS (Max)	~1 s (Block Interval)	QBFT (BFT Variant)
[43]	Permissioned	Reported (PoA/BFT finality model)	~150 TPS (Peak)	<2 s (Finality ~5 s)	Clique PoA/QBFT
Proposed System	Permissioned	Observed (conservative, sequential workflow; not a benchmark)	~0.32–2.0 TPS	<30 ms (p99)	Raft (CFT)
Proposed System	Permissioned	Isolated benchmark (ledger-level, controlled load)	199.2 TPS (sustainable)	~80–100 ms	Raft (CFT)

An examination of the data presented in Table 9 indicates that reported throughput and latency values are strongly influenced by consensus mechanisms, deployment models,

and evaluation objectives. Studies relying on public blockchain infrastructures such as Ethereum exhibit throughput levels in the range of approximately 7–20 transactions per second (TPS), accompanied by block confirmation times reaching up to 15 s. While such configurations provide strong decentralization guarantees, they may introduce non-negligible delays under time-critical disaster response conditions. In contrast, studies adopting permissioned blockchain architectures based on platforms such as Hyperledger Besu and Quorum report substantially higher throughput values, ranging from approximately 95.66 TPS to 150 TPS, with transaction finality typically achieved within a few seconds. These results confirm the well-established advantage of permissioned blockchain systems in terms of throughput predictability and operational efficiency under controlled governance models. Within this comparative context, the distinguishing performance characteristic of the proposed SecureRelief platform emerges primarily in its low-latency behavior rather than raw throughput. While throughput values observed for SecureRelief remain deliberately conservative due to sequential transaction submission and end-to-end workflow execution, the system achieves transaction finality below 30 ms at the p99 percentile by leveraging a Raft-based ordering service. To further contextualize these observations, an isolated ledger-level benchmark conducted under controlled load conditions (Section 3.2.2) demonstrates that the underlying Hyperledger Fabric network can reliably process approximately 199 TPS with a 100% success rate. This result confirms that the conservative throughput values reported for SecureRelief arise from application-level execution choices rather than infrastructural limitations. Furthermore, the stabilization of processing latency at approximately 4.75 ms at the p95 percentile demonstrates the system's ability to support near real-time verification events, such as QR-based delivery confirmations, during intensive disaster operations.

These results indicate that the SecureRelief architecture has been intentionally optimized for low-latency finality and operational responsiveness rather than for maximizing throughput under benchmark-oriented conditions. Such a design choice aligns with the real-time verification and auditability requirements inherent to disaster relief workflows. Overall, the experimental findings are consistent with the prevailing view in the literature that permissioned blockchain networks provide a more predictable and sustainable performance foundation for institutional disaster management processes when compared to public blockchain infrastructures.

4.2. Privacy-Preserving Identity and Access Management

In disaster scenarios, protecting the identity and attribute information of donors and affected individuals is as critical as ensuring process transparency. The proposed system has preserved Fabric's MSP-based organizational identity model while extending it with SSI/DID and Verifiable Credentials at the user level. In privacy-sensitive cases, Zero-Knowledge Proofs (ZKPs) have been employed to enable authorization and membership verification without explicit identity disclosure.

Experimental results have shown that the cryptographic computation overhead introduced by the privacy-preserving layer remains negligible within the overall transaction lifecycle, typically on the order of milliseconds. Nevertheless, the observed increase in end-to-end latency has not been solely attributable to cryptographic computation, but also to inter-service communication, secure channel establishment, and verification workflows inherent in distributed system integration. Specifically, while standard transactions have completed in approximately 2.1 s, privacy-preserving transactions have required between 3.5 and 4.5 s. This observation highlights the importance of distinguishing cryptographic overhead from distributed system orchestration costs when evaluating the performance impact of privacy-enhancing mechanisms.

From a usability–security trade-off perspective, OTP-supported onboarding and WebAuthn/Passkey-based authentication have been designed to reduce password management risks. Compared with similar schemes in the literature [39], this approach has the potential to reduce operator-induced errors in field conditions; however, its effectiveness should be further validated through user studies and pilot deployments.

4.3. Data Integrity and Hybrid Storage Strategy

The high volume of evidential data generated in disaster zones, such as photographs, delivery reports, and official documents, renders direct on-chain storage impractical. SecureRelief has addressed this challenge by storing files on IPFS and recording only content identifiers (CIDs) on-chain, thereby decoupling ledger growth from file size.

Storage optimization results have demonstrated that a fixed-size on-chain reference (CID) can be maintained regardless of the underlying file size, providing a scalable solution to the *ledger bloat* problem. These findings offer a strong counterargument to approaches that rely on off-chain storage without robust integrity binding, where trust in off-chain data has been frequently questioned [36]. By anchoring cryptographic references on-chain, SecureRelief has ensured verifiable integrity without sacrificing scalability.

From an operational latency perspective, IPFS upload combined with hash and CID generation has taken approximately 95 ms for a 1 MB file and 195 ms for a 10 MB file. Given that blockchain consensus processes operate on the order of seconds, these millisecond-level delays have not imposed a limiting effect on real-time data flows under the tested conditions.

4.4. Theoretical and Practical Implications

From a theoretical perspective, the proposed SecureRelief architecture contributes to the literature on blockchain-enabled multi-stakeholder systems by formalizing trust, accountability, and governance as first-class architectural concerns rather than as deployment-specific features. In contrast to prior works that primarily focus on transaction-level performance or single-organization control models, this study conceptualizes disaster relief logistics as a partially trusted coordination problem, requiring explicit mechanisms for multi-party attestation, verifiable delivery confirmation, and policy-enforced governance.

The trust architecture of the platform has been designed to mitigate the risk of trust erosion that may arise when delivery verification relies on a single institutional declaration. To this end, an m-of-n threshold approval mechanism has been employed, with a default policy of 2-of-3 requiring approvals from at least two of *AidOrg*, *GovOrg*, and *LogisticsOrg*, while allowing certain approvals to be designated as mandatory.

This multi-stakeholder verification model reduces the risk of erroneous or biased reporting by individual actors and enhances retrospective auditability. Notably, the policy semantics have also accounted for rejection sensitivity: in defined scenarios, a single *REJECT* decision may invalidate a delivery. While this conservative design functions as a fail-safe mechanism in cases of evidence inconsistency or incorrect delivery, it may increase operational costs due to false-positive rejections. Consequently, policy parameters should be adapted based on disaster type and operational risk levels.

In terms of transparency, the provision of a public verification endpoint has enabled external stakeholders to query the donation–shipment–delivery chain without requiring authentication. While this feature is expected to enhance donor trust and institutional coordination, robust academic validation will require pilot deployments and empirical analysis of user behavior.

The proposed SecureRelief platform goes beyond an implementation-level solution by offering a generalized architectural abstraction for secure and accountable disaster relief

systems operating in multi-stakeholder environments. Rather than optimizing a single performance metric, the design explicitly addresses the trade-offs between security, privacy, accountability, and operational efficiency through the combined use of permissioned blockchain governance, decentralized identity management, zero-knowledge verification, and multi-party attestation mechanisms. In this respect, the proposed architecture contributes to the existing literature by demonstrating how heterogeneous trust-enabling components can be systematically integrated into a unified design framework. This integration provides conceptual insights applicable not only to disaster logistics but also to other safety-critical and resource-constrained domains requiring auditable coordination among partially trusted actors.

These implications extend beyond the specific disaster relief scenario considered in this study. The architectural abstractions introduced in SecureRelief provide a transferable conceptual model for other safety-critical and resource-constrained domains, such as healthcare logistics, humanitarian supply chains, and inter-organizational emergency coordination, where trust fragmentation, accountability, and privacy preservation must be jointly addressed.

4.5. Limitations

While the experimental evaluation confirms the technical feasibility and robustness of the SecureRelief architecture, certain limitations remain inherent to the system design and experimental scope, particularly under real-world disaster conditions.

First, the operation of both IPFS-based off-chain storage and blockchain synchronization depends on the availability of network connectivity. In disaster scenarios where communication infrastructures are damaged or unstable, delayed synchronization and the use of local buffering or deferred submission mechanisms become critical considerations. However, the SecureRelief architecture is not strictly dependent on continuous real-time connectivity for local operation. Core components, including the permissioned blockchain ledger and IPFS-based evidence handling, support deferred propagation and eventual consistency once network access is restored, allowing evidence generation and transaction preparation at the edge to be decoupled from immediate global synchronization. At the same time, the current implementation does not yet provide a fully orchestrated offline-first execution layer across all components, meaning that prolonged connectivity disruptions may delay global state convergence rather than prevent local operational continuity. Nevertheless, the prototype already incorporates limited resilience features that reduce operator friction under transient network disruptions, including client-side session persistence, temporary local buffering of evidence artifacts prior to IPFS publication, and short-lived caching of gateway connections. These mechanisms support local continuity during brief outages, although comprehensive offline-first operation remains a subject of future work.

Second, the current experimental setup involved sequential transaction submission, under which no Multi-Version Concurrency Control (MVCC) conflicts have been observed. In practical deployments, however, higher levels of concurrent write operations may lead to increased contention, altered conflict rates, and queuing effects that could influence overall system throughput and latency.

Third, the permissioned network model assumes reliable management of organizational identities and Membership Service Provider (MSP) configurations. Although the proposed architecture incorporates technical controls to mitigate insider misuse and misconfiguration risks, such measures cannot fully eliminate organizational or human-related vulnerabilities. Finally, while Passkey and QR-based workflows enhance usability and reduce authentication friction, effective adoption in field operations may require structured

training procedures and well-defined operational guidelines for personnel involved in disaster response activities.

4.6. Future Work

Although the experimental results demonstrate the feasibility and robustness of the SecureRelief architecture under controlled conditions, several extensions are necessary to address the complexity and uncertainty inherent in real-world disaster environments. Large-scale deployments, intermittent connectivity, heterogeneous data sources, and human-in-the-loop operations introduce additional challenges that extend beyond the scope of the present evaluation and motivate further investigation.

One important direction for future work involves the development of offline-first extensions, including store-and-forward mechanisms that leverage alternative communication infrastructures such as mesh networks or LoRaWAN, to enable local evidence collection during prolonged connectivity outages. Another promising avenue is the integration of IoT- and sensor-based data sources, such as temperature and humidity sensors for cold-chain logistics, to automate evidence generation and directly associate sensor readings with shipment-related smart contracts.

Building upon these directions, future work will further extend the existing delay-tolerant characteristics of SecureRelief toward a fully offline-first operational model. In particular, a hybrid delay-tolerant architecture is envisioned, combining offline-first client capabilities with optional edge-level buffering nodes deployed at distribution points. On the client side, operations such as donation creation, delivery attestations, and evidence capture may be recorded locally using an outbox-style queuing pattern and synchronized opportunistically once connectivity becomes available. In parallel, lightweight edge nodes operating within local connectivity islands, such as temporary distribution centers or field offices, can batch transactions and evidence artifacts and perform deferred submission to the consortium network when backhaul connectivity is restored. This approach would enable sustained operation under prolonged network disruption while preserving the integrity, auditability, and governance guarantees of the permissioned blockchain infrastructure. In addition, future evaluations will quantify system behavior under intermittent and partially connected network conditions using metrics such as resynchronization latency, convergence time to global consistency, and the availability of verification services in fragmented disaster environments.

Further research is also required to conduct comprehensive stress testing and field pilots, encompassing multi-region deployments, varying peer counts, network delays, and high levels of transaction concurrency. Such evaluations would allow a deeper assessment of system stability and performance, as well as an examination of how transparency mechanisms influence donor behavior and inter-institutional coordination through both quantitative and qualitative methodologies. Finally, future extensions may incorporate a decision-support layer based on machine learning and operations research techniques to support demand prediction and allocation optimization while balancing efficiency and equity objectives. In parallel, the exploration of tokenized cash assistance modules represents another potential research direction, although such extensions would necessitate a distinct threat model and carefully designed pilot studies due to regulatory, custody, misuse, and compliance considerations.

5. Conclusions

In this study, an integrated platform named SecureRelief has been proposed to ensure the secure and verifiable delivery of humanitarian aid materials to beneficiaries in disaster scenarios. The platform has automated the donation–shipment–delivery lifecycle through

smart contracts deployed on a permissioned Hyperledger Fabric network, while high-volume evidence artifacts have been stored on IPFS, with only content identifiers (CIDs) and SHA-256 hashes anchored on-chain. This design has aimed to jointly achieve data integrity and system scalability. In addition, SSI/DID, WebAuthn/Passkeys, context-aware ABAC policies, and zero-knowledge proofs for privacy preservation have been brought together within a unified architecture, thereby offering a balanced design between operational speed and data protection requirements under crisis conditions.

The trust model has been strengthened through an m-of-n threshold approval mechanism to avoid reliance on single-entity declarations during delivery verification. With a default 2-of-3 policy and mandatory organization approval constraints, signed attestations from field actors have produced an auditable and non-repudiable record trail, while conservative fail-safe termination has been enabled in cases of inconsistency or suspicion. From a transparency perspective, a public verification approach that does not require identity authentication has allowed donors and third parties to holistically query the linkage between donation, shipment, and delivery records, thereby providing a verifiable answer to the question of whether the aid has actually reached its destination.

In the experimental evaluation, the system has demonstrated stable performance under scenario-based workload profiles in a laboratory environment. The results have shown that low latency at the millisecond level has been preserved even under increased transaction loads, while maintaining a level of throughput suitable for disaster management operations. These findings indicate that chaincode execution and the Execute–Order–Validate pipeline have not introduced significant bottlenecks under moderate workloads. Moreover, the IPFS-based off-chain evidence approach has supported auditability while mitigating on-chain data bloat.

With respect to the research questions formulated in this study, the proposed SecureRelief architecture provides the following answers. To address RQ1, the use of a permissioned blockchain infrastructure combined with a unified data model eliminates data fragmentation by ensuring that donation, shipment, and delivery records are immutably recorded and verifiably linked across the entire relief lifecycle. To address RQ2, the integration of SSI/DID, WebAuthn-based authentication, and context-aware ABAC enables secure and fine-grained authorization while maintaining operational usability in dynamic disaster environments. Regarding RQ3, the blockchain-based ledger and layered smart contract design collectively support secure execution, performance stability, and data integrity across all operational stages, from donation initiation to final delivery. Finally, in response to RQ4, the resulting verifiable and auditable records, reinforced through multi-party attestation mechanisms, provide a transparent foundation that enhances trust among donors, non-governmental organizations, and other stakeholders involved in humanitarian relief operations.

In this context, SecureRelief brings together permissioned blockchain governance, privacy-preserving identity verification, off-chain evidence binding, and multi-party delivery attestation within a unified operational workflow, thereby offering a reference architecture that goes beyond fragmented blockchain-based disaster management solutions.

Author Contributions: Conceptualization of the study and definition of the research scope: G.G. and Ö.K. Design of the overall system architecture and permissioned blockchain model: Ö.K. Analysis of disaster relief logistics requirements and problem formulation: G.G. and Ö.K. Design of the identity, access control, and privacy-preserving architecture (SSI/DID, ABAC, ZKP): Ö.K. Development of the hybrid on-chain/off-chain data storage model using IPFS: G.G. and Ö.K. Design and implementation of smart contract logic for donation, shipment, and delivery processes: Ö.K. Formal modeling, algorithm design, and mathematical formulation of the system components: Ö.K. Design and execution of experimental setup and performance evaluation: G.G. and Ö.K. Analysis and interpretation of

experimental results and performance metrics: G.G. and Ö.K. Comparative analysis with existing blockchain-based disaster management solutions: G.G. and Ö.K. Identification of limitations and formulation of future research directions: G.G. and Ö.K. Writing—original draft preparation: G.G. and Ö.K. Writing—review and editing: G.G. and Ö.K. Supervision, coordination, and final approval of the manuscript: Ö.K. All authors have read and agreed to the published version of the manuscript.

Funding: This study is supported by the Institution of Firat University Scientific Research Projects (FUBAP) under project number MF.25.102, with the APC funded by FUBAP.

Data Availability Statement: The data supporting the findings of this study are contained within the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The authors acknowledge the CHIST-ERA Di4SPDS project (CHIST-ERA-22-SPiDDS-01) and its national partner project, TUBITAK 223N142, conducted under the TUBITAK 1071 International Collaboration Program. Additionally, this study has been carried out within the scope of a master's thesis titled "Afetlerde Yardım Malzemelerinin Blockchain Teknolojisi Kullanılarak Güvenli Şekilde Ulaştırılması (Secure Delivery of Relief Supplies using Blockchain Technology in Disasters)" by Gülsena Gülhas, under the supervision of Özgür Karaduman, at Fırat University, Department of Software Engineering.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Aksoy, C.G.; Chupilkin, M.; Koczan, Z.; Plekhanov, A. Unearthing the impact of earthquakes: A review of economic and social consequences. *J. Policy Anal. Manag.* **2025**, *44*, 1450–1471. [CrossRef]
2. Zhang, F.; Liu, Y.; Yun, H.; Cao, C.; Liu, X. Pre and Post-Disaster Allocation Strategies of Relief Items in the Presence of Resilience. *Systems* **2025**, *13*, 704. [CrossRef]
3. Wang, S.; Sun, B. Model of multi-period emergency material allocation for large-scale sudden natural disasters in humanitarian logistics: Efficiency, effectiveness and equity. *Int. J. Disaster Risk Reduct.* **2023**, *85*, 103530. [CrossRef]
4. Li, S.; Zhang, N.; Qin, J. An Inter-Regional Lateral Transshipment Model to Massive Relief Supplies with Deprivation Costs. *Mathematics* **2025**, *13*, 2298. [CrossRef]
5. Köstepen, Z.N.; Selim, H. Literature Review on Humanitarian Logistics in Disaster Management: A Risk-Oriented Approach. *Sustainability* **2025**, *17*, 9773. [CrossRef]
6. Wang, B.; He, C.; Guo, Y. Building Disaster Resilience: A Sustainable Approach to Integrated Road Rehabilitation and Emergency Logistics Optimization in Extreme Events. *Sustainability* **2025**, *17*, 10591. [CrossRef]
7. Khaddour, L.A.; Kazbek, C.; Elhassnaoui, I. Evaluating Flood Risk Assessment in Turkey: Advancing Climate Change Adaptation and Resilience. *Eng. Proc.* **2025**, *112*, 49.
8. Safak, I.; Okan, T.; Karademir, D. Perceptions of Turkish Forest Firefighters on In-Service Trainings. *Fire* **2023**, *6*, 38. [CrossRef]
9. Atalay, H.; Dervisoglu, A.; Sunar, A.F. Exploring Forest Fire Dynamics: Fire Danger Mapping in Antalya Region, Türkiye. *ISPRS Int. J. Geo-Inf.* **2024**, *13*, 74. [CrossRef]
10. Ülkü, M.A.; Bookbinder, J.H.; Yun, N.Y. Leveraging Industry 4.0 Technologies for Sustainable Humanitarian Supply Chains: Evidence from the Extant Literature. *Sustainability* **2024**, *16*, 1321. [CrossRef]
11. Shevchuk, R.; Lishchynskyy, I.; Ciura, M.; Lyzun, M.; Kozak, R.; Kasianchuk, M. Application of Blockchain Technology in Emergency Management Systems: A Bibliometric Analysis. *Appl. Sci.* **2025**, *15*, 5405. [CrossRef]
12. Karaduman, Ö.; Gülhas, G. Blockchain-Enabled Supply Chain Management: A Review of Security, Traceability, and Data Integrity Amid the Evolving Systemic Demand. *Appl. Sci.* **2025**, *15*, 5168. [CrossRef]
13. Rinaldi, M.; Caterino, M.; Riemma, S.; Macchiaroli, R.; Fera, M. Emergency Supply Chain Resilience Enhanced Through Blockchain and Digital Twin Technology. *Logistics* **2025**, *9*, 43. [CrossRef]
14. Astarita, V.; Martino, G.; Guido, G.; Haghshenas, S.S.; Haghshenas, S.S. Risk Management in Transportation Systems: The use of Blockchain for risk reduction in disaster's occurrence. *Procedia Comput. Sci.* **2025**, *257*, 428–435. [CrossRef]
15. Jha, P.K.; Priya, S.; Chandrakar, K.; Pradhan, A. Blockchain-Enabled Supply Chain Transparency and Smart Contracts for Efficient Humanitarian Aid Operations in NGO. In Proceedings of the 2024 5th International Conference for Emerging Technology (INCET), Belgaum, India, 24–26 May 2024; pp. 1–6.
16. Nazerath, P.; Uppoor, P.; Laxman, B.P.; Nayak, S.M.; Jagadeesha, S.G. Blockchain in Disaster Management: A Systematic Review of Emerging Frameworks, Applications, and Future Directions. In Proceedings of the 2025 4th International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Tirupur, India, 3–5 September 2025; pp. 505–510.

17. Abdullah, A.; Satria, A.; Mulyati, H.; Arkeman, Y.; Indrawan, D. Blockchain-Enabled Supply Chain Finance: A Bibliometric Review and Research Agenda. *Adm. Sci.* **2024**, *14*, 298. [CrossRef]
18. Xia, J.; Li, H.; He, Z. The Effect of Blockchain Technology on Supply Chain Collaboration: A Case Study of Lenovo. *Systems* **2023**, *11*, 299. [CrossRef]
19. Lakshmi, B.S.; Rekha, K.S. Blockchain Technology and Supply Chain Management in a Charity Donation System: A Literature Review. In Proceedings of the 2023 International Conference on Recent Trends in Electronics and Communication (ICRTEC), Mysore, India, 10–11 February 2023; pp. 1–4.
20. Adediran, F.E.; Okunade, B.A.; Daraojimba, R.E.; Adewusi, O.E.; Odulaja, B.A.; Igbokwe, J.C. Blockchain for social good: A review of applications in humanitarian aid and social initiatives. *Int. J. Sci. Res. Arch.* **2024**, *11*, 1203–1216. [CrossRef]
21. Khan, M.M.; Khan, F.S.; Nadeem, M.; Khan, T.H.; Haider, S.; Daas, D. Scalability and Efficiency Analysis of Hyperledger Fabric and Private Ethereum in Smart Contract Execution. *Computers* **2025**, *14*, 132. [CrossRef]
22. Javadpour, A.; AliPour, F.S.; Sangaiah, A.K.; Zhang, W.; Ja'far, F.; Singh, A. An IoE blockchain-based network knowledge management model for resilient disaster frameworks. *J. Innov. Knowl.* **2023**, *8*, 100400. [CrossRef]
23. Raj, V.H.; Sreevani, N.; Thethi, H.P.; Nainwal, A.; Ziara, S.; Kumari, M.S. Blockchain and Machine Learning for Predictive Resource Distribution in Disaster Response Management. In Proceedings of the 2024 1st International Conference on Sustainable Computing and Integrated Communication in Changing Landscape of AI (ICSCAI), Greater Noida, India, 4–6 July 2024; pp. 1–8.
24. Bonfanti, R.C.; Oberti, B.; Ravazzoli, E.; Rinaldi, A.; Ruggieri, S.; Schimmenti, A. The Role of Trust in Disaster Risk Reduction: A Critical Review. *Int. J. Environ. Res. Public Health* **2024**, *21*, 29. [CrossRef]
25. Negi, S. A blockchain technology for improving financial flows in humanitarian supply chains: Benefits and challenges. *J. Humanit. Logist. Supply Chain Manag.* **2024**, *15*, 54–174. [CrossRef]
26. Inayatulloh. Increasing the Efficiency of Disaster Relief Assistance based on Blockchain Technology with Blockchain Sharding. In Proceedings of the 2025 7th International Conference on Intelligent Sustainable Systems (ICISS), Tirunelveli, India, 12–14 March 2025; pp. 496–500.
27. Cheng, F.; Wei, X. Research on Application Solutions of Blockchain Technology in Multi-Party Collaborative Emergency Logistics System. *Int. Bus. Econ. Stud.* **2024**, *6*, 9–21. [CrossRef]
28. Lathkar, M.; Deshmukh, P.; Patil, A.; Shelke, P. Increasing Donation Transparency in Disaster Relief: A Blockchain-based Solution. In Proceedings of the 2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS), Manama, Bahrain, 28–29 January 2024; pp. 1527–1532.
29. Ankit, K.C. Blockchain Based Donation Management in Disaster Response. *J. Innov. Inf. Technol. Appl.* **2024**, *6*, 45–59.
30. Inayatulloh; Susanto, B.H.; Setiabudiarto, N.; Teguh, W.; Erin; Jariyah, A. Blockchain Technology Model to Increase Transparency in the Distribution of Aid to Disaster Victims. In Proceedings of the 2024 IEEE 2nd International Conference on Electrical Engineering, Computer and Information Technology (ICEECIT), Jember, Indonesia, 22–23 November 2024; pp. 222–226.
31. Darmawan, B.; Okitasari, H.; Dwiyant, V. Preliminary research: Blockchain system design for post-disaster management in Indonesia. *J. Eng. Sci. Technol.* **2024**, *19*, 1268–1279.
32. Gao, X.; Chen, Z.; Huang, G.; Hezam, I.M. Blockchain-enabled safeguard mechanism in SCP-based relief supply chain designs in response to long-term disasters. *IEEE Access* **2024**, *12*, 133054–133066. [CrossRef]
33. Rajagopal, M.; Ramkumar, S.; Thimmiaraja, J.; Gobinath, R.; Kumar, K.S. Blockchain-based model for disaster relief supply chain management. In *The Role of Blockchain in Disaster Management*; Academic Press: Cambridge, MA, USA, 2025; pp. 33–49.
34. Mosallanezhad, B.; Hajiaghaei-Keshteli, M.; Cornejo, N.R.S.; Calvo, E.Z.R. An IoMT platform for an integrated sustainable energy-efficient disaster relief supply chain to prevent severity-driven disruptions during pandemics. *J. Ind. Inf. Integr.* **2023**, *35*, 100502. [CrossRef]
35. Mandal, S.; Kar, A.K.; Gupta, S.; Sivarajah, U. Achieving food supply chain resilience during natural disasters through industry 5.0 enablers-empirical insights based on an FsQCA approach. *Inf. Syst. Front.* **2023**, 1–24. [CrossRef]
36. Kolhatkar, D.; Jain, O.; Patil, S.; Shelke, P.; Mirajkar, R.; Wawage, P. Blockchain applications in disaster management systems. In Proceedings of the 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 26–28 May 2023; pp. 1–7.
37. Damaševičius, R.; Maskeliūnas, R. Blockchain-enabled smart contracts for secure and transparent timber traceability. *J. Ind. Inf. Integr.* **2025**, *48*, 100934. [CrossRef]
38. Meybodian, E.; Mostafavi, S.; Dargahi, T.; Bafghi, V.R. Secure and scalable trust management in IoT: A hierarchical blockchain-based approach. *Comput. Electr. Eng.* **2025**, *127*, 110631. [CrossRef]
39. Fani, V.; Ciccullo, F.; Bandinelli, R.; Pero, M. Cultivating trust: An empirical exploration of blockchain's adoption within the Italian wine supply chain. *Electron. Mark.* **2025**, *35*, 1–21. [CrossRef]
40. He, B.; Feng, T.; Fang, J.; Liu, C.; Su, C. A Secure and Efficient Charitable Donation System Based on Ethereum Blockchain and Searchable Encryption. *IEEE Trans. Consum. Electron.* **2023**, *70*, 263–276. [CrossRef]

41. Karaarslan, E.; Akdik, B. Toward Reliable Disaster Data Sharing with Blockchain and Zero-Knowledge Proofs. *Computer* **2025**, *58*, 30–40. [CrossRef]
42. Zhou, Y.; Lei, H.; Bao, Z. Eisdspa: An Efficient and Secure Blockchain-Based Donation Scheme with Privacy Protection and Auditability. *IEEE Open J. Commun. Soc.* **2024**, *5*, 7498–7510. [CrossRef]
43. Ghani, A.; Zinedine, A.; El Mohajir, M. Eghatha: A Blockchain-Based System to Enhance Disaster Preparedness. *Computers* **2025**, *14*, 405. [CrossRef]
44. Ju, C.; Shen, Z.; Bao, F.; Wen, Z.; Ran, X.; Yu, C.; Xu, C. Blockchain Traceability System in Complex Application Scenarios: Image-Based Interactive Traceability Structure. *Systems* **2022**, *10*, 78. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.