future internet

MDPI

Article

Decentralized and Secure Blockchain Solution for Tamper-Proof Logging Events

J. D. Morillo Reina and T. J. Mateo Sanguino *

Department of Electronics, Computer Systems, and Automation Engineering, University of Huelva, Avda. de las Artes S/N, 21007 Huelva, Spain; juandiego.morillo@alu.uhu.es
* Correspondence: tomas.mateo@diesia.uhu.es

Abstract: Log files are essential assets for IT engineers engaged in the security of server and computer systems. They provide crucial information for identifying malicious events, conducting cybersecurity incident analyses, performing audits, system maintenance, and ensuring compliance with security regulations. Nevertheless, there is still the possibility of deliberate data manipulation by own personnel, especially with regard to system access and configuration changes, where error tracking or debugging traces are vital. To address tampering of log files, this work proposes a solution to ensure data integrity, immutability, and non-repudiation through different blockchain-based public registry systems. This approach offers an additional layer of security through a decentralized, tamper-resistant ledger. To this end, this manuscript aims to provide a solid guideline for creating secure log storage systems. For this purpose, methodologies and experiments using two different blockchains are presented to demonstrate their effectiveness in various contexts, such as transactions with and without metadata. The findings suggest that Solana's response times make it well suited for environments with moderately critical records requiring certification. In contrast, Cardano shows higher response times, thus making it suitable for less frequent events with metadata that requires legitimacy.

Keywords: blockchain; Cardano; Solana; logs; Raspberry Pi; networks

Academic Editor: Paolo Bellavista

Received: 9 January 2025
Revised: 20 February 2025
Accepted: 27 February 2025
Published: 1 March 2025

Citation: Morillo Reina, J.D.; Mateo Sanguino, T.J. Decentralized and Secure Blockchain Solution for Tamper-Proof Logging Events. *Future Internet* **2025**, *17*, 108. https://doi.org/10.3390/fi17030108

1. Introduction

Log files are used as a control mechanism to detect security incidents, policy violations, regulatory compliance, and operational problems in computer applications. These logs contain information about specific events on an organization's networks or systems. They provide both the current states of the systems and the traceability of the actions performed on the network devices, providing a detailed description of their behavior. Logs are therefore essential for forensic investigation and security auditing, such as compliance with regulations and standards like the Health Insurance Portability and Accountability Act (HIPAA) in the United States or the General Data Protection Regulation (GDPR) in the European Union [1].

Different organizations are defining diverse standards for managing log events that lack global uniformity. This is the case for the regulations to which a sector is subject in a specific country (e.g., Law 25/2007 in Spain) or the international standards such as PCI DSS (Payment Card Industry Data Security Standard). Either way, different types of logs include audit logs, whose primary purpose is to track changes made to systems or their behavior during use, and application logs, written by applications indicating what happens at runtime [2].

One of the classic challenges in log management is preserving the integrity of stored information to ensure its availability and reliability. Data can be subject to threats from third parties, such as encryption, rendering it unusable through ransomware infection, alteration, or content deletion to cover the trace of cybercriminals, as well as manipulation of information by insiders seeking to take advantage of a situation. Therefore, it is recommended to follow a proactive Zero Trust approach that ensures trust in log records about events occurring in our network rather than automatically relying on the trust of a user, device, or network [3]. Recently, blockchain technology has become popular as it provides secure, decentralized data storage. However, other inherent characteristics of this technology, such as immutability or complete traceability of the stored data, have also generated interest in its potential use. Consequently, researchers have conducted several studies on utilizing blockchain for storing certain types of data, such as access, configurations, or payments [4].

Given these precedents, the present hypothesis posits the potential use of a public blockchain to certify the integrity, immutability, and non-repudiation of the system's activity log messages. By leveraging this technology, it may be possible to establish a reliable and secure system for recording and verifying data, thereby enhancing the trust and transparency of the overall system. Therefore, this solution has the potential to be universally adopted as an openly auditable service by organizations looking to reinforce their existing logging infrastructures while ensuring critical audit events remain tamper-proof.

In the related work, several blockchain-based approaches have been proposed to ensure the integrity and immutability of log events. Some solutions, such as Exonum and Hyperledger, employ private blockchains to offer improved control; however, they encounter issues related to reduced transparency and centralized trust. Alternatively, solutions based on public blockchain, like Bitcoin, provide a high level of security but are limited by high transaction costs and scalability challenges. The proposed system addresses these limitations by implementing a decentralized, open-source log storage mechanism that utilizes public blockchains optimized for cost efficiency and transaction speed. Unlike previous methods, the proposed solution can effectively manage a substantial rate of log events on public blockchains, thereby enhancing trust, transparency, and compliance with industry regulations while ensuring practical deployment feasibility.

This work arises as a new feature of the DRACSC system, the acronym in Spanish for Automatic Recovery and Configuration Device for Communication Systems, whose objective is to facilitate and centralize typical management tasks of network equipment (e.g., applying new configurations, installing firmware, creating, or restoring backups). To illustrate the relevance of secure logging, consider a real-world use case involving a network administrator managing router configurations (Figure 1). This administrator is authorized to execute sensitive actions, including adding routes to the routing table, creating user accounts, modifying firewall rules, and enabling promiscuous mode. Each of these actions carries inherent security risks; unauthorized modifications could potentially disrupt network traffic, grant unintentional access, expose systems to threats, or facilitate the interception of sensitive data [5].

This solution is part of a larger educational project, which was introduced in routers and switches in a network laboratory at the University of Huelva (Spain) and validated by students, teachers, and IT professionals [6–8].

In summary, the contributions of this work concerning the previous ones are the following: (i) to present a novel solution that allows immutable storage and recording of log events generated by network equipment; (ii) to solve the limitations of the current state of the art in terms of visibility and transparency of log records through public blockchains; (iii) to generalize the applicability of the system in other contexts through use cases that

use transactions with and without metadata; and (iv) to provide the source code to the community in an open way to achieve transparency and move towards the establishment of a standard for log storage on the blockchain.

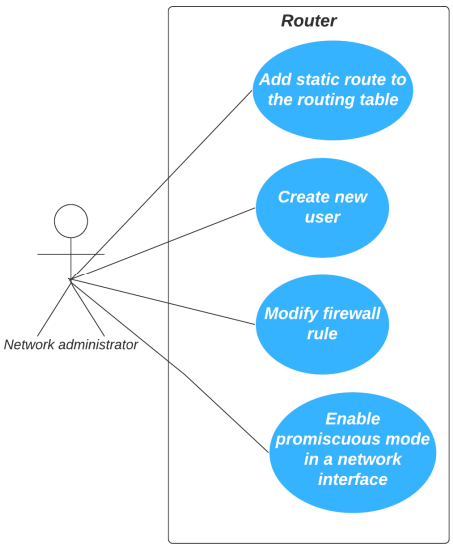

Figure 1. Use case on secure access in router management.

In order to study the feasibility of this solution, two public blockchains (i.e., Cardano and Solana) have been selected, both with a level of maturity suitable for use in a production environment. The reason for choosing these blockchains is that they allow low-cost transactions with a high number of events, such as those that commonly occur in a communication network (i.e., our test scenario). Therefore, the authors have analyzed the response times through comprehensive experimentation with different rates of events per second.

To this end, a complete software system has been developed to handle everything from data acquisition to persistence storage in public blockchains. Additionally, it includes a server for reading and processing the information effectively. This solution has been made available as open-source software, which enhances its transparency and facilitates its adoption by secure IT professionals, allowing them to establish customized functionalities and collaborate in its continuous development and improvement [9].

The article is structured as follows. Section 2 presents state-of-the-art solutions using blockchain for log storage. Section 3 describes the complete architecture of the system by separating it into three subsections with each of the different implementations. Section 4 presents the experiment and analyzes the impact of the solution in different scenarios. Section 5 provides the outcomes and corresponding discussions. Finally, Section 6 provides the conclusions, limitations, and future work.

2. Related Work

There are various studies in the state of the art that focus on the use of blockchain technology for the registration of information. In the context of this work, we highlight those solutions focused on the persistence of activity records. Log files generated by systems cover a wide range of possibilities, including user access, security incidents, and events generated by both applications and networks, among others [10]. The auditing processes also pursue the persistence of log files, which may seek to know the degree of compliance an entity has with a specific regulation [1].

Outstanding research proposes an auditable system using Exonum [11], a private blockchain with several salient features. These include using the Byzantine Fault Tolerance

(BFT) consensus algorithm, the ability to execute smart contracts, and the use of public blockchains to additionally establish periodic trustworthiness via Bitcoin. This system integrates with a Security Information and Event Management (SIEM) solution to acquire events via the Syslog standard. Its hybrid architecture guarantees the immutability and integrity of logs by storing them both on-chain, using hashes, and off-chain in a local cluster. At the same time, it allows efficient access and proper management of security events. Nevertheless, the level of transparency and reliability of the system is compromised by storing a part of the data outside the public blockchain. In addition, the use of smart contracts complicates its implementation.

A different study [12] uses Hyperledger, a private blockchain implemented to store log files encrypted by users presumably before they are sent. Storing log files directly on the blockchain raises concerns about ensuring data integrity as it becomes impossible to verify that the data have not been tampered before being included in the blockchain. Additionally, similar to the previous scenario, implementing a private blockchain reduces the transparency and reliability of the system.

In [13], a system is proposed to help auditors verify compliance with regulations using Bitcoin, a public blockchain that uses the proof of work (PoW) consensus mechanism. Data are stored both on-chain and off-chain, but the system is not designed to store events generated by network equipment. In addition, neither the price of fees nor the number of transactions per second of the Bitcoin network are scalable for use in this context.

Several authors propose BlockAudit [14], a secure and transparent system for auditing logs in charge of storing the actions executed in the system. It records on-chain data through the private Hyperledger blockchain, so it lacks transparency and reliability compared to public blockchains.

An outstanding work [15] has proposed an autonomous log storage management system for IoT devices that uses both public and private blockchains using Ethereum and Hyperledger, respectively. The system stores the content of off-chain logs and their signatures on the private blockchain, whose transactions are managed through smart contracts. A summary of these transactions is signed and stored in the public blockchain, providing greater transparency and reliability. However, there is still a risk of tampering with the blocks on the private blockchain before they are sent to the public blockchain. Additionally, the system is subject to the transaction price volatility imposed by Ethereum.

Moreover, [16] introduced the blockchain-based secure log management system for cloud computing (BCALS) using Multichain, a private blockchain that stores log messages on-chain. The data contained in the event logs are sent to Elasticsearch for further exploitation, an open-source project used to search, analyze, and visualize large amounts of information in real-time. As in previous cases, the use of a private blockchain implies a compromise in the level of transparency and reliability of the system.

Another work [17] proposes using Hyperledger as a private blockchain, where smart contracts manage the transactions and the information associated with events is stored on-chain. This open-source project is specifically designed for IoT devices and focuses on their security. However, the transparency and reliability of a private network differ significantly from those of a public blockchain.

Furthermore, [18] proposes a blockchain-based service used in supply chain and logistics management, also known as Logchain Logistics as a Service (LCaaS). This service is divided into two levels of hierarchy to achieve scalability. The first one stores signatures on a blockchain that can be either public (i.e., Ethereum) or private (i.e., IBM blockchain), while the data corresponding to the signatures is stored both off-chain and on-chain. However, the service makes it impossible to certify that said data have not been modified before inclusion when the log files are stored on the chosen blockchain.

Another study has presented a Blockchain-Enabled Scalable Network Log Management System [19]. This system utilizes Multichain to store hashes, timestamps, and ownership data while employing the InterPlanetary File System (IPFS) to store complete logs. It also features a query mechanism that enables efficient retrieval and verification of logs. This effectively addresses the limitations often encountered in other blockchain-based log management systems, which typically lack structured search capabilities. However, similar to previous examples, relying on a private blockchain may diminish the overall transparency and reliability of the system.

In [20], a decentralized data storage network is introduced, utilizing a custom-built blockchain written in Go to prevent data manipulation. This system employs a PoW consensus mechanism to maintain data integrity and uses a Gossip Protocol for automatic block repair, enabling nodes to detect inconsistencies and retrieve valid copies of the blockchain from their peers. The data are stored on-chain to ensure immutability and an API Gateway facilitates log transactions through RESTful API endpoints. Nevertheless, being a private and custom blockchain raises concerns regarding transparency and reliability.

Finally, a blockchain-based audit log system has been proposed to ensure the data integrity of log files [21]. The system assumes that both loggers and auditors may be untrustworthy and mitigates these risks by utilizing smart contracts within Hyperledger Fabric. This method presents a way to generate on-chain integrity proofs using Non-Fungible Tokens (NFT), with log files stored off-chain in the IPFS. While this system enhances scalability and security, it is limited to a permissioned blockchain, which may decrease transparency compared to public blockchain solutions.

Table 1 presents a comparison of the previous solutions compared to the approach presented in this paper. It should be noted that only our solution and the one presented in [11] use the Syslog standard as a data source to ensure compatibility and interoperability with numerous tools and applications designed to work with this standard. It is also worth noting that most of the proposed works use private networks due to the cost of storing data in public blockchains, implying less transparency in the process. On the other hand, only [11,15,19] use some additional tools to visualize the stored information in a user-friendly way, as in the proposed solution. Finally, it is worth highlighting the open-source nature of the proposed system together with the proposal of [18], which allows transparency and facilitates project release for reproducibility within the scientific community, aiming to establish a standard for storing registries in the blockchain.

Table 1. Related work.

Reference	Data Storage	Kind of Blockchain	Consensus Algorithm	Transfer	Specific for Logging	Data Visualization	Multi Blockchain	Open Source	Year
[11]	Off-chain/On-chain	Private	BFT	Direct	✔	Web app	No	No	2019
[12]	On-chain	Private	BFT	Direct	✔	No	No	No	2018
[13]	Off-chain/On-chain	Public	PoW	Direct	✔	No	No	No	2017
[14]	On-chain	Private	BFT	Direct	✔	No	No	No	2018
[15]	Off-chain/On-chain	Both	PoW	Smart Contracts	✔	Web app	No	No	2020
[16]	On-chain	Private	PBFT	Direct	✔	Elasticsearch	No	No	2021
[17]	On-chain	Private	PBFT	Smart Contracts	No	No	No	No	2022
[18]	Off-chain/On-chain	Private	BFT	Direct	✔	No	✔	✔	2018
[19]	Off-chain/On-chain	Private	Permission-based mining	Direct	✔	Web app	No	No	2022
[20]	On-chain	Private	PoW	Direct	✔	No	No		2024
[21]	Off-chain/On-chain	Private	PBFT	Smart Contracts	✔	No	No	No	2024
Authors	Off-chain/On-chain	Public	Various *	Direct	✔	Web app	✔	✔	2024

* PoH + PoS for Solana and PoS for Cardano.

Finally, a comparative overview of blockchain applications is provided in Table 2 to emphasize the key aspects discussed in this section. Public blockchains, including Bitcoin, Ethereum, Cardano, and Solana, are characterized by open participation and decentralized governance. For instance, Bitcoin employs a PoW mechanism, allowing for approximately seven transactions per second (TPS). However, it experiences high and unpredictable fees due to market-driven congestion and offers limited smart contract functionality. Ethereum has transitioned to a PoS consensus model, achieving around 15 TPS. Despite this improvement, transaction fees remain high and variable because of a dynamic gas market, although it supports comprehensive smart contracts. In contrast, Cardano utilizes a deterministic fee model within its PoS system, reaching approximately 250 TPS while maintaining low fees and full smart contract capabilities. Solana enhances performance with a hybrid PoH and PoS mechanism, enabling the processing of up to 65,000 TPS at low and variable fees. On the private blockchain side, platforms such as Multichain, Exonum, Hyperledger Fabric, and IBM Blockchain are designed for enterprise applications. These systems generally feature configurable consensus mechanisms and administratively controlled fee structures, which can result in deterministic fees, sometimes as low as zero. However, standard TPS figures are not reported for these platforms, as they depend on specific configurations and the number of nodes utilized [22].

Table 2. Comparison between the several blockchains studied in the related work.

Blockchain	Kind of Blockchain	Consensus Algorithm	TPS	Transaction Fees	Deterministic Fees	Smart Contract Support	Year Launched
Bitcoin	Public	PoW	7	High	No	Limited	2009
Ethereum	Public	PoS	15	High	No	✔	2015
Multichain	Private	Several	*	*	✔	Limited	2015
Exonum	Private	BFT	*	*	✔	✔	2016
Hyperledger Fabric	Private	Several	*	*	✔	✔	2016
Cardano	Public	PoS	250	Low	✔	✔	2017
IBM Blockchain	Private	Several	*	*	✔	✔	2017
Solana	Public	PoH + PoS	65,000	Low	No	✔	2020

* These features on private blockchain depend on the configuration.

3. Materials and Methods

This section outlines the system components, types of transactions used for certifying log events, and blockchains deployed.

3.1. Components of the System

This section describes the components used in the system and their relationship, which are illustrated in Figure 2. The process is broken down into four steps from left to right: (1) the managed equipment (e.g., router, switch or PC) sends a log event to the Syslog server located in the DRACSC system; (2) the events are then queued in the message broker to be further processed by the log module; (3) depending on the blockchain used, the log module will perform specific tasks such as obtaining the hash of the event information, connecting to a blockchain node and sending the corresponding transaction to the network; and (4) when the hash obtained in the previous step is available and signed, it is sent to the log management server for storage and further exploitation.

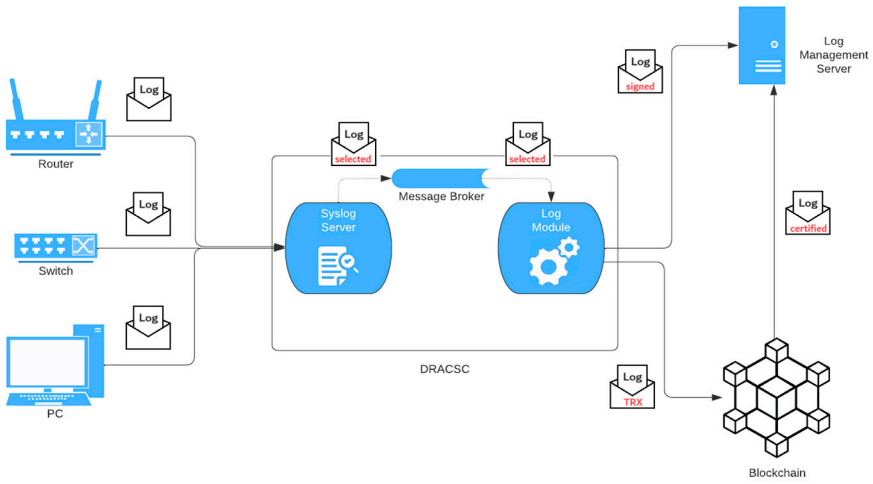

Figure 2. System components.

3.1.1. Managed Equipment

Network devices such as routers, switches, computers, servers or APs continuously log events and send corresponding log messages to the Syslog server.

3.1.2. Syslog Server

This component is responsible for receiving, filtering, transforming, storing, or redirecting log records generated by the managed network equipment. It is relevant to highlight that, to operate, the network equipment must activate a Syslog event notification service based on RFC 5424 [23], a widely accepted standard. Typically, the configuration process is straightforward. In our case, the Syslog server was installed in the DRACSC system. An open-source version of Syslog-ng has been chosen for this work, specifically version 4.1.1. This software can easily filter the event logs by utilizing attributes such as the severity level, IP address, or payload. This possibility of filtering optimizes log delivery, enhances profitability, and enables the adaptation of our solution to public blockchain performance. Eventually, the data received by the Syslog server is forwarded to the message broker via the advanced message queuing protocol (AMQP) for further processing in subsequent stages.

3.1.3. Message Broker

The purpose of this component is to temporarily store event messages while the log module is handling them. The message broker was implemented in the DRACSC system using RabbitMQ, specifically version 3.10.5. It is based on AMQP, which brings the advantage of scalability and asynchrony [24]. Additionally, the software offers other benefits, such as decoupling with other system services or assuring that events will not be lost due to the public blockchain being highly congested at a specific moment or failures in the log module service that the broker feeds. Other factors influencing their choice include a lightweight design, which is crucial because of hardware resource limitations and ease of use for prototyping.

3.1.4. Log Module

This service resides on the DRACSC system and receives the log records filtered through the message broker. Once received, it processes them to adapt them for both the blockchain format and the log management server. The programming language used is Node.js, specifically version 18.21. The choice was motivated by its high performance and asynchrony, as it is event-driven [25].

3.1.5. Blockchain

A blockchain is a distributed ledger technology (DLT). It is a database that can be shared by many entities in a peer-to-peer manner and allows information to be stored in an immutable and ordered way. In general, once a transaction has occurred, it cannot be altered or deleted. One of the fundamental elements is the consensus algorithm used, which is responsible for validating the information added to the blockchain between all the nodes that make it up and ensuring that all transactions are correct. There are a multitude of consensus algorithms, such as the two related to this work (i.e., Proof of History and Proof of Stake), which are described in the section on implemented blockchains. It is important to note that once the transaction is sent to the blockchain, it persists after the nodes that make up the network validate the transaction and add its information to a block after a few transactions established by the network.

3.1.6. Log Management Server

This module is a web service designed to receive Syslog messages and blockchain transaction data through a RESTful API, which are then stored for later exploitation. The server is also responsible for communicating with the blockchain to retrieve the timestamp of the generated block that permanently records the transaction (i.e., the certification of a log event). Moreover, this server offers an intuitive interface for easy access to the stored information as seen in Figure 3. To this end, Angular, an open-source JavaScript framework for the web interface, has been used in the development, specifically version 14. On the server side, the service uses version 18.21 of Node.js.

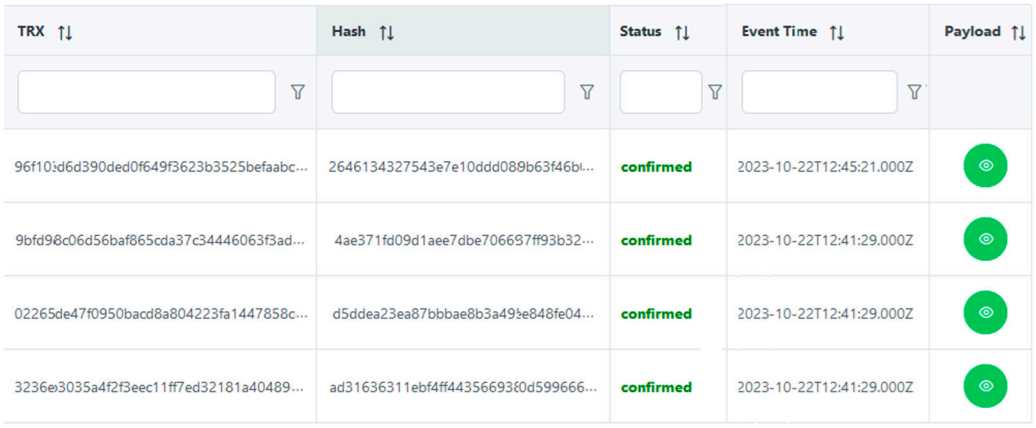

Figure 3. Main view of the log management server interface.

3.2. Types of Transactions

This section details two different implementations aimed at providing a universal, blockchain-independent solution for certifying registration events. One approach involves using a blockchain that can append metadata to transactions, thereby including the hash of the event generated by the managed equipment. The second approach provides a solution using a blockchain that avoids attaching metadata to transactions.

When an event from a managed network device arrives at the Syslog server, it is automatically forwarded to the message broker. Once there, the event will wait for the log module service to process it for subsequent transfer to the blockchain. When the blockchain selected does not allow metadata, the log module will perform a transaction with the blockchain that will require the payment of a certain number of tokens. The transaction ID generated in response will be used as the key, and the content of the log event as the message, thus letting us calculate a SHA2-based hash message authentication key (HMAC) [26]. These three elements (i.e., log event, transaction ID and resulting hash)

are then sent to the log management server for safekeeping. Finally, the log module will also send confirmation to the message broker indicating that the log event has been registered and is waiting for further events. Figure 4 shows the interaction process, where the managed device (e.g., a router) that has the Syslog protocol configured generates the event with a size of 256 bytes (e.g., incorrect login access).

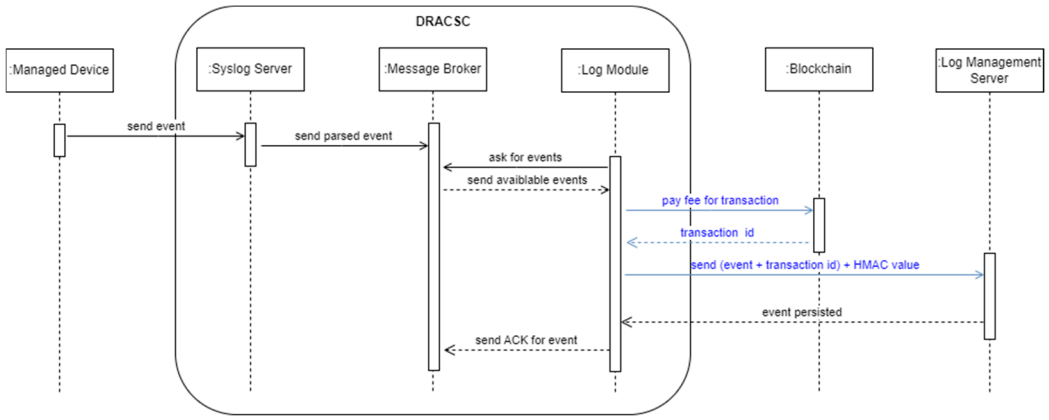

Figure 4. Storage of a log event using a transaction without metadata.

When the blockchain allows metadata, the log module receives a log event and calculates a hash of the event's content. This hash is then included in the metadata field before the corresponding transaction is initiated, and the result will persist in the blockchain once the block is completed. As in the previous case, a transaction ID will be received and sent to the log management server along with this event's hash and content. Figure 5 illustrates the interaction process, which differs from the previous method after the log module. Specifically, the log module will generate a SHA2 hash of the event and perform the transaction in the blockchain, adding the hash to the transaction's metadata and obtaining a transaction ID. Once the transaction ID is received, the log module will send the log event, transaction ID, and resulting hash as in the previous case.

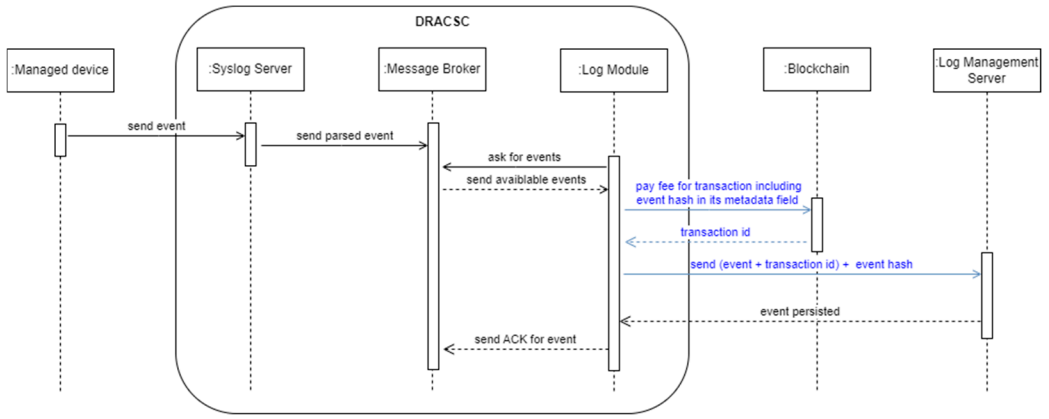

Figure 5. Storage of a log event using a transaction with metadata.

3.3. Case Studies

This section provides a concise depiction of the two blockchains examined in this manuscript, with a comparative analysis delineated in Table 3.

Table 3. Comparison of deployed blockchains.

Name	Token	Metadata	TPS	Consensus Algorithm	Deterministic Fee	Average Transaction Cost	Average Transaction Price (€) *	Release Year
Solana	SOL	No	65.000	PoH, PoS	No	5.001×10^{-6} SOL	0.019	2020
Cardano	ADA	✔	250	PoS	✔	0.2 ADA	0.064	2017

* Average price during the study in 2024.

3.3.1. Solana

This is an open source blockchain platform launched in 2020 [27]. As usual, among blockchains that do not include metadata in transactions, Solana offers a high processing capacity measured in transactions per second (TPS), exceeding 60,000 operations at maximum for all users [28]. It uses a combination of two consensus algorithms. Proof of History (PoH) generates chained timestamps to cryptographically certify the time and order of events on the blockchain. Additionally, it uses Proof of Stake (PoS) to decide which blocks should be added to the network.

The blockchain's native token is SOL, but transactions are measured in Lamports. The smallest unit of measurement in the Solana network is Lamport, equivalent to 10^{-9} SOL. It is important to note that the transaction fee cost depends on network congestion, so it is not deterministic.

To enable programmatic interaction with this blockchain, the official library 'solana-web3.js' has been used to communicate with Solana's Remote Procedure Call (RPC) JavaScript Object Notation (JSON) API. This same library was used by both the log module and the log management server. One of the key features of this library is that it establishes the connection with the blockchain completely, simplifying the process by avoiding using other additional software packages (e.g., installation of a blockchain node or wallet management software).

Several functions have been implemented for the operation of the log module: the 'initSolana' function is in charge of establishing the connection with the blockchain (i.e., the testnet in our case); the 'createWallet' function creates a new wallet if it does not already exist; the 'writeLog' function sends a transaction to the blockchain to store a log event; and the 'getBalance' function checks the amount of SOL tokens available in the wallet. Moreover, the log management server also uses the 'solana-web3.js' library to check if the block containing the log record persists correctly on the blockchain.

3.3.2. Cardano

This project, launched in 2017 [29], is also open source. Its blockchain allows the inclusion of metadata in transactions, which implies a lower transaction rate per second due to the higher complexity. This results in 250 TPS at maximum for all users [30]. Cardano uses the Ouroboros consensus algorithm based on PoS, where validator nodes are selected based on the number of tokens they hold. The native token used in this network is ADA, and the smallest unit of measurement is known as Lovelace, which is equivalent to 10^{-6} ADA. This blockchain uses the Unspent Transaction Output (UTxO) scheme, which means that the transaction rate will be deterministic, allowing for more accurate planning.

The official library 'cardano-wallet-js' has been used to carry out the interaction between the log module and the log management server with the block network. This library communicates with a blockchain node via the Cardano Wallet software (v2023-07-18). This node and the Cardano Wallet run in the cloud in conjunction with the log management server, so it plays an active role in the Cardano network by maintaining a copy of the blockchain, participating in validation, and propagating both transactions and blocks. Furthermore, the Cardano Wallet allows secure management of transactions and wallets.

As a major disadvantage, using these three elements implies greater implementation complexity than the Solana network.

The log module has various functions for interacting with the Cardano network. The 'initCardano' function connects to the Preprod network (i.e., testnet), while the 'getBalance' function checks the available ADA amount. Additionally, the 'createWallet' and 'writeLog' functions have been implemented to support backward compatibility with the previous implementation. The log management server uses these three elements to verify if the information has persisted in a block on the blockchain.

4. Experimentation

To assess the feasibility of the secure log storage, this section presents both the procedures implemented in the stress tests conducted and presents the results derived from the analysis of the test data acquired.

The hardware and software components were implemented within a local network to minimize latency in the scenario used, except for the public blockchains. On the one hand, there is a Raspberry Pi 4 Model, which uses a Broadcom BCM2711 Quad core Cortex-A72 (ARM v8) 64-bit SoC 1.5 GHz processor and 4 GB of LPDDR4-3200 SDRAM running the Raspberry Pi OS 5.10. This OS version is a Debian variant optimized for Raspberry Pi. On the other hand, a server with an Intel Core i7-10870H 2.20 GHz processor and 16 GB of DDR4 RAM running the Ubuntu 22.0.4.3 LTS operating system was used to install the Syslog-ng server, the RabbitMQ Message Broker, and the log module. This server supported the log server management, as well as the node and wallet in the Cardano scenario.

It is important to note that in the log module, which communicates directly with the message broker, several tests have been performed depending on the prefetch (i.e., the number of messages in the queue that are consumed in a request). For instance, in case it is set to 1, it will wait until all the processing is finished before consuming the second message. Given the asynchronous nature of NodeJS, when blocking operations (e.g., HTTP requests, disk accesses, etc.) are encountered in the processing of each message, this language could continue with the actions of the second message until the previous one is unblocked, which is a significant improvement in efficiency, as can be seen in Figure 6, an example execution with a prefetch of 3 events. As shown in the example, after retrieving the events from the message broker, the potential requests that could cause blockages are sent to the blockchain and the log management server. However, certain factors beyond our control, such as high congestion in the blockchain or temporary network outages, may impact these requests. In these cases, NodeJS will continue executing other instructions regardless of the blocked flow. The optimal prefetch value will depend on the specific scenario, considering factors such as the number of logs per second being processed or log module instances. Achieving an equilibrium between these variables will provide optimal system performance.

A comprehensive experiment was conducted using Loggen, a high-performance log generation tool designed for stress testing syslog servers. This tool enables benchmarking and load testing by generating and transmitting syslog messages at configurable rates. The UDP protocol was utilized during the tests, and the maximum average message size parameter was set to 8192 bytes. A single instance of the log module was used, with prefetch values of 1, 10, 20, and 50 tested for loads of 1 event/s, 10 events/s, 20 events/s, and 50 events/s on both blockchains. We also evaluated several processing times, such as the processing time within the log module, the time from when the event is generated until it is certified in the blockchain, the time from when the event is generated until it has persisted in the log management server, and the total execution time of each test bench. We performed 10 iterations for each case to average values and used the NodeJS performance API with millisecond level resolution to measure time.

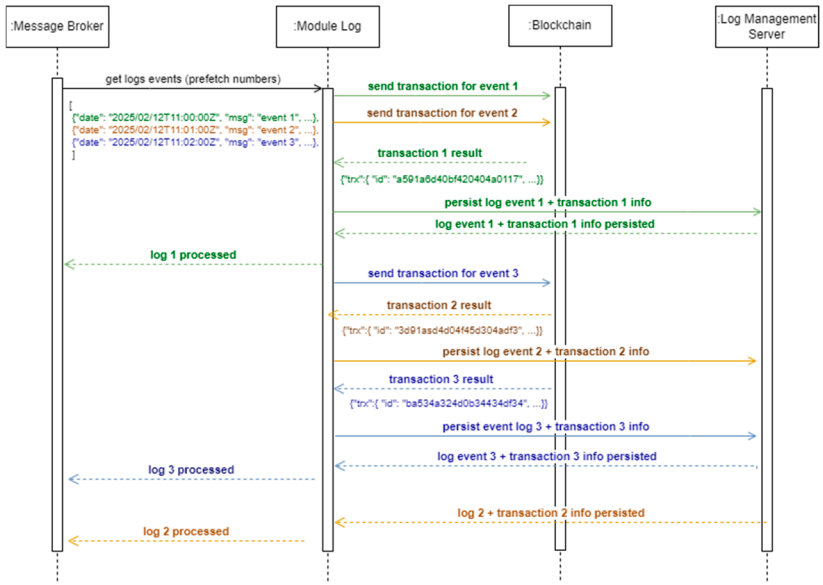

Figure 6. Example of asynchronous execution of messages in the experimentation carried out.

5. Results and Discussion

This section analyzes test results on Solana and Cardano blockchains, comparing average processing times at various event rates. Subsequently, the section examines enterprise log data management, blockchain certification costs, and suggests advanced filtering to reduce expenses by focusing on crucial logs.

5.1. DRACSC Performance Test

The results of the tests on the Solana blockchain indicate that the average time for a request per second is over 12.82 s for 1 event/s, 17.94 s for 10 events/s, 22 s for 20 logs/s, and 31.77 s for 50 events/s, as shown in Figure 7. Table 4 illustrates the average outcomes for 10 iterations and their standard deviation. The "Log Module" column represents the event's duration in the log module service. The "From event to blockchain" column measures the interval from an event's arrival at the Syslog server until its recording on the blockchain. The last column, "From event to Log Management Server" indicates the time taken from the event's arrival until its persistence in the log management system.

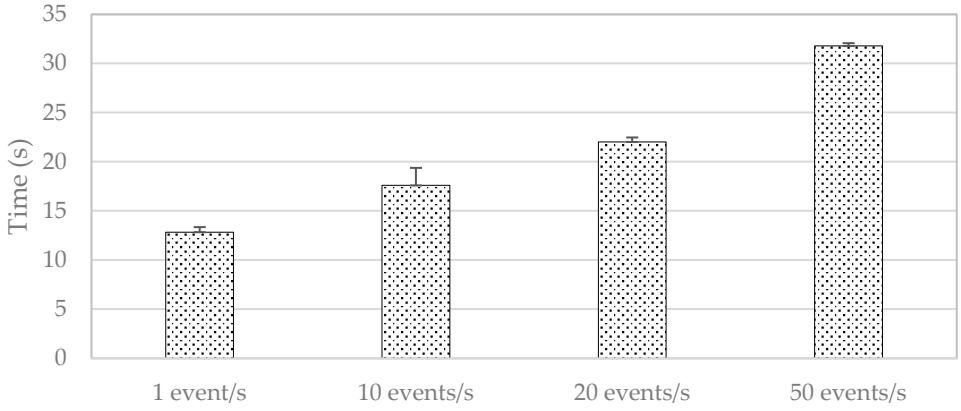

Figure 7. Execution times with the Solana blockchain.

Table 4. Average time for a request in Solana.

Events/Second	Log Module (ms)	From Event to Blockchain (ms)	From Event to Log Management Server (ms)
1	5.45 ± 4.69	12,836.45 ± 293.33	12,824.20 ± 287.71
10	5.99 ± 3.32	1835.59 ± 1192.15	14,810.23 ± 1056.99
20	5.26 ± 4.08	3866.75 ± 2430.11	15,532.33 ± 1234.68
50	5.17 ± 4.41	7546.99 ± 5501.71	18,623.97 ± 10,797.62

The experimental results on Solana indicate that the average processing time for individual requests remains relatively stable across different event rates. Nonetheless, as the event rate increases, there is a noticeable increase in time, especially in the interaction of the event with the blockchain. This suggests that while Solana demonstrates consistency in processing individual requests, the system's scalability and responsiveness may be challenged under higher loads of concurrent events. On the other hand, the results suggest that the system appears well suited for environments that require certification for a moderate number of critical logs; however, further optimization strategies should be considered for scenarios involving increased event volumes. Complementarily, higher hardware resources would be required to support the proposed system if the number of module log instances exceeds this limit due to the limitations of the Raspberry Pi.

The results found with the Cardano blockchain that supports metadata achieved 4.35 s for 1 event/s, 626.55 s for 10 events/s, 1098.45 s for 20 events/s, and 3071.65 s for 50 events/s, as shown in Figure 8. The average results for the 10 iterations and their standard deviation can be seen in Table 5. The experimentation reveals that the log module processing times fluctuate with increasing event rates. The times from event to blockchain and to the log management server experience significant increases, particularly under higher event rates. This suggests that Cardano exhibits longer processing times for individual requests than Solana, and scalability challenges become more evident in scenarios with increased concurrent events. Consequently, Cardano appears viable only in scenarios where certification events occur very infrequently. This is due to two factors: on the one hand, there is a restriction on the number of transactions per second imposed by the Cardano network. On the other hand, Cardano Mempool's deliberate design prevents certain actors from monopolizing the entire blockchain bandwidth by processing transactions in a FIFO queue. In conclusion, optimization measures may need to be implemented to improve overall system responsiveness.

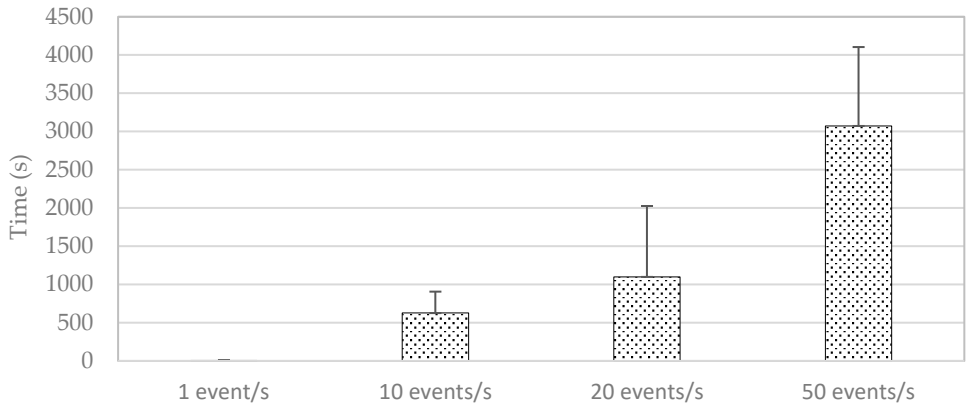

Figure 8. Execution times with the Cardano blockchain.

Table 5. Average time for a request in Cardano.

Events/Second	Log Module (ms)	From Event to Blockchain (ms)	From Event to Log Management Server (ms)
1	16.57 ± 6.48	22,171.70 ± 30,389.99	4357.8 ± 6855.37
10	7.59 ± 4.43	240,995.88 ± 164,862.90	219,455.22 ± 167,407.68
20	7.61 ± 4.58	313,123.15 ± 302,980.38	305,814.52 ± 304,370.36
50	4.95 ± 2.96	298,423.43 ± 246,633.25	274,762.13 ± 245,587.09

5.2. Feasibility in an Enterprise System

The proposed solution is designed to effectively manage the large volume of log data generated within an enterprise system. However, the cost of sending transactions to the blockchain for certification remains a concern. In the case of Cardano, transaction costs are influenced by two components as shown in Figure 9: the transaction fee and the UTxO storage cost [31]. In our study, only one SHA2 hash was stored for each transaction, resulting in a fixed size of 32 KB with a corresponding cost of 200,000 lovelaces. In contrast, transaction costs on the Solana blockchain vary based on network congestion; therefore, they are non-deterministic [32]. These costs are calculated as the sum of a base fee and a priority fee as shown in Figure 10. Costs increase when more compute units (CU) are consumed, reflecting the computational resources used by the transaction on the network. In our study, the average cost was 5001 lamports per transaction.

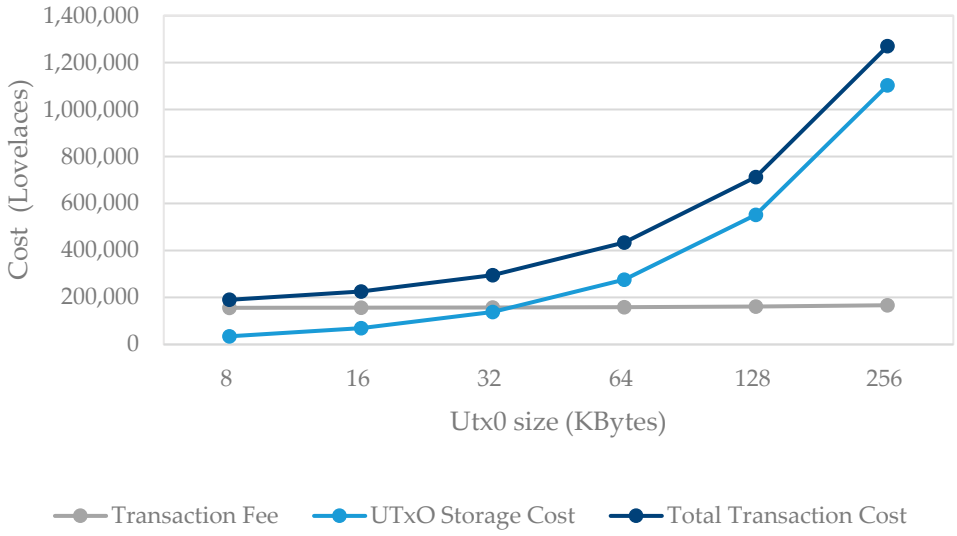

Figure 9. Transaction cost breakdown in Cardano.

The rate of logs generated per second varies for each company, but for the purpose of this discussion, we will consider a rate of one log per second, which implies around 86,400 logs daily [11]. Approximately 10% of these logs are critical, which equates to 8640 logs per day. Through experimentation, we have found that sending all critical logs is not affordable due to cost constraints, specifically 164.16 € per day in Solana and 552.96 € per day in Cardano. To minimize costs, it is advisable to operate during periods of low congestion or reduce the transaction priority, which allows taking advantage of the lowest fees available. To further mitigate this issue, we can employ the advanced filtering functions of Syslog-ng to selectively filter the most crucial logs, thereby reducing costs while ensuring that the most important events are certified.

Although the costs may be substantial, they are justifiable given the critical importance of data integrity, legal compliance, and security, particularly in industries such as finance

and healthcare, where non-repudiation and long-term verifiability are paramount. In the finance sector, immutable log storage plays a crucial role in preventing fraud and adhering to regulatory standards like PCI DSS [33]. The investment in this technology can be rationalized by selectively applying blockchain to high-risk transactions. Likewise, in healthcare, it is essential to uphold the integrity of electronic health records (EHR) and manage access to sensitive medical information in order to comply with regulations like HIPAA [34].

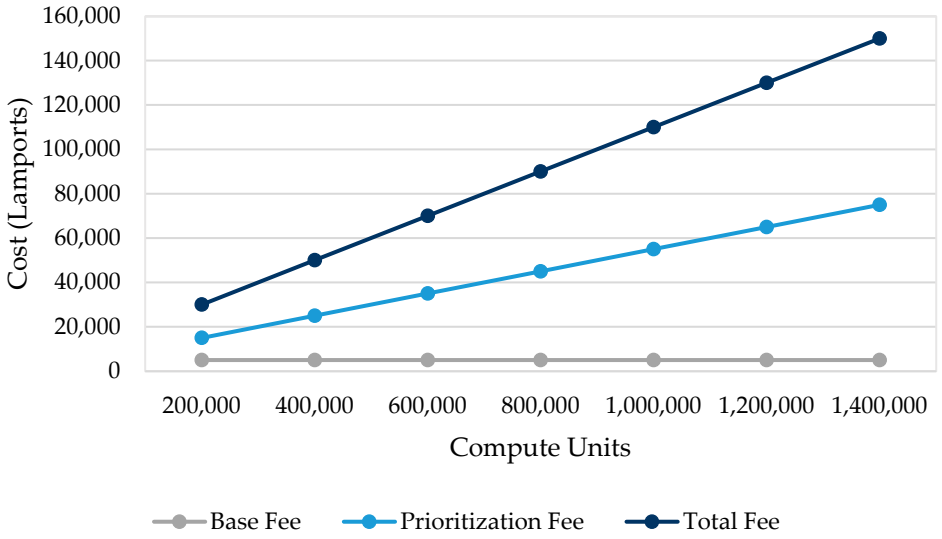

Figure 10. Transaction cost breakdown in Solana.

6. Conclusions

The secure storage of log events is a complex task, becoming increasingly important due to the complete digitalization of production processes. This is related to different factors, such as the growing complexity of IT systems, society digitization, or the use of technology in signature processes. Traditionally, these events produced by computer systems have been used internally as a reference for detecting erroneous behavior or tracing operations. Therefore, it could be challenging to use in a judicial process due to the doubts raised by these data that do not comply with the principles of integrity, immutability, or non-repudiation.

The DRACSC system introduced a new blockchain-based approach to overcome the limitations of traditional log methods used for communication networks. This solution is designed to address common problems in managing communication equipment, such as error recovery, device configuration, and certification of critical log events generated by company network components.

The research produced mixed outcomes during the experimentation phase. On the one hand, the methodology followed fulfilled its intended function of preserving coherence, persistence and non-repudiation of the data. However, considering the inherent characteristics of blockchains, the choice of implementation becomes a crucial factor in achieving satisfactory performance. Solana demonstrated exceptional performance, establishing itself as a viable choice for implementing this solution with favorable times for a moderate number of events and lower transaction costs. Conversely, Cardano's feasibility is somewhat limited, as it exhibits higher response times for this specific use case. This limitation significantly reduces the number of events, resulting in higher transaction costs. However, it is worth noting that Cardano's capability to include metadata in transactions adds substantial value for certifying log events. Therefore, each option caters to opposing scenarios, and this duality should be regarded as an advantageous aspect

of the study. The main limitation of the proposed approach lies in the bandwidth and congestion issues associated with public blockchains. Despite this, the selection of a new network with optimal performance could be compromised in the medium term due to the widespread adoption of blockchain technology. This work also contributes by making the research code accessible as open source, promoting transparency and collaboration within the scientific community. The code will be publicly accessible through the specified repository at https://github.com/jdmorei/seclogs (accessed on 26 February 2025), encouraging collaboration and advancements in the field.

Future work will focus on enhancing security, cost efficiency, and scalability. Regarding security, one of the next developments will be creating a blockchain migration tool due to advancements in classical and quantum computing, which poses risks to existing blockchain protocols. This tool will facilitate secure data and state transfers between blockchains and act as a contingency plan for transitioning to more secure platforms if vulnerabilities arise or if quantum computing threatens current cryptographic methods, thereby ensuring continuity and data integrity.

In parallel with these security enhancements, another goal is to increase the number of messages per second and reduce costs. One potential option would be the use of a layer-2 blockchain such as Cardano Hydra. This network improves Cardano's scalability and performance, reaching 10^6 TPS and introducing the concept of microtransactions at a low cost, making it well-suited for high-frequency payment systems. Hydra achieves this efficiency through State Channels, a technique used in blockchain networks to reduce the number of transactions required for each interaction on the main chain. State Channels are smart contracts that apply predefined rules for managing transactions between parties. Said channels originate from the main blockchain and are merged back into them, verifying their validity through cryptographic methods. The previous features and the ability to add metadata to transactions, thanks to the UTxO schema, are the reasons we will research Cardano Hydra. In addition, other technologies are also being explored, such as DAG (Directed Acyclic Graphs), a mathematical and computational construct predating the blockchain concept with many functional similarities. In our case, we will study the use of IOTA 2.0, which uses the Nakamoko Consensus on a DAG model. This concept brings three main features: parallel writing, where blocks can be validated and added in parallel; on tangle voting, where each validating node has voting power thanks to staking and validation; and approval weight where, in cases of double-spend, the transaction with higher weight will win. These features create a seamless payment network, making it particularly attractive for micropayments and IoT-based transactions. Given the current extensible architecture of the software created for this work, integrating these technologies will require only the development of new connectors. This flexibility ensures that the system remains adaptable to emerging blockchain advancements while maintaining compatibility with existing infrastructures. Therefore, integrating Cardano Hydra and IOTA 2.0 will overcome these limitations by lowering transaction costs and supporting high-frequency transactions with minimal latency.

Finally, we will explore advanced filtering techniques that integrate machine learning (ML) to prioritize critical logs and reduce unnecessary blockchain transactions. Techniques such as anomaly detection and risk-based classification, together with the Syslog-ng filters, could dynamically determine which logs require blockchain certification, optimizing both security and cost-effectiveness.

Author Contributions: Conceptualization, J.D.M.R. and T.J.M.S.; methodology, T.J.M.S.; software, J.D.M.R.; validation, T.J.M.S.; formal analysis, T.J.M.S.; investigation, J.D.M.R. and T.J.M.S.; resources, J.D.M.R. and T.J.M.S.; data curation, J.D.M.R. and T.J.M.S.; writing—original draft preparation, J.D.M.R.; writing—review and editing, T.J.M.S.; visualization, J.D.M.R. and T.J.M.S.; supervision,

T.J.M.S.; project administration, T.J.M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Dataset available on request from the authors.

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

Abbreviations

The following abbreviations are used in this manuscript:

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
AP	Access Point
BCALS	Blockchain-based Secure Log Management System for Cloud Computing
BFT	Byzantine Fault Tolerance
CU	Compute Unit
DAG	Directed Acyclic Graph
DLT	Distributed Ledger Technology
EHR	Electronic Health Record
FIFO	First In, First Out
GB	Gigabyte
GDPR	General Data Protection Regulation
HIPAA	Health Insurance Portability and Accountability Act
HMAC	Hash-based Message Authentication Code
HTTP	Hypertext Transfer Protocol
IPFS	InterPlanetary File System
IoT	Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation
KB	Kilobyte
LCaaS	Logchain Logistics as a Service
LTS	Long-Term Support
NFT	Non-Fungible Token
ML	Machine Learning
OS	Operating System
PBFT	Practical Byzantine Fault Tolerance
PC	Personal Computer
PCI DSS	Payment Card Industry Data Security Standard
PoH	Proof of History
PoS	Proof of Stake
PoW	Proof of Work
RAM	Random-Access Memory
RFC	Request for Comments
RPC	Remote Procedure Call
SDRAM	Synchronous Dynamic Random-Access Memory
SIEM	Security Information and Event Management
SoC	System on a Chip
TPS	Transactions Per Second
UTxO	Unspent Transaction Output

References

1. Vazão, A.P.; Santos, L.; Costa, R.L.d.C.; Rabadão, C. Implementing and evaluating a GDPR-compliant open-source SIEM solution. *J. Inf. Secur. Appl.* **2023**, *75*, 103509. [CrossRef]
2. Khan, S.; Gani, A.; Wahab, A.W.A.; Bagiwa, M.A.; Shiraz, M.; Khan, S.U.; Buyya, R.; Zomaya, A.Y. Cloud Log Forensics: Foundations, State of the Art, and Future Directions. *ACM Comput. Surv.* **2017**, *49*, 6. [CrossRef]
3. Rose, S.; Borchert, O.; Mitchell, S.; Connelly, S. *NIST SP 800-207: Zero Trust Architecture*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2020; p. 10.
4. Silva, R.; Inácio, H.; Marques, R.P. Blockchain implications for auditing: A systematic literature review and bibliometric analysis. *Int. J. Digit. Account. Res.* **2022**, *22*, 163–192. [CrossRef] [PubMed]
5. Bellovin, S.M.; Bush, R. Configuration management and security. *IEEE J. Sel. Areas Commun.* **2009**, *27*, 268–274. [CrossRef]
6. Sanguino, T.M.; González, I.F.V.; Fernández, J.E.; Domínguez, A.G. Using Identity Provider and Automatic Resource Management to Improve a Remote Networking Lab. *IEEE Lat. Am. Trans.* **2018**, *16*, 1547–1556. [CrossRef]
7. Morillo Reina, J.D.; Mateo Sanguino, T.J. Portable Device for Easy Management and Automatic Recovery of Networking Systems. *IEEE Lat. Am. Trans.* **2019**, *17*, 401–408. [CrossRef]
8. Morillo Reina, J.D.; Mateo Sanguino, T.J. Cloud-Based Automatic Configuration and Disaster Recovery of Communication Systems Applied in Engineering Training. *Electronics* **2024**, *13*, 4203. [CrossRef]
9. Russo, D. Benefits of Open Source Software in Defense Environments. In Proceedings of the 4th International Conference in Software Engineering for Defence Applications, Rome, Italy, May 2016; Advances in Intelligent Systems and Computing. 2016; Volume 422.
10. Landauer, M.; Skopik, F.; Wurzenberger, M.; Rauber, A. System log clustering approaches for cyber security applications: A survey. *Comput. Secur.* **2020**, *92*, 101739. [CrossRef]
11. Putz, B.; Menges, F.; Pernul, G. A secure and auditable logging infrastructure based on a permissioned blockchain. *Comput. Secur.* **2019**, *87*, 101602. [CrossRef]
12. Landauer, M.; Skopik, F.; Wurzenberger, M.; Rauber, A. EngraveChain: A Blockchain-Based Tamper-Proof Distributed Log System. *Future Internet* **2021**, *13*, 143. [CrossRef]
13. Sutton, A.; Samavi, R. Blockchain Enabled Privacy Audit Logs. In *The Semantic Web—ISWC 2017*; Springer: Cham, Switzerland, 2017; Volume 10587, pp. 645–660.
14. Ahmad, A. Towards Blockchain-Driven, Secure and Transparent Audit Logs. In Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous '18), New York, NY, USA, 5–7 November 2018; pp. 443–448.
15. Hsu, C.L.; Chen, W.X.; Le, T.V. An Autonomous Log Storage Management Protocol with Blockchain Mechanism and Access Control for the Internet of Things. *Sensors* **2020**, *20*, 6471. [CrossRef] [PubMed]
16. Ali, A.; Khan, A.; Ahmed, M.; Jeon, G. BCALS: Blockchain-based secure log management system for cloud computing. *Trans. Emerg. Telecommun. Technol.* **2021**, *33*, e4272. [CrossRef]
17. Na, D.; Park, S. IoT-Chain and Monitoring-Chain Using Multilevel Blockchain for IoT Security. *Sensors* **2022**, *22*, 8271. [CrossRef] [PubMed]
18. Pourmajidi, W.; Miransky, A. Logchain: Blockchain-Assisted Log Storage. In Proceedings of the IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2–7 July 2018; pp. 978–982.
19. Rakib, M.H.; Hossain, S.; Jahan, M.; Kabir, U. A Blockchain-Enabled Scalable Network Log Management System. *J. Comput. Sci.* **2022**, *18*, 496–508. [CrossRef]
20. Putra, R.A.; Ardiansyah, R.; Pusadan, M.Y.; Kasim, A.A.; Joefrie, Y.Y. Developing Decentralized Data Storage Network Using Blockchain Technology to Prevent Data Alteration. *Adv. Sustain. Sci. Eng. Technol.* **2024**, *6*, 02401017. [CrossRef]
21. Liu, Z.; Zhang, X.; Li, G.; Cui, H.; Wang, J.; Xiao, B. A Secure and Reliable Blockchain-Based Audit Log System. In Proceedings of the IEEE International Conference on Communications, Denver, CO, USA, 9–13 June 2024; pp. 2010–2015.
22. Dinh, T.T.A.; Wang, J.; Chen, G.; Liu, R.; Ooi, B.C.; Tan, K.L. BLOCKBENCH: A framework for analyzing private blockchains. In Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD '17), Chicago, IL, USA, 14–19 May 2017; pp. 1085–1100.
23. Ćatović, A.; Buzađija, N.; Lemes, S. Microservice development using RabbitMQ message broker. *Sci. Eng. Technol.* **2022**, *2*, 30–37. [CrossRef]
24. Luo, J.; Zhou, B.; Zheng, Y.; Pan, W. *Research on High Performance Web Service Construction Method Based on JavaScript Asynchronous Programming Technique*; Sciendo: Warsaw, Poland, 2024. [CrossRef]
25. Sahni, N. A review on cryptographic hashing algorithms for message authentication. *Int. J. Comput. Appl.* **2015**, *120*, 29–32. [CrossRef]
26. Gerhards, R. *RFC 5424: The Syslog Protocol*; RFC Editor: Marina del Rey, CA, USA, 2009.

27. Hebooks. *The Complete Solana Guide: All You Need to Know About SOL Crypto Before Investing*; Amazon Digital Services LLC—Kdp: Washington, DC, USA, 2023; p. 10.
28. Li, X.; Wang, X.; Kong, T.; Zheng, J.; Luo, M. From Bitcoin to Solana—Innovating Blockchain Towards Enterprise Applications. In *Blockchain—ICBC 2021*; Springer: Cham, Switzerland, 2022; Volume 12991.
29. Houben, R.; Snyers, A. *Cryptocurrencies and Blockchain: Legal Context and Implications for Financial Crime, Money Laundering and Tax Evasion*; European Parliament: Strasbourg, France, 2018; p. 40.
30. Kiayias, A.; Russell, A.; David, B.; Oliynykov, R. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In Proceedings of the 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2017; Volume 10401.
31. Chegenizadeh, M.; Larionov, N.; Niya, S.R.; Yanovich, Y.; Tessone, C.J. Cardano Shared Send Transactions Untangling in Numbers. *Blockchain Res. Appl.* **2025**, *2025*, 100269. [CrossRef]
32. Ashraf, M.; Heavy, C. A prototype of supply chain traceability using Solana as blockchain and IoT. *Procedia Comput. Sci.* **2023**, *217*, 948–959. [CrossRef]
33. King, J.T.; Williams, L.A. Secure Logging and Auditing in Electronic Health Records Systems: What Can We Learn from the Payment Card Industry. In Proceedings of the 3rd USENIX Conference on Health Security and Privacy, Berkeley, CA, USA, 6–7 August 2012.
34. Gaynor, M.; Bass, C.; Duepner, B. A tale of two standards: Strengthening HIPAA security regulations using the PCI-DSS. *Health Syst.* **2015**, *4*, 111–123. [CrossRef]